# On the Power of Secure Two-Party Computation

Carmit Hazay[⋆] and Muthuramakrishnan Venkitasubramaniam[⋆⋆]

**Abstract.** Ishai, Kushilevitz, Ostrovsky and Sahai (STOC 2007, SIAM JoC 2009) introduced the powerful "MPC-in-the-head" technique that provided a general transformation of information-theoretic MPC protocols secure against passive adversaries to a ZK proof in a "black-box" way. In this work, we extend this technique and provide a generic transformation of any semi-honest secure two-party computation (2PC) protocol (with mild adaptive security guarantees) in the so called *oblivious-transfer* hybrid model to an *adaptive* ZK proof for any NP-language, in a "black-box" way assuming only one-way functions. Our basic construction based on Goldreich-Micali-Wigderson's 2PC protocol yields an adaptive ZK proof with communication complexity proportional to quadratic in the size of the circuit implementing the NP relation. Previously such proofs relied on an expensive Karp reduction of the NP language to Graph Hamiltonicity (Lindell and Zarosim (TCC 2009, Journal of Cryptology 2011)). We also improve our basic construction to obtain the first linear-rate adaptive ZK proofs by relying on efficient maliciously secure 2PC protocols. Core to this construction is a new way of transforming 2PC protocols to efficient (adaptively secure) instance-dependent commitment schemes.

As our second contribution, we provide a general transformation to construct a randomized encoding of a function $f$ from any 2PC protocol that securely computes a related functionality (in a black-box way). We show that if the 2PC protocol has mild adaptive security guarantees then the resulting randomized encoding (RE) can be decomposed to an offline/online encoding.

As an application of our techniques, we show how to improve the construction of Lapidot and Shamir (Crypto 1990) to obtain a four-round ZK proof with an "input-delayed" property. Namely, the honest prover's algorithm does not require the actual statement to be proved until the last round. We further generalize this to obtain a four-round "commit and prove" zero-knowledge with the same property where the prover commits to a witness $w$ in the second message and proves a statement $x$ regarding the witness $w$ that is determined only in the fourth round.

**Keywords:** adaptive zero-knowledge proofs, secure two-party computation, randomized encoding, interactive hashing, instance-dependent commitments

[⋆] Bar-Ilan University, Israel. Email: carmit.hazay@biu.ac.il. Research was partially supported by the European Research Council under the ERC consolidators grant agreement n. 615172 (HIPS), and by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Minister's Office.

[⋆⋆] University of Rochester, Rochester, NY 14611, NY. Email: muthuv@cs.rochester.edu. Research supported by Google Faculty Research Grant and NSF Award CNS-1526377.

## 1 Introduction

In this work we establish new general connections between three fundamental tasks in cryptography: secure two-party computation, zero-knowledge proofs and randomized encoding. We begin with some relevant background regarding each of these tasks.

*Secure multiparty computation.* The problem of *secure multiparty computation* (MPC) [Yao86,CCD87,GMW87,BGW88] considers a set of parties with private inputs that wish to jointly compute some function of their inputs while preserving certain security properties. Two of these properties are *privacy*, meaning that the output is learned but nothing else, and *correctness*, meaning that no corrupted party or parties can cause the output to deviate from the specified function. Security is formalized using the simulation paradigm where for every adversary $\mathcal{A}$ attacking a real protocol, we require the existence of a simulator $\mathcal{S}$ that can cause the same damage in an ideal world, where an incorruptible trusted third party computes the function for the parties and provides them their output.

*Honest vs. dishonest majority.* Generally speaking, there are two distinct categories for MPC protocols: (1) one for which security is guaranteed only when a *majority* of the parties are honest, and (2) one for which security is guaranteed against an arbitrary number of corrupted parties. In the former category it is possible to construct "information-theoretic" secure protocols where security holds unconditionally,[1] whereas in the latter only computational security can be achieved while relying on cryptographic assumptions.[2] The former setting necessarily requires 3 or more parties while the latter can be constructed with just two parties. In this work, we will focus on the latter setting, considering secure two-party computation.

*Semi-honest vs. malicious adversary.* The adversary may be *semi-honest*, meaning that it follows the protocol specification but tries to learn more than allowed, or *malicious*, namely, arbitrarily deviating from the protocol specification in order to compromise the security of the other players in the protocol. Constructing semi-honestly secure protocols is a much easier task than achieving security against a malicious adversary.

*Static vs. adaptive corruption.* The initial model considered for secure computation was one of a *static adversary* where the adversary controls a subset of the parties (who are called *corrupted*) before the protocol begins, and this subset cannot change. A stronger corruption model allows the adversary to choose which parties to corrupt throughout the protocol execution, and as a function of its view; such an adversary is called *adaptive*. Adaptive corruptions model "hacking" attacks where an external attacker breaks into parties' machines in the midst of a protocol execution and are much harder to protect against. In particular, protocols that achieve adaptivity are more complex and the computational hardness assumptions needed seem stronger; see [CLOS02] [KO04,CDD$^+$04,IPS08]. Achieving efficiency seems also to be much harder.

[1] Namely, against computationally unbounded adversaries.

[2] If one is willing to provide ideal access to an oblivious-transfer functionality then one can achieve information-theoretic security even in the honest minority setting [GMW87,CvdGT95,IPS08].

*Zero-knowledge.* Zero-knowledge (ZK) interactive protocols [GMR89] are paradoxical constructs that allow one party (denoted the prover) to convince another party (denoted the verifier) of the validity of a mathematical statement $x \in \mathcal{L}$, while providing *zero additional knowledge* to the verifier. Beyond being fascinating in their own right, ZK proofs have numerous cryptographic applications and are one of the most fundamental cryptographic building blocks. The zero-knowledge property is formalized using the *simulation paradigm*. That is, for every malicious verifier $\mathcal{V}^*$, we require the existence of a simulator $\mathcal{S}$ that reproduces a view of $\mathcal{V}^*$ that is indistinguishable from a view when interacting with the honest prover, given only the input $x$. Zero-knowledge protocols can be viewed as an instance of secure two-party computation where the function computed by the third-party simply verifies the validity of a witness held by the prover.

*Static vs. adaptive.* Just as with general secure computation, the adversary in a zero-knowledge protocol can be either static or adaptive. Security in the presence of a statically corrupted prover implies that the protocol is sound, namely, a corrupted prover cannot convince a verifier of a false statement. Whereas security in the presence of a statically corrupted verifier implies that the protocol preserves zero-knowledge. Adaptive security on the other hand requires a simulator that can simulate adaptive corruptions of both parties.

Much progress has been made in constructing highly efficient ZK proofs in the static setting. In a recent breakthrough result, Ishai, Kushilevitz, Ostrovsky and Sahai [IKOS09] provided general constructions of ZK proofs for any $\mathsf{NP}$ relation $\mathcal{R}(x, \omega)$ which make a "black-box" use of an MPC protocol for a related multiparty functionality $f$, where by black-box we mean that $f$ can be programmed to make only black-box (oracle) access to the relation $\mathcal{R}$. Leveraging the highly efficient MPC protocols in the literature [DI06] they obtained the first "constant-rate" ZK proof. More precisely, assuming one-way functions, they showed how to design a ZK proof for an arbitrary circuit $\mathrm{C}$ of size $s$ and bounded fan-in, with communication complexity $O(s) + \mathsf{poly}(\kappa, \log s)$ where $\kappa$ is the security parameter. Besides this, the work of [IKOS07,IKOS09] introduced the very powerful "MPC-in-the-head" technique that has found numerous applications in obtaining "black-box" approaches, such as unconditional two-party computation [IPS08], secure computation of arithmetic circuits [IPS09], non-malleable commitments [GLOV12], zero-knowledge PCPs [IW14], resettably-sound ZK [OSV15] to name a few, as well as efficient protocols, such as oblivious-transfer based cryptography [HIKN08,IPS08,IPS09] and homomorphic UC commitments [CDD$^+$15].

In contrast, in the adaptive setting, constructing adaptive zero-knowledge proofs is significantly harder and considerably less efficient. Beaver [Bea96] showed that unless the polynomial hierarchy collapses the ZK proof of [GMR89] is not secure in the presence of adaptive adversaries. Quite remarkably, Lindell and Zarosim showed in [LZ11] that adaptive zero-knowledge proofs for any $\mathsf{NP}$ language can be constructed assuming only one-way functions. However, it is based on reducing the statement that needs to be proved to an $\mathsf{NP}$ complete problem, and is rather inefficient. In fact, the communication complexity of the resulting zero knowledge is $O(s^4)$ where $s$ is the size of the circuit. A first motivation for our work is the goal of finding alternative approaches of constructing (efficient) adaptive ZK proofs without relying on the expensive Karp-reduction step.

*Randomized Encoding (RE).* The third fundamental primitive considered in this work is *randomized encoding* (RE). Formalized in the works of [IK00,IK02,AIK06], randomized encoding explores to what extent the task of securely computing a function can be simplified by settling for computing an "encoding" of the output. Loosely speaking, a function $\widehat{f}(x, r)$ is said to be a randomized encoding of a function $f$ if the output distribution depends only on $f(x)$. More formally, the two properties required of a randomized encoding are: (1) given the output of $\widehat{f}$ on $(x, r)$, one can efficiently compute (decode) $f(x)$, and (2) given the value $f(x)$ one can efficiently sample from the distribution induced by $\widehat{f}(x, r)$ where $r$ is uniformly sampled. One of the earliest constructions of a randomized encoding is that of "garbled circuits" and originates in the work of Yao [Yao86]. Additional variants have been considered in the literature in the early works of [Kil88,FKN94]. Since its introduction, randomized encoding has found numerous applications, especially in parallel cryptography where encodings with small parallel complexity yields highly efficient secure computation [IK00,IK02,AIK06]. (See also [GKR08,GGP10,AIK10,GIS$^+$10,BHHI10,BHR12,App14] for other applications).

*Statistical vs. Computational.* Randomized encodings can be statistical or computational depending on how close the sampled distribution is to the real distribution of $\widehat{f}$. While statistical randomized encodings exist for functions computable by $\mathsf{NC}^1$ circuits, only computational REs are known for general polynomial-time computable function. We refer the reader to [AIKP15] for a more detailed investigation on the class of languages that have statistical REs.

*Online/offline complexity.* In an online/offline setting [AIKW13], one considers an encoding $\widehat{f}(x, r)$ which can be split as an offline part $\widehat{f}_{\text{OFF}}(r)$ which only depends on the function $f$, and an online part $\widehat{f}_{\text{ON}}(x, r)$ that additionally depends on input $x$. This notion is useful in a scenario where a weak device is required to perform some costly operation $f$ on sensitive information $x$: In an offline phase $\widehat{f}_{\text{OFF}}(r)$ is published or transmitted to a cloud, and later in an online phase, the weak device upon observing the sample $x$, transmits the encoding $\widehat{f}_{\text{ON}}(x, r)$. The cloud then uses the offline and online parts to decode the value $f(x)$ and nothing else. The goal in such a setting is to minimize the online complexity, namely the number of bits in $\widehat{f}_{\text{ON}}(x, r)$. In the classic garbled circuit construction, the online complexity is proportional to $|x|\mathsf{poly}(\kappa)$ where $\kappa$ is the security parameter. More recently, Applebaum, Ishai, Kushilevitz and Waters showed in [AIKW13] how to achieve constant online rate of $(1 + o(1))|x|$ based on concrete number-theoretic assumptions.

A notoriously hard question here is to construct an *adaptively secure* RE where privacy is maintained even if the online input $x$ is *adaptively* chosen based on the offline part. In fact, the standard constructions of garbled circuits (with short keys) do not satisfy this stronger property unless some form of "exponentially-hard" assumption is made [GKR08] or analyzed in the presence of the so-called *programmable random-oracle* model [AIKW13]. In fact, it was shown in [AIKW13] that any adaptively secure randomized encoding must have an online complexity proportional to the output length of the function. The work of Hemenway [HJO$^+$15] provided the first constructions of adaptively-secure RE based on the minimal assumption of one-way functions.

While the connection between RE and secure computation has been explored only in one direction, where efficient RE yield efficient secure computation, we are not aware of any implication in the reverse direction. A second motivation of our work is to understand this direction while better understanding the complexity of constructing secure protocols by relying on the lower bounds established for the simpler RE primitive.

## 1.1 Our Contribution

In this work we present the following transformations:

1. A general construction of a *static* zero-knowledge proof system $\Pi_{\mathcal{R}}$ for any $\mathsf{NP}$ relation $\mathcal{R}(x, \omega)$ that makes a black-box use[3] of a two-party protocol $\Pi_f^{\mathrm{OT}}$, carried out between parties $P_1$ and $P_2$, for a related functionality $f$ in the oblivious-transfer (OT) hybrid model,[4] along with a (statically secure) bit commitment protocol,[5] that can be realized assuming only one-way functions. The requirement on our protocol $\Pi_f^{\mathrm{OT}}$ is: Perfect (UC) security against static corruptions by semi-honest adversaries. For example, the standard versions of the known [GMW87] protocol (denoted by GMW) and [Yao86]'s protocol satisfy these requirements.
2. A general construction of an *adaptively secure* zero-knowledge proof system $\Pi_{\mathcal{R}}$ for any $\mathsf{NP}$ relation $\mathcal{R}(x, \omega)$ that makes a black-box use of a two-party protocol $\Pi_f^{\mathrm{OT}}$, carried out between parties $P_1$ and $P_2$, for a related functionality $f$ in the oblivious-transfer (OT) hybrid model, along with a (statically secure) bit commitment protocol, that can be realized assuming only one-way functions. The requirements on our protocol $\Pi_f^{\mathrm{OT}}$ are: (1) Perfect (UC) security against semi-honest parties admitting a static corruption of $P_1$ and an adaptive corruption of $P_2$, and (2) $P_1$ is the sender in all OT invocations. We remark that the semi-honest version of the GMW protocol satisfies these requirements. In fact, we will only require milder properties than perfect privacy (namely, robustness and invertible sampleability) and adaptive corruption (namely, one-sided semi-adaptive [GWZ09]) which will be satisfied by the standard Yao's protocol [Yao86] based on garbled circuits.
3. A general construction of a *randomized encoding* for any function $f$ that makes a black-box use (a la [IKOS09]) of a two-party computation protocol $\Pi_f^{\mathrm{OT}}$, carried out between parties $P_1$ and $P_2$, for a related functionality $g$ in the OT-hybrid assuming only one-way functions. If we start with the same requirements as our first transformation (namely, only security against static adversaries) then we obtain a standard randomized encoding. However, if we start with a protocol as required in our second transformation with the additional requirement that it admits (full) adaptive corruption of $P_2$, we obtain an online/offline RE. Moreover, our construction makes a black-box use of a randomized encoding for the functionality $f$. Finally, we also show how to obtain an adaptive ZK proof for an $\mathsf{NP}$ relation $\mathcal{R}$ using a

[3] The functionality $f$ can be efficiently defined by making only a black-box (oracle) access to the $\mathsf{NP}$ relation $\mathcal{R}$.

[4] Where all parties have access to an idealized primitive that implements the OT functionality, namely, the functionality upon receiving input $(s_0, s_1)$ from the sender and a bit $b$ from the receiver, returns $s_b$ to the receiver and nothing the sender.

[5] We will be able to instantiate our commitment schemes using a statistically-binding commitment scheme for commitments made by the prover in the ZK protocol, and by a statistically-hiding commitment scheme for commitments made by the verifier.

slightly stronger version of RE (that our second instantiation above will satisfy). An important corollary we obtain here is that starting from an RE that is additionally secure against adaptive chosen inputs we obtain the—so called—input-delayed ZK proof in the static setting.

A few remarks are in order.

*Remark 1.* In transformations 2 and 3 we require the underlying 2PC protocol to be one-sided semi-adaptive (where the sender is statically corrupted, and the receiver is adaptively corrupted). This security notion is a weak requirement and almost all known protocols that are secure in the static setting are also semi-adaptive secure. Namely, the 2PC protocols based on [Yao86] and [GMW87] are one-sided semi-adaptive secure in our sense. In most cases, the semi-adaptive simulation can be accomplished by honestly generating the simulation of one party and then upon adaptive corruption of the other party, simulation can be accomplished by relying on the semi-adaptive simulation of OT calls (which in turn can be achieved using only one-way functions).

*Remark 2.* Our online/offline RE based on (semi-adaptive) 2PC protocols is efficient only for certain protocols. Looking ahead, the offline complexity of the resulting RE is proportional to the honest algorithm of party $P_1$ and the online complexity is proportional to the semi-adaptive simulation of party $P_2$. In the case of [Yao86], applying our transformation yields the standard RE based on garbled circuits. We note that while we do not obtain any new constructions of RE, our transformation relates the semi-adaptive simulation complexity of a protocol to the efficiency of a corresponding RE.

*Comparison with [IKOS09].* We remark that the approach of [IKOS09] that transforms general MPC protocols cannot be used "directly" to yield our first result concerning static ZK. This is because all constructions presented in their work require to instantiate the MPC protocol with at least three parties. In work subsequent to this, Ishai et. al [IKPY16] show how to extend the [IKOS09]-transformation to obtain our first result in a more communication efficient way. Our second and third tranformations, allows a strengthening of our first result to additionally achieves an input-delayed property. We obtain this stronger property by crucially relying on the semi-adaptive simulation. We remark that, both the approaches of [IKOS09] and [IKPY16] cannot yield such a protocol as the views of all parties are committed to by the prover in the first round and there is no mechanism to equivocate the views as required in the application. Another important distinction is that we only commit to the transcript of the interaction in the first round while [IKOS09] commits to each individual view. On the other hand, our approach cannot be applied to information theoretic protocols as the transcript of the interaction information theoretically binds the inputs and outputs of all parties.

## 1.2 Applications

We list a few of the applications of our techniques and leave it as future work to explore the other ramifications of our transformations.

COMMIT AND PROVE INPUT-DELAYED ZK PROOFS. In [LS90], a three-round witness-indistinguishable (WI) proof had been shown for Graph Hamiltonicity with a special "input-delayed" property: namely, the prover uses the statement to be proved only in the last round. Recently, in [CPS$^+$15] it was shown how to obtain efficient input-delayed variants of the related "Sigma protocols" when used in a restricted setting of an OR-composition. We show that starting from a robust RE that is additionally secure against adaptive inputs, we can obtain general constructions of input-delayed zero-knowledge proofs that yield an efficient version of the protocol of [LS90] for arbitrary $NP$-relations. We remark that our work is stronger than [CPS$^+$15] in that it achieves the stronger adaptive soundness property (which is satisfied by [LS90,FLS99]).The communication complexity in our protocol depends only linearly on the size of the circuit implementing the $\mathsf{NP}$ relation. As in our other transformation, this transformation will only depend on the relation in a black-box way. Finally, we show how to realize robust RE secure against adaptive inputs based on recent work of Hemenway et al. [HJO$^+$15].

The "commit-and-prove" paradigm considers a prover that first commits to a witness $w$ and then, in a second phase upon receiving a statement $x$ asserts whether a particular relation $R(x, w) = 1$ without revealing the committed value,. This paradigm implicit in the work of [GMW87], later formalized in [CLOS02], is a powerful mechanism to strengthen semi-honest secure protocols to maliciously secure ones. The MPC-in-the-head approach of [IKOS09] shows how to obtain a commit and prove protocol in the commitment-hybrid model thereby providing a construction that relies on the underlying commitment (in turn the one-way function) in a black-box way. This has been used extensively in several works to close the gap between black-box and non-black-box constructions relying on one-way functions (cf. [GLOV12,GOSV14,OSV15] for a few examples). We show that our input-delayed ZK proof further supports the commit-and-prove paradigm. In fact, using our approach, we provide the first constructions of commit-and-prove protocol with this property that relies on the underlying commitment functionality in a black-box way. Instantiating the underlying non-interactive commitment scheme with one-way permutation, we obtain a black-box construction of a 4-round commit and prove protocol with the input-delayed property.

INSTANCE-DEPENDENT TRAPDOOR COMMITMENT SCHEMES. As a side result, we show that our constructions imply instance-dependent trapdoor commitment schemes, for which the witness $\omega$ serves as a trapdoor that allows to equivocate the commitment into any value. Specifically, this notion implies the same hiding/binding properties as any instance-dependent commitment scheme with the additional property that the witness allows to decommit a commitment into any message. To the best of our knowledge, our construction is the first trapdoor commitment for all $\mathsf{NP}$. Prior constructions were known only for $\Sigma$-protocols [Dam10] and for Blum's Graph-Hamiltonicity [FS89].

### 1.3 Our Techniques

In this section, we provide an overview of our transformations and the techniques.

*Static ZK via (semi-honest) 2PC or "2PC-in-the-head".* We begin with a perfectly-correct 2PC protocol $\Pi_f$ between parties $P_1$ and $P_2$ that securely implements the following functionality $f$: $f(x, \omega_1, \omega_2)$ outputs 1 if and only if $(x, \omega_1 \oplus \omega_2) \in \mathcal{R}$ where $\omega_1$ and $\omega_2$ are the private inputs of $P_1$ and $P_2$ in the two party protocol $\Pi_f$. We require that the 2PC protocol admits semi-honest UC security against static corruption of $P_1$ and $P_2$. Our first step in constructing a ZK proof involves the prover $P$ simulating an honest execution between $P_1$ and $P_2$ by first sampling $\omega_1$ and $\omega_2$ at random such that $\omega_1 \oplus \omega_2 = \omega$, where $\omega$ is the witness to the statement $x$ and then submitting the transcript of the interaction to the verifier $V$. The verifier responds with a bit $b$ chosen at random. The prover then reveals the view of $P_1$ if $b = 0$ and the view of $P_2$ if $b = 1$, namely it just provides the input and randomness of the respective parties. Soundness follows from the perfect correctness of the protocol. Zero-knowledge, on the other hand, is achieved by invoking the simulation of parties $P_1$ and $P_2$ depending on the guess that the simulator makes for the verifier's bit $b$.

This general construction, however, will inherit the hardness assumptions required for the 2PC, which in the case of [Yao86] and [GMW87] protocols will require the existence of an oblivious-transfer protocol. We next show how to modify the construction to rely only on one-way functions. The high-level idea is that we encode the transcript of all oblivious-transfer invocations by using a "randomized encoding" of the oblivious-transfer functionality based on one-way functions as follows:

- For every OT call where $P_1$'s input is $(s_0, s_1)$ and $P_2$'s input is $t$, we incorporate it in the transcript $\tau$ by generating a transcript containing the commitments $c_0$ and $c_1$ of $s_0$ and $s_1$ using a statistically binding commitment scheme $\mathsf{com}$ (which can be based on one-way functions), placing the decommitment information of $c_t$ in $P_2$'s random tape.[6]

This protocol results in an interactive commitment phase as we rely on a statistically-binding commitment scheme and the first message corresponding to all commitments needs to be provided by the receiver.

Compared to [IKOS09,IPS08], we remark that our ZK proof does not provide efficiency gains (using OT-preprocessing) as we require a commitment for every oblivious-transfer and in the case of compiling [GMW87] results in $O(s)$ commitments where $s$ is the size of the circuit. Nevertheless, we believe that this compilation illustrates the simplicity of obtaining a ZK proof starting from a 2PC protocol.

*Adaptive ZK via "2PC-in-the-head".* First, we recall the work of Lindell and Zarosim [LZ11] that showed that constructing adaptively secure ZK proofs can be reduced to constructing *adaptive instance-dependent commitment* schemes [BMO90,IOS97,OV08] [LZ11]. In fact, by simply instantiating the commitments from the prover in the (static) ZK proofs of [IKOS09] with instance-dependent commitments, we can obtain an adaptive ZK proof. Briefly, instance-dependent commitment schemes are defined with respect to a language $\mathcal{L} \in \mathsf{NP}$ such that for any statement $x$ the following holds. If $x \in \mathcal{L}$ then the commitment associated with $x$ is computationally hiding, whereas if $x \notin \mathcal{L}$ then the commitment associated with $x$ is perfectly binding. An adaptively secure instance-dependent commitment scheme additionally requires that there be a

[6] Note that, in Naor's statistically binding commitment scheme [Nao91] the decommitment information is the inverse under a pseudorandom generator that is uniformly sampled, and hence can be placed in the random tape.

"fake" commitment algorithm which can be produced using only the statement $x$, but later, given a witness $\omega$ such that $(x,\omega) \in \mathcal{R}$, be opened to both 0 and 1.

First, we describe an instance-dependent commitment scheme using a (perfectly-correct) 2PC protocol $\Pi_f$ engaged between parties $P_1$ and $P_2$ that securely implements the following functionality $f$: $f(x,\omega_1,\omega_2)$ outputs 1 if and only if $(x,\omega_1 \oplus \omega_2) \in \mathcal{R}$ where $\omega_1$ and $\omega_2$ are the private inputs of $P_1$ and $P_2$ in the two party protocol $\Pi_f$. We will require that only $P_2$ receives an output and that $\Pi_f$ is (UC) secure against the following adversaries: (1) A semi-honest adversary $\mathcal{A}_1$ that statically corrupts $P_1$, and (2) A semi-honest adversary $\mathcal{A}_2$ that statically corrupts $P_2$.

Given such a 2PC $\Pi_f$ a commitment to the message 0 is obtained by committing to the view of party $P_1$ in an interaction using $\Pi_f$, using the simulator $\mathcal{S}_1$ for adversary $\mathcal{A}_1$ as follows. The commitment algorithm runs $\mathcal{S}_1$ on input a random string $\omega_1$ that serves as the input of $P_1$. The output of the commitment on input 0 is $\tau$ where $\tau$ is the transcript of the interaction between $P_1$ and $P_2$ obtained from the view of $P_1$ generated by $\mathcal{S}_1$. A commitment to 1 is obtained by running the simulator $\mathcal{S}_2$ corresponding to $\mathcal{A}_2$ where the input of $P_2$ is set to a random string $\omega_2$. The output of the commitment is transcript $\tau$ obtained from the view of $P_2$ output by $\mathcal{S}_2$. Decommitting to 0 simply requires producing input and output $(\omega_1, r_1)$ for $P_1$ such that the actions of $P_1$ on input $\omega_1$ and random tape $r_1$ are consistent with the transcript $\tau$. Decommitting to 1 requires producing input and randomness $(\omega_2, r_2)$ for $P_2$ consistent with $\tau$ and $P_2$ outputs 1 as the output of the computation. The hiding property of the commitment scheme follows from the fact that the transcript does not reveal any information regarding the computation (i.e. transcript can be simulated indistinguishably). The binding property for statements $x \notin \mathcal{L}$, on the other hand, relies on the perfect correctness of the protocol. More precisely, if a commitment phase $\tau$ is decommitted to both 0 and 1, then we can extract inputs and randomness for $P_1$ and $P_2$ such that the resulting interaction with honest behavior yields $\tau$ as the transcript of messages exchanged and $P_2$ outputting 1. Note that this is impossible since the protocol is perfectly correct and 1 is not in the image of $f$ for $x \notin \mathcal{L}$.

Next, to obtain an *adaptively secure* instance-dependent commitment scheme we will additionally require that $\Pi_f$ be secure against a semi-honest adversary $\mathcal{A}_3$ that first statically corrupts $P_1$ and then adaptively corrupts $P_2$ at the end of the execution. This adversary is referred to as a semi-adaptive adversary in the terminology of [GWZ09]. The fake commitment algorithm follows the same strategy as committing to 0 with the exception that it relies on the simulator $\mathcal{S}_3$ of $\mathcal{A}_3$. $\mathcal{S}_3$ is a simulator that first produces a view for $P_1$ and then post execution produces a view for $P_2$. More formally, the fake commitment algorithm sets $P_1$'s input to a random string $\omega_1$ and produces $P_1$'s view using $\mathcal{S}_3$ and outputs $\tau$ where, $\tau$ is the transcript of the interaction. Decommitting to 0 follows using the same strategy as the honest decommitment. Decommitting to 1, on the other hand, requires producing input and randomness for $P_2$. This can be achieved by continuing the simulation by $\mathcal{S}_3$ post execution. However, to run $\mathcal{S}_3$ it needs to produce an input for party $P_2$ such that it outputs 1. This is possible as the decommitting algorithm additionally receives the real witness $\omega$ for $x$, using which it sets $P_2$'s input as $\omega_2 = \omega \oplus \omega_1$.

In fact, we will only require adversaries $\mathcal{A}_2$ and $\mathcal{A}_3$, as the honest commitment to $0$ can rely on $\mathcal{S}_3$. Indistinguishability of the simulation will then follow by comparing the simulations by $\mathcal{S}_2$ and $\mathcal{S}_3$ with a real-world experiment with adversaries $\mathcal{A}_2, \mathcal{A}_3$ where the parties inputs are chosen at random subject to the condition that they add up to $\omega$ and using the fact that the adversaries are semi-honest.

We will follow an approach similar to our previous transformation to address calls to the OT functionality. We will additionally require that $P_1$ plays the sender's role in all OT invocations. We note that our encoding accommodates an adaptive corruption of $P_2$, as it enables us to equivocate the random tape of $P_2$ depending on its input $t$.

To instantiate our scheme, we can rely on [Yao86] or [GMW87] to obtain an adaptive instance-dependent commitment scheme. Both commitments results in a communication complexity of $O(s \cdot \mathsf{poly}(\kappa))$ where $s$ is the size of the circuit implementing the relation $\mathcal{R}$ and $\kappa$ is the security parameter. Achieving adaptive zero-knowledge is then carried out by plugging in our commitment scheme into the prover's commitments in the [IKOS09] zero-knowledge (ZK) construction, where it commits to the views of the underlying MPC protocol. The resulting protocol will have a complexity of $O(s^2 \cdot \mathsf{poly}(\kappa))$ and a negligible soundness error. We remark that this construction already improves the previous construction of Lindell and Zarosim that requires the expensive Karp reduction to Graph Hamiltonicity. Our main technical contribution is showing how we can further improve our basic construction to achieve a complexity of $O(s \cdot \mathsf{poly}(\kappa))$ and therefore obtaining a "linear"-rate *adaptive* ZK proof.

*RE from (semi-honest) 2PC.* To construct a RE for a function $f$, we consider an arbitrary 2PC protocol that securely realizes the related function $g$ that is specified as follows: $g(a_1, a_2) = f(a_1 \oplus a_2)$ where $a_1$ and $a_2$ are the private inputs of $P_1$ and $P_2$ in the two party protocol $\Pi_g$. We will make the same requirements on our 2PC as in the previous case, namely, security with respect to adversaries $\mathcal{A}_1$ and $\mathcal{A}_2$. The offline part of our encoding function $\widehat{f}_{\mathrm{OFF}}(r)$ is defined using the simulator $\mathcal{S}_3$ for adversary $\mathcal{A}_3$ that proceeds as follows. Upon corrupting $P_1$, $\mathcal{S}_3$ is provided with a random input string $a_1$, where the simulation is carried out till the end of the execution and temporarily stalled. The output of $\widehat{f}_{\mathrm{OFF}}(r)$ is defined to be the simulated transcript of the interaction between parties $P_1$ and $P_2$. Next, upon receiving the input $x$, the online part $\widehat{f}_{\mathrm{ON}}(x, r)$ continues the simulation by $\mathcal{S}_1$ which corrupts $P_2$ post execution (at the end of the protocol execution), where $P_2$'s input is set as $a_2 = x \oplus a_1$ and its output is set as $f(x)$. Finally, the output of $\widehat{f}_{\mathrm{ON}}(x, r)$ is defined by the input and random tape of $P_2$. In essence, $\widehat{f}(x, r) = (\widehat{f}_{\mathrm{OFF}}(r), \widehat{f}_{\mathrm{ON}}(x, r))$ constitutes the complete view of $P_2$ in an execution using $\Pi_g$. The decoder simply follows $P_2$'s computation in the view and outputs $P_2$'s output, which should be $f(x)$ by the correctness of the algorithm. The simulation for our randomized encoding $\mathcal{S}$ relies on the simulator for the adversary $\mathcal{A}_2$, denoted by $\mathcal{S}_2$. Namely, upon receiving $f(x)$, $\mathcal{S}$ simply executes $\mathcal{S}_2$. Recalling that $\mathcal{S}_2$ corrupts $P_2$, $\mathcal{S}$ simply provides a random string $a_2$ as its input and $f(x)$ as the output. Finally, the offline and online parts are simply extracted from $P_2$'s view accordingly. Privacy will follow analogously as in our previous case.

Note that the offline complexity of our construction is equal to the communication complexity of the underlying 2PC protocol $\Pi_g$, whereas the online complexity amounts to the input plus the randomness complexity of $P_2$. The efficiency of our randomized encoding ties the offline part with the static simulation of party $P_1$ and the online part with the semi-adative simulation of $P_2$. Moreover, this protocol can be instantiated by the [Yao86] and [GMW87] protocols, where the OT sub-protocols are implemented using one-way functions as specified before. We remark that the protocol of [Yao86] does not, in general, admit adaptive corruptions, yet it is secure in the presence of a semi-adaptive adversary that adaptively corrupts $P_2$ after statically corrupting $P_1$. The [Yao86] based protocol will result in an offline complexity of $O(s \cdot \mathsf{poly}(\kappa))$ and an online complexity of $O(n \cdot \mathsf{poly}(\kappa))$ where $s$ is the size of the circuit implementing $f$ and $n$ is the input length.[7] Whereas the [GMW87] protocol will result in an offline and online complexities of $O(s \cdot \mathsf{poly}(\kappa))$. While this might not be useful in the "delegation of computation" application of randomized encoding as the online encoding is not efficient, it can be used to construct an instance-dependent commitment scheme where we are interested only in the total complexity of the encoding. Finally, we remark that if we are not interested in an offline/online setting and just require a standard randomized encoding we will requite $\Pi_f$ to be secure only against a static corruption of $P_2$ by $\mathcal{A}_2$ and the honest encoding can be carried out by emulating the real world experiment (as opposed to relying on the simulation by $\mathcal{S}_3$).

Next, we provide a construction of instance-dependent commitments based on online/offline RE. Standard RE will not be sufficient for this and we introduce a stronger notion of *robustness* for RE and show that the preceeding construction already satisfies this. Then based on a robust RE we show how to get an instant-dependent commitment scheme. In fact, we can get an adaptive instance-dependent commitment scheme if the underlying RE has a corresponding adaptive property. Since adaptive instance-dependent comitment schemes are sufficient to realize adaptive ZK, this provides a transformation from RE to adaptive ZK.

*"Linear"-rate adaptive ZK proof from malicious 2PC.* The main drawback in our first construction of adaptive ZK proofs was in the equivocation parameter of our instance-dependent commitment. Namely, to equivocate one bit, we incurred a communication complexity of $O(s \cdot \mathsf{poly}(\kappa))$. To improve the communication complexity one needs to directly construct an instance-dependent commitment scheme for a larger message space $\{0,1\}^\ell$. We show how to construct a scheme where the communication complexity depends only *additively* on the equivocation parameter, implying $O((s+\ell)\mathsf{poly}(\kappa))$ overhead. Combining such a scheme with the [IKOS09] ZK proof results in a protocol with communication complexity of $O(n \cdot s \cdot \mathsf{poly}(\kappa) + \sum_{i=1}^{n} \ell_i \cdot \mathsf{poly}(\kappa))$ where $\ell_i$ is the length of the $i^{th}$ commitment made by the prover. Setting $n = \omega(\log k)$ results and using $\sum_i \ell_i = s \cdot \mathsf{poly}(\kappa)$ in an adaptive ZK proof with negligible soundness error and complexity $O(s \cdot \mathsf{poly}(\kappa))$. We remark here that by linear rate, we mean we obtain a protocol whose communication complexity that depends linearly on the circuit size. This stands in contrast of the previous approach by Lindell and Zarosim [LZ11] that depends at least cubic in the circuit size. In comparison, for the static case, [IKOS09] provide a "constant" rate Static ZK proof, i.e. a ZK proof whose communication complexity is $O(s + \mathsf{poly}(k))$.

[7] We note that the online complexity can be improved by relying on the work of [AIKW13].

Our approach to construct an instance-dependent commitment scheme for larger message spaces is to rely on a maliciously secure two-party computation. Specifically, suppose that for a polynomial-time computable Boolean function $f(x, y)$ we have a 2PC protocol $\Pi_f$ with parties $P_1$ and $P_2$, where $P_2$ receives the output of the computation and satisfies all the conditions required in our original transformation. In addition we require it to satisfy statistical security against a malicious $P_1$ (in the OT-hybrid). In fact, it suffices for the protocol to satisfy the following "soundness" condition: If there exists no pair of inputs $x, y$ such that $f(x, y) = 1$ then for any malicious $P_1^*$, the probability that an honest $P_2$ outputs 1 is at most $2^{-t}$, where the probability is taken over the randomness of party $P_2$. Then, using such a protocol, we can provide a framework to construct an instance-dependent commitment scheme where the soundness translates to the equivocation parameter, namely, it will be $O(t)$ for soundness $2^{-t}$.

Concretely, given an input statement $x$ we consider a protocol $\Pi_f$ that realizes function $f$ defined by: $f(\omega_1, \omega_2) = 1$ iff $(x, \omega_1 \oplus \omega_2) \in \mathcal{R}$. We first describe an (incorrect) algorithm as a stepping stone towards explaining the details of the final construction. The commitment algorithm on input a message $m$, (just as in our transformation to RE) invokes the simulator $\mathcal{S}_2$ that corresponds to the adversary $\mathcal{A}_2$, which statically corrupts $P_2$ with an input set to a random string $\omega_2$ and output 1. Upon completing the simulation, the committer submits to the receiver the transcript of the interaction and $\mathsf{Ext}(r_2) \oplus m$ where $r_2$ is the randomness of $P_2$ output by the simulation and $\mathsf{Ext}(\cdot)$ is a randomness extractor that extracts $R - \Omega(t)$ bits where $R$ is the length of $P_2$'s random tape. A decommitment simply produces $m$ along with $P_2$'s input and randomness corresponding to the transcript output in the commitment phase. Intuitively, binding follows directly from the soundness condition as no adversarial committer can produce two different random strings for $P_2$, as the entropy of all "accessible" random tapes for $P_2$ is "extracted" out by Ext.[8] The fake commitment, on the other hand, relies as above on a simulator corresponding to $\mathcal{A}_1$ that statically corrupts $P_1$ and adaptively corrupts $P_2$, where instead of $\mathsf{Ext}(r_2) \oplus m$ it simply sends a random string. Equivocation, on the other hand, is achievable if the simulation can additionally access the entire space of consistent random tapes of $P_2$ and invert Ext. Several problems arise when materializing this framework.

The first issue is that we cannot rely on an extractor as the adversary can adaptively decide on $r_2$ given the description of Ext. Now, since extractors are only statistically secure, this implies that for certain (very small) set of values for $r_2$ there could be multiple pre-images with respect to Ext. Instead, we rely on an *interactive hashing* protocol [NOVY98,DHRS04,HR07] that guarantees binding against computationally unbounded adversaries. More precisely, an interactive hashing protocol ensures that if the set of random tapes accessible to the adversary is at most $2^{R-\Omega(t)}$ then except with negligible probability it cannot obtain two random tapes that are consistent with the transcript of the hashing protocol. This protocol will additionally require to satisfy an invertible sampleability property where given an interaction it is possible to compute efficiently a random input consistent with the transcript. We will not be able to rely on the efficient 4-message protocol of [DHRS04] but will rely on the original protocol of

[8] This is not entirely accurate and is presented just for intuition. More details are presented in next paragraph.

[NOVY98] that proceeds in a linear number of rounds (linear in the message length) where inverting simply requires solving a system of linear equations in a finite field.

Another major issue is that the space of consistent random tapes might not be "nice" to be invertible. Namely, to adaptively decommit a fake commitment to an arbitrary message we require that the space of consistent random tapes for $P_2$, i.e. consistent with the transcript $\tau$ of the protocol and the transcript of the interactive-hashing protocol in the commitment phase, to be "uniform" over a nice domain . We thus consider a variant of the protocol in [IPS08] so that the space of consistent random tapes will be uniform over the bits of a specified length. While this modification solves the problem of "nice" random tapes, it requires re-establishing a certain "soundness" condition in the compilation of [IPS08].

As mentioned before we combine our adaptive instance-dependent commitment scheme with the ZK protocol of [IKOS09]. We will rely on a variant where the MPC protocol in their construction will be instantiated with the classic [BGW88] protocol, as opposed to highly-efficient protocol of [DI06]. The reason is that we will additionally require a reconstructability property[9] of the MPC protocol that can be shown to be satisfied by [BGW88]. Secondly, relying on this efficient variant anyway does not improve the asymptotic complexity to beyond a linear-rate. As an independent contribution we also provide a simple adaptive ZK protocol based on garbled circuits that satisfies reconstructability but will only achieve soundness error $1/2$ (see Section 6).

## 1.4 Perspective

Our work is similar in spirit to the work of [IKOS09,IPS08] that demonstrated the power information-theoretic MPC protocols in constructing statically-secure protocols. Here, we show the power of (adaptively-secure) 2PC protocols in the OT-hybrid helps in constructing adaptively-secure protocols and randomized encodings. Instantiating our 2PC with the standard protocols of [Yao86] and [GMW87] yields simple constructions of adaptive ZK proofs and randomized encodings. While ZK can be viewed as a special instance of a two-party computation protocol, the resulting instantiation requires stronger assumptions (such as enhanced trapdoor permutations). On the other hand, our transformation requires only one-way functions. As mentioned earlier, we not only provide adaptive ZK proofs, but we obtain two new simple static ZK proofs from our instance-based commitments.

A second contribution of our construction shows a useful class of applications for which 2PC protocols can be used to reduce the round complexity of black-box constructions. The well known and powerful "MPC-in-the-head" technique has found extensive applications in obtaining black-box construction of protocols that previously depended on generic Karp reductions. In many cases their approach was used to close the gap between black-box and non-black-box constructions. In particular, their approach provided the first mechanism to obtain a commit-and-prove protocol that depended on the underlying commitment in a black-box way. We believe that our technique yields an

[9] Informally, reconstructability requires that given the views of $t$ out of $n$ players in an instance of the protocol, and the inputs of all parties, it is possible to reconstruct the views of the remaining parties consistent with views of the $t$ parties.

analogous “2PC-in-the-head” technique which in addition to admitting similar commit-and-prove protocols can improve the round complexity as demonstrated for the case of non-malleable commitments. This is because of the input-delayed property that is achievable for our commit-and-prove protocols.

In addition, we believe it will be useful in applications that rely on certain special properties of the Blum’s Graph-Hamiltonicity ZK proof (BH). Concretely, we improve the [LZ11] adaptive ZK proof and the input-delayed protocol from [LS90] both of which relied on BH ZK proof. More precisely, by relying on our ZK proof based on our instance-dependent commitment schemes that, in turn, depends on the $\mathsf{NP}$ relation in a black-box way, we save the cost of the expensive Karp reduction to Graph Hamiltonicity. We leave it as future work to determine if other applications that rely on the BH ZK proof can be improved (e.g., NIZK).

## 2 Preliminaries

We denote the security parameter by $\kappa$. We say that a function $\mu : \mathbb{N} \to \mathbb{N}$ is *negligible* if for every positive polynomial $p(\cdot)$ and all sufficiently large $\kappa$’s it holds that $\mu(\kappa) < \frac{1}{p(\kappa)}$. We use the abbreviation PPT to denote probabilistic polynomial-time. For an $\mathsf{NP}$ relation $\mathcal{R}$, we denote by $\mathcal{R}_x$ the set of witnesses of $x$ and by $\mathcal{L}_\mathcal{R}$ its associated language. That is, $\mathcal{R}_x = \{\omega \mid (x,\omega) \in \mathcal{R}\}$ and $\mathcal{L}_\mathcal{R} = \{x \mid \exists\, \omega\ s.t.\ (x,\omega) \in \mathcal{R}\}$.

### 2.1 Adaptive Instance-Dependent Commitment Schemes [LZ11]

We extend the instance-dependent commitment scheme definition of [LZ11], originally introduced for the binary message space, to an arbitrary message space $\mathcal{M}$.

*Syntax.* Let $\mathcal{R}$ be an $\mathsf{NP}$ relation and $\mathcal{L}$ be the language associated with $\mathcal{R}$. A (non-interactive) adaptive instance dependent commitment scheme (AIDCS) for $\mathcal{L}$ is a tuple of probabilistic polynomial-time algorithms $(\mathsf{Com}, \mathsf{Com}', \mathsf{Adapt})$, where:

- $\mathsf{Com}$ is the commitment algorithm: For a message $m \in \mathcal{M}_n$, an instance $x \in \{0,1\}^*$, $|x| = n$ and a random string $r \in \{0,1\}^{p(|x|)}$ (where $p(\cdot)$ is a polynomial), $\mathsf{Com}(x, m; r)$ returns a commitment value $c$.
- $\mathsf{Com}'$ is a “fake” commitment algorithm: For an instance $x \in \{0,1\}^*$ and a random string $r \in \{0,1\}^{p(|x|)}$, $\mathsf{Com}'(x; r)$ returns a commitment value $c$.
- $\mathsf{Adapt}$ is an adaptive opening algorithm: Let $x \in \mathcal{L}$ and $\omega \in \mathcal{R}_x$. For all $c$ and $r \in \{0,1\}^{p(|x|)}$ such that $\mathsf{Com}'(x; r) = c$, and for all $m \in \mathcal{M}_n$, $\mathsf{Adapt}(x, \omega, c, m, r)$ returns a pair $(m, r')$ such that $c = \mathsf{Com}(x, m; r')$. (In other words, $\mathsf{Adapt}$ receives a “fake” commitment $c$ and a message $m$, and provides an explanation for $c$ as a commitment to the message $m$.)

*Security.* We now define the notion of security for our commitment scheme.

**Definition 21 (AIDCS)** *Let $\mathcal{R}$ be an $\mathsf{NP}$ relation and $\mathcal{L} = \mathcal{L}_\mathcal{R}$. We say that* $(\mathsf{Com}, \mathsf{Com}', \mathsf{Adapt})$ *is a secure AIDCS for $\mathcal{L}$ if the following holds:*

1. *Computational hiding: The ensembles* $\{\mathsf{Com}(x, m)\}_{x \in \mathcal{L}, m\{0,1\}^{|x|}}$, *and* $\{\mathsf{Com}'(x)\}_{x \in \mathcal{L}}$ *are computationally indistinguishable.*
2. *Adaptivity: The distributions* $\{\mathsf{Com}(x, m; U_{p(|x|)}), m, U_{p(|x|)}\}_{x \in \mathcal{L}, \omega \in \mathcal{R}_{\mathcal{L}}, m \in \{0,1\}^{|x|}}$ *and*
$\{\mathsf{Com}'(x; U_{p(|x|)}), m, \mathsf{Adapt}(x, \omega, \mathsf{Com}'(x; U_{p(|x|)}), m)\}_{x \in \mathcal{L}, \omega \in \mathcal{R}_{\mathcal{L}}, m \in \{0,1\}^{|x|}}$ *are computationally indistinguishable (that is, the random coins that are generated by* Adapt *are indistinguishable from real random coins used by the committing algorithm* Com*).*
3. *Statistical binding: For all* $x \notin \mathcal{L}$, $m, m' \in \mathcal{M}_{|x|}$, *and a commitment* $c$, *the probability that there exist* $r, r'$ *for which* $c = \mathsf{Com}(x, m; r)$ *and* $c = \mathsf{Com}(x, m'; r')$ *is negligible in* $\kappa$.

## 2.2 Zero-Knowledge Proofs

**Definition 22 (Interactive proof system)** *A pair of* PPT *interactive machines* $(\mathcal{P}, \mathcal{V})$ *is called an* interactive proof system for a language $\mathcal{L}$ *if there exists a negligible function* negl *such that the following two conditions hold:*

1. COMPLETENESS: *For every* $x \in \mathcal{L}$,

$$\Pr[\langle \mathcal{P}, \mathcal{V} \rangle(x) = 1] \geq 1 - \mathsf{negl}(|x|).$$

2. SOUNDNESS: *For every* $x \notin L$ *and every interactive* PPT *machine* $B$,

$$\Pr[\langle B, \mathcal{V} \rangle(x) = 1] \leq \mathsf{negl}(|x|).$$

**Definition 23 (Zero-knowledge)** *Let* $(\mathcal{P}, \mathcal{V})$ *be an interactive proof system for some language* $\mathcal{L}$. *We say that* $(\mathcal{P}, \mathcal{V})$ *is* computational zero-knowledge *if for every* PPT *interactive machine* $\mathcal{V}^*$ *there exists a* PPT *algorithm* $\mathcal{S}$ *such that*

$$\{\langle \mathcal{P}, \mathcal{V}^* \rangle(x)\}_{x \in \mathcal{L}} \overset{c}{\approx} \{\langle \mathcal{S} \rangle(x)\}_{x \in \mathcal{L}}$$

*where the left term denote the output of* $\mathcal{V}^*$ *after it interacts with* $\mathcal{P}$ *on common input* $x$ *whereas, the right term denote the output of* $\mathcal{S}$ *on* $x$.

*Input-delayed zero-knowledge proofs.* We will construct zero-knowledge proofs with an "input-delayed" property. Roughly speaking, this property allows an honest prover to generate all messages except from the last one, without knowledge of the statement. In such a situation, the soundness and zero-knowledge properties can additionally be required to be *adaptively* secure. Namely, soundness is required to hold even if the cheating prover adaptively chooses the statement (before the last message). Zero-knowledge, in the other hand, is required to hold even if the malicious verifier chooses a (true) statement before the last round.

*Adaptive zero-knowledge.* This notion considers the case for which the prover is adaptively corrupted. Loosely speaking, the simulator obtains a statement $x \in \mathcal{L}$. Moreover, at any point of the execution, the adaptive adversary is allowed to corrupt the prover. It is then required that zero-knowledge holds even in the presence of an adaptive adversary.

## 2.3 Garbled Circuits

Our notion of garbled circuits includes an additional algorithm of oblivious generation of a garbled circuit. Namely, given the randomness used to produce a garbled circuit $\widetilde{\mathrm{C}}$ of some circuit C, the algorithm generates new randomness that explains $\widetilde{\mathrm{C}}$ as the outcome of the simulated algorithm. We note that this modified notion of garbled circuits can be realized based on one-way functions, e.g., the construction from [LP09], for instance when the underlying symmetric key encryption used for garbling has an additional property of oblivious ciphertext generation (where a ciphertext can be sampled without the knowledge of the plaintext). Then the simulated garbling of a gate produces a garbled table using three obliviously generated ciphertexts and one ciphertext that encrypts the output label. We note that the ability to switch from a standard garbled circuit to a simulated one will be exploited in our constructions below in order to equivocate a commitment to 0 into a commitment to 1. Towards introducing our definition of garbled circuits we denote vectors by bold lower-case letters and use the parameter $n$ to denote the input and output length for the Boolean circuit C.

**Definition 24 (Garbling scheme)** *A garbling scheme* $\mathsf{Garb} = (\mathsf{Grb}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *consists of four polynomial-time algorithms that work as follows:*

- $(\widetilde{\mathrm{C}}, \mathbf{dk}, \mathsf{sk}) \leftarrow \mathsf{Grb}(1^\kappa, \mathrm{C}; r_{\mathsf{Grb}})$*: is a probabilistic algorithm with randomness* $r_{\mathsf{Grb}}$ *that takes as input a circuit* C *with* $2n$ *input wires and* $n$ *output wires and returns a garbled circuit* $\widetilde{\mathrm{C}}$*, a set of decoding keys* $\mathbf{dk} = (\mathrm{dk}_1, \dots, \mathrm{dk}_n)$ *and a secret key* $\mathsf{sk}$.
- $\widetilde{\mathbf{x}} := \mathsf{Enc}(\mathsf{sk}, \mathbf{x})$ *is a deterministic algorithm that takes an input a secret key* $\mathsf{sk}$*, an input* $\mathbf{x}$ *and returns an encoded input* $\widetilde{\mathbf{x}}$*. We denote this algorithm by* $\widetilde{\mathbf{x}} := \mathsf{Enc}(\mathsf{sk}, \widetilde{\mathbf{x}})$*. In this work we consider* decomposable garbled schemes. *Namely, the algorithm takes multiple input bits* $\mathbf{x} = (x_1, \dots, x_n)$*, runs* $\mathsf{Enc}(\mathsf{sk}, \cdot)$ *on each* $x_i$ *and returns the garbled inputs* $\widetilde{x}_1$ *through* $\widetilde{x}_n$*, denoted by input labels.*
- $\widetilde{\mathbf{y}} := \mathsf{Eval}(\widetilde{\mathrm{C}}, \widetilde{\mathbf{x}})$*: is a deterministic algorithm that takes as input a garbled circuit* $\widetilde{\mathrm{C}}$ *and encoded inputs* $\widetilde{\mathbf{x}}$ *and returns encoded outputs* $\widetilde{\mathbf{y}}$.
- $\{\perp, y_i\} := \mathsf{Dec}(\mathrm{dk}_i, \widetilde{y}_i)$*: is a deterministic algorithm that takes as input a decoding key* $\mathrm{dk}_i$ *and an encoded output* $\widetilde{y}_i$ *and returns either the failure symbol* $\perp$ *or an output* $y_i$*. We write* $\{\perp, \mathbf{y}\} := \mathsf{Dec}(\mathbf{dk}, \widetilde{\mathbf{y}})$ *to denote the algorithm that takes multiple garbled outputs* $\widetilde{\mathbf{y}} = (\widetilde{y}_1 \dots \widetilde{y}_n)$*, runs* $\mathsf{Dec}(\mathrm{dk}_i, \cdot)$ *on each* $\widetilde{y}_i$ *and returns the outputs* $y_1$ *through* $y_n$.

*Correctness.* We say that Garb is correct if for all $n \in \mathbb{N}$, for any polynomial-size circuit C, for all inputs $\mathbf{x}$ in the domain of C, for all $(\widetilde{\mathrm{C}}, \mathbf{dk}, \mathsf{sk})$ output by $\mathsf{Grb}(1^\kappa, \mathrm{C})$, for $\widetilde{\mathbf{x}} := \mathsf{Enc}(\mathsf{sk}, \mathbf{x})$ and $\widetilde{\mathbf{y}} := \mathsf{Eval}(\widetilde{\mathrm{C}}, \widetilde{\mathbf{x}})$ and for all $i \in [n]$, $y_i := \mathsf{Dec}(\mathrm{dk}_i, \widetilde{y}_i)$, where $(y_1, \dots, y_n) = \mathrm{C}(\mathbf{x})$.

*Security.* We say that a garbling scheme Garb is secure if there exists a PPT algorithm SimGC such that for any polynomial-size circuit C, for all inputs $\mathbf{x}$ in the domain of C, for all $(\widetilde{\mathrm{C}}, \mathbf{dk}, \mathsf{sk})$ output by $\mathsf{Grb}(1^\kappa, \mathrm{C})$ and $\widetilde{\mathbf{x}} := \mathsf{Enc}(\mathsf{sk}, \mathbf{x})$ it holds that,

$$(\widetilde{\mathrm{C}}, \widetilde{\mathbf{x}}, \mathbf{dk}) \overset{\mathrm{c}}{\approx} \mathsf{SimGC}\left(1^\kappa, \mathrm{C}, \mathbf{y}\right), where\ \ \mathbf{y} = \mathrm{C}(\mathbf{x}).$$

*Oblivious sampling.* There exists a PPT algorithm $\mathsf{OGrb}$ such that for any polynomial-time circuit $\mathrm{C}$ and for all input/output pairs $(\mathbf{x}, \mathbf{y})$ such that $\mathrm{C}(\mathbf{x}) = \mathbf{y}$ it holds that,

$$\{r'_{\mathsf{Grb}}, \mathsf{SimGC}\left(1^\kappa, \mathrm{C}, \mathbf{y}; r'_{\mathsf{Grb}}\right)\}_{r'_{\mathsf{Grb}} \leftarrow \{0,1\}^*} \overset{\mathrm{c}}{\approx} \{\hat{r}_{\mathsf{Grb}}, \widetilde{\mathrm{C}}, \tilde{x}, \mathbf{dk}\}_{(\hat{r}_{\mathsf{Grb}}, \tilde{x}) \leftarrow \mathsf{OGrb}(1^\kappa, \mathrm{C}, \mathbf{x}, r_{\mathsf{Grb}})}$$

where $(\widetilde{\mathrm{C}}, \mathbf{dk}, \mathsf{sk}) \leftarrow \mathsf{Grb}(1^\kappa, \mathrm{C}; r_{\mathsf{Grb}})$.

Note that correctness is perfect by our definition, which implies that a garbled circuit must be evaluated to the correct output. We further note that this notion is achieved by employing the point-and-permute optimization [PSSW09] to the garbling construction, as the evaluator of an honestly generated circuit always decrypts a single ciphertext for each gate which leads to the correct output. Furthermore, we assume that giving the secret key it is possible to verify that the garbled circuit was honestly generated. Again, this holds with respect to existing garbling schemes, as the secret key includes the encoding of all input labels which allows to recompute the entire garbling and verifying the correctness of each gate.

## 2.4 Randomized Encoding

We review the definition of randomized encoding from [IK00,AIK04].

**Definition 25 (Randomized Encoding)** *Let $f : \{0,1\}^n \to \{0,1\}^\ell$ be a function. Then a function $\widehat{f} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^s$ is said to be a* randomized encoding *of $f$, if:*

**Correctness:** *There exists a decoder algorithm $B$ such that for any input $x \in \{0,1\}^n$, except with negligible probability over the randomness of the encoding and the random coins of $B$, it holds that $B(\widehat{f}(x, U_m)) = f(x)$.*

**Computational (statistical) privacy:** *There exists a* PPT *simulator $\mathcal{S}$, such that for any input $x \in \{0,1\}^n$ the following distributions are computationally (statistically) indistinguishable over $n \in \mathbb{N}$:*
- $\{\widehat{f}(x, U_m)\}_{n \in \mathbb{N}, x \in \{0,1\}^n}$,
- $\{\mathcal{S}(f(x))\}_{n \in \mathbb{N}, x \in \{0,1\}^n}$.

We require our randomized encoding to satisfy some additional properties:

1. **Robustness:** Applebaum et al. introduced in [AIKW13] the measures of *offline* and *online* complexities of an encoding, where the offline complexity refers to the number of bits in the output of $\widehat{f}(x, r)$ that solely depend on $r$ and the online complexity refers to the number of bits that depend on both $x$ and $r$. The motivation in their work was to construct *online efficient* randomized encoding, where the online complexity is close to the input size of the function. In our construction, we are not concerned specifically with the online complexity, but we require that there exists an offline part of the randomized encoding that additionally satisfies a robustness property. We present the definition of robustness for boolean functions $f$ as it suffices for our construction.
We say that $\widehat{f}$ is a *robust encoding* of $f$ if there exist functions $\widehat{f}_{\mathrm{OFF}}$ and $\widehat{f}_{\mathrm{ON}}$ such that $\widehat{f}(x, r) = (\widehat{f}_{\mathrm{OFF}}(r), \widehat{f}_{\mathrm{ON}}(x, r))$ and, in addition, it holds that: if there exists

no $x$ such that $f(x) = 1$, then for any $r$, there exists no $z$ such that $B(\widehat{f}_{\text{OFF}}(r), z)$ outputs 1. Intuitively, robustness ensures that if the offline part was honestly computed using $\widehat{f}_{\text{OFF}}$ then there cannot exist any online part that can make the decoder output an element not in the range of the function $f$. We remark that it is possible to rewrite any randomized encoding as $(\widehat{f}_{\text{OFF}}(r), \widehat{f}_{\text{ON}}(x, r))$ for some functions $\widehat{f}_{\text{OFF}}$ and $\widehat{f}_{\text{ON}}$ (for instance, by setting $\widehat{f}_{\text{OFF}}$ to be the function that outputs the empty string and $\widehat{f}_{\text{ON}} = \widehat{f}$). Nevertheless, in order for the encoding to be robust there must exist a way to split the output bits of $\widehat{f}(x, r)$ into an offline part $\widehat{f}_{\text{OFF}}(r)$ and online part $\widehat{f}_{\text{ON}}(x, r)$ such that they additionally satisfy the robustness property. As mentioned before, it will not always be important for us to minimize the online complexity, where instead we require that the encoding is robust while minimizing the total (online+offline) complexity. We note that our definition is in the spirit of the authenticity definition with respect to garbled schemes from [BHR12].

2. **Oblivious sampling:** We require an additional oblivious property, as for the definition of garbling schemes, (that, looking ahead, will enable equivocation in our instance-dependence commitment schemes where a randomized encoding of function $f$ can be explained as a simulated encoding). We denote this algorithm by $\mathsf{ORE}$ and define this new security property as follows.
   For any function $f$ as above and for all input/output pairs $(x, y)$ such that $f(x) = y$ it holds that, $\{r', \mathcal{S}(y; r')\}_{r' \leftarrow \{0,1\}^*} \overset{c}{\approx} \{r', \widehat{f}_{\text{OFF}}(r), \widehat{f}_{\text{ON}}(x, r)\}_{r' \leftarrow \mathsf{ORE}(x,r)}$ where $r$ is the randomness for generating $\widehat{f}$.

In Section 5, we show how to realize a robust randomized encoding scheme based on any two-party computation protocol (that meets certain requirements), which, in particular is satisfied by the [Yao86] and [GMW87] protocols. While this construction does not achieve any "non-trivial" online complexity, it will be sufficient for our application, as the total complexity will be $O(s\kappa)$. We note that garbling schemes meet our definition of robust randomized encoding. Therefore, we have the following theorem:

**Theorem 26** *Assuming the existence of one-way functions. Then, for any polynomial time computable boolean function $f : \{0,1\}^n \to \{0,1\}$, there exists a robust randomized encoding scheme $(\widehat{f}_{\text{OFF}}, \widehat{f}_{\text{ON}}, \mathcal{S})$ such that the offline complexity is $O(s \cdot \mathsf{poly}(\kappa))$ and online complexity is $O(n \cdot \mathsf{poly}(\kappa))$ where $s$ is the size of the circuit computing $f$, $n$ is the size of the input to $f$ and $\kappa$ is the security parameter.*

## 3 Warmup: Static Zero-Knowledge Proofs from 2PC

Our technique also imply static ZK proofs from any two-party protocol that provides perfect correctness. Intuitively speaking, consider a two-party protocol that is secure in the presence of static adversaries with perfect correctness. Then, the prover generates the transcript of an execution where the parties' inputs are secret shares of the witness $\omega$. That is, the parties' inputs are $\omega_1$ and $\omega_2$, respectively, such that $\omega = \omega_1 \oplus \omega_2$. Upon receiving a challenge bit from the verifier, the prover sends either the input and randomness of $P_1$ or $P_2$, for which the verifier checks for consistency with respect to the transcript, and that $P_2$ outputs 1. From the correctness of the underlying two-party protocol it holds that a malicious prover will not be able to answer both challenges, as that requires generating a complete accepting view. On the other hand, zero-knowledge is implied by the privacy of the two-party protocol. We now proceed with the formal description of our zero-knowledge proof. Let $x$ denote a statement in an $\mathsf{NP}$ language $\mathcal{L}$, associated with relation $\mathcal{R}$, let C be a circuit that outputs 1 on input $(x, \omega)$ only if $(x, \omega) \in \mathcal{R}$, and let $\Pi_g^{\mathrm{OT}} = \langle \pi_1, \pi_2 \rangle$ denote a two-party protocol that privately realizes $C$ with perfect correctness; see Section 5 for the complete details of protocol $\Pi_g^{\mathrm{OT}}$ when embedded with our OT encoding. Our protocol is specified in Figure 1. We note that our protocol implies the *first static zero-knowledge proof* based on (the two-party variant of) [GMW87] and [Yao86]. In Section 5 we discuss how to rely solely on one-way functions. In [HV16] we prove the following claim,

**Static Zero-Knowledge Proof for any Language $\mathcal{L} \in \mathsf{NP}$**

*Inputs:* A circuit C that computes the function $f(x, \omega) = \mathcal{R}(x, \omega)$ and a public statement $x \in \mathcal{L}$ for both. A witness $\omega$ for the validity of $x$ for the prover $\mathcal{P}$.

*The protocol:*

1. $\boldsymbol{\mathcal{P}} \rightarrow \boldsymbol{\mathcal{V}}$: $\mathcal{P}$ invokes $\Pi_g^{\mathrm{OT}}$ and emulates the roles of $P_1$ and $P_2$ on random shares $\omega_1, \omega_2$ of $\omega$, and randomness $r_1, r_2$. Let $\tau$ be the transcript of messages exchanged between these parties. $\mathcal{P}$ sends $\tau$ to the verifier.
2. $\boldsymbol{\mathcal{V}} \rightarrow \boldsymbol{\mathcal{P}}$: The verifier sends a random challenge bit $b \leftarrow \{0, 1\}$.
3. $\boldsymbol{\mathcal{P}} \rightarrow \boldsymbol{\mathcal{V}}$: Upon receiving the bit $b$ the prover continues as follows,
   - If $b = 0$ then the prover sends $(r_1, \omega_1)$.
   - Else, if $b = 1$ then the prover sends $(r_2, \omega_2)$.
4. The verifier checks that the randomness and input are consistent with $\tau$ by emulating the corresponding party. In case of emulating $P_2$, the verifier checks that it further outputs 1.

**Fig. 1.** Static zero-knowledge proof for any language $\mathcal{L} \in \mathsf{NP}$

**Theorem 31** *Assume the existence of one-way functions. Then, the protocol presented in Figure 1 is a static honest verifier zero-knowledge proof for any language in $\mathsf{NP}$.*

## 4 Instance-Dependent Commitments from Garbled Schemes

As a warmup, we present our first adaptive instance-dependent commitment scheme based on our garbled circuits notion as formally defined in Section 2.3 which, in turn, implies a construction for the binary message space $\{0, 1\}$ based on one-way functions (see more detailed discussion in Section 2.3). Let $x$ denote a statement in an $\mathsf{NP}$ language $\mathcal{L}$, associated with relation $\mathcal{R}$, and let C be a circuit that outputs 1 on input $(x, \omega)$ only if $(x, \omega) \in \mathcal{R}$.[10] Intuitively speaking, our construction is described as follows.

[10] More explicitly, we assume that the common statement $x$ is embedded inside the circuit and only $\omega$ is given as its input.

A commitment to the bit $0$ is defined by a garbling of circuit $\mathrm{C}$ , i.e., $\mathsf{Grb}(\mathrm{C})$, and a commitment to the secret key whereas a commitment to the bit $1$ is defined by a simulated garbling of the circuit $\mathrm{C}$ with output set to $1$, i.e., the garbled circuit output by $\mathsf{SimGC}(\mathrm{C}, 1)$, and a commitment the input encoding $\tilde{z}$ that is output by $\mathsf{SimGC}(\mathrm{C}, 1)$. The decommitment to the bit $0$ requires revealing the secret key (all input labels) with which the receiver checks that $\mathsf{Grb}(\mathrm{C})$ is indeed a garbling of $\mathrm{C}$. On the other hand, the decommitment to the bit $1$ requires decommitting to $\tilde{z}$ with which the receiver checks that the simulated garbled circuit evaluates to $1$. Importantly, if the committer knows a witness $\omega$ for the validity of $x$ in $\mathcal{L}$, then it can always honestly commit to a garbling of circuit $\mathrm{C}$ and later decommit to both $0$ and $1$. For statements $x \in \mathcal{L}$, the hiding property of the commitment scheme follows directly from the indistinguishability of the simulated garbled circuit and the hiding property of the underlying commitment scheme. Whereas, for $x \notin \mathcal{L}$, the commitment is perfectly binding as even an unbounded committer cannot provide a honestly generated garbled circuit, and at the same time provide an encoding of some input that evaluates the garbled circuit to $1$ (as there exists no witness $\omega$ for $x$). Finally, considering garbling constructions from the literature, such as the [LP09] scheme, we note that the communication complexity of our construction for committing a single bit equals $O(s \cdot \mathsf{poly}(\kappa))$ where $s$ is the circuit's size and $\kappa$ is the security parameter. In [HV16], a formal proof of the following theorem is provided.

**Theorem 41** *Assume the existence of one-way functions. Then, there exists a secure adaptive instance-dependent commitment scheme for any language in* $\mathsf{NP}$.

## 5 Randomized Encoding from Two-Party Computation

In this section, we show how to construct a randomized encoding for any function $f$, given a two-party computation in the oblivious transfer (OT)-hybrid. This is opposed to prior works that have established the usefulness of randomized encoding in constructing efficient multiparty computation [IK00,AIK04,DI06].

Let $f : \{0,1\}^n \to \{0,1\}$ be an arbitrary polynomial-time computable function. We define $g(a, b) = f(a \oplus b)$ and view $g$ as a two-party functionality. Then let $\Pi_g^{\mathrm{OT}} = \langle \pi_1, \pi_2 \rangle$ be a two-party protocol which realizes $g$ with the following guarantees:

1. It guarantees UC security against semi-honest adversaries in the OT-hybrid that can statically corrupt either $P_1$ or $P_2$ and adaptively corrupt $P_2$. Looking ahead, we consider two different adversaries: (1) adversary $\mathcal{A}_1$ that corrupts $P_1$ at the beginning of the execution and adaptively corrupts $P_2$ post-execution (further denoted as a semi-adaptive adversary [GWZ09]) and (2) adversary $\mathcal{A}_2$ that corrupts $P_2$ at the beginning of the execution. We denote the corresponding simulators by $\mathcal{S}_1$ and $\mathcal{S}_2$.
2. Finally, we require that $P_1$ is the (designated) sender for all OT instances and that the output of the computation is obtained only by $P_2$.

We remark that both the classic Yao's garbled circuit construction [Yao86] and the [GMW87] protocol satisfy these conditions in the OT-hybrid. We further stress that while garbled circuit constructions do not (in general) admit adaptive corruptions, we show that the specific corruption by adversary $\mathcal{A}_1$ can be simulated in the OT-hybrid. In

[HV16] we discuss these two realizations in more details. We next demonstrate how to transform any two-party computation protocol that satisfies the properties listed above to a randomized encoding. Our first construction will rely on trapdoor permutations to realize the OT functionality. We then relax this requirement and show how to rely on one-way functions.

Given any protocol $\Pi_g^{\mathrm{OT}}$ we consider a protocol $\widetilde{\Pi}$ that is obtained from $\Pi_g^{\mathrm{OT}}$ by replacing every OT call with the enhanced trapdoor permutation based OT protocol of [EGL85]. Let $\{f_{\mathrm{TDP}} : \{0,1\}^n \rightarrow \{0,1\}^n\}$ be a family of trapdoor permutations and $h$ be the corresponding hard-core predicate. More precisely,

– For every OT call where $P_1$'s input is $(s_0, s_1)$ and $P_2$'s input is $t$, we require $P_1$ to send the index of a trapdoor permutation $f_{\mathrm{TDP}}$ to $P_2$. Next, $P_2$ samples $v_{1-t}$ and $u_t$ uniformly at random from $\{0,1\}^n$ and sets $v_t = f_{\mathrm{TDP}}(u_t)$. $P_2$ sends $(v_0, v_1)$ to $P_1$, that is followed by the message $(c_0, c_1)$ from $P_1$ to $P_2$ where $c_0 = h(u_0) \oplus s_0$ and $c_1 = h(u_1) \oplus s_1$ and $u_0 = f_{\mathrm{TDP}}^{-1}(v_0), u_1 = f_{\mathrm{TDP}}^{-1}(v_1)$.

We need to verify that $\widetilde{\Pi}$ satisfies all the required properties.

1. It follows from the fact that if $\Pi_g^{\mathrm{OT}}$ implements $g$ with UC security against semi-honest adversaries $\mathcal{A}_1$ and $\mathcal{A}_2$, then $\widetilde{\Pi}$ achieves the same against corresponding adversaries that corrupt the same parties and finally output the view of $P_2$. In more details, recall that $\mathcal{A}_1$ corrupts $P_1$ at the beginning and $P_2$ post execution (adaptively). Now, since $\Pi_g^{\mathrm{OT}}$ admits simulation of $\mathcal{A}_1$ in the OT-hybrid, for the same property to hold for $\widetilde{\Pi}$, it suffices to achieve simulation of the OT protocol where the sender is corrupted at the beginning and the receiver is corrupted post execution. It is easy to see that the [EGL85] protocol satisfies this requirement since the receiver is equivocable. Next, to see that $\mathcal{A}_2$ can be simulated we rely on the fact that the OT protocol described above admits (semi-honest) receiver's simulation. Therefore, $\widetilde{\Pi}$ satisfies all the required properties.
2. This property directly holds as we rely on the same instructions to determine the sender and receiver of the OT calls.

*Our randomized encoding.* We now proceed with the description of our robust randomized encoding of $f$ as formalized in Definition 25 by specifying the functions $\widehat{f}_{\mathrm{OFF}}, \widehat{f}_{\mathrm{ON}}$ and the simulation $\mathcal{S}$. Towards describing our algorithms, we consider a real world experiment carried out between parties $P_1$ and $P_2$ that engage in an execution of $\widetilde{\Pi}$ with environment $\mathcal{Z}$. Let $\mathbf{REAL}_{\widetilde{\Pi},\mathcal{A},\mathcal{Z}}(\kappa, x, \mathbf{r})$ denote the output of $\mathcal{Z}$ on input $x$, random tape $r_{\mathcal{Z}}$ and a security parameter $\kappa$ upon interacting with $\mathcal{A}$ with random tape $r_{\mathcal{A}}$ and parties $P_1, P_2$ with random tapes $r_1, r_2$, respectively, that engage in protocol $\widetilde{\Pi}$ where the inputs are determined by $\mathcal{Z}$ and $\mathbf{r} = (r_{\mathcal{Z}}, r_{\mathcal{A}}, r_1, r_2)$. Let $\mathbf{REAL}_{\widetilde{\Pi},\mathcal{A},\mathcal{Z}}(\kappa, x)$ denote a random variable describing $\mathbf{REAL}_{\widetilde{\Pi},\mathcal{A},\mathcal{Z}}(\kappa, x, \mathbf{r})$ where the random tapes are chosen uniformly. We denote by $\mathbf{IDEAL}_{g,\mathcal{S},\mathcal{Z}}(\kappa, x, \mathbf{r})$ the output of $\mathcal{Z}$ on input $x$, random tape $r_{\mathcal{Z}}$ and security parameter $\kappa$ upon interacting with $\mathcal{S}$ and parties $P_1, P_2$, running an ideal process with random tape $r_{\mathcal{S}}$, where $\mathbf{r} = (r_{\mathcal{Z}}, r_{\mathcal{S}})$. Let $\mathbf{IDEAL}_{g,\mathcal{S},\mathcal{Z}}(\kappa, x)$ denote a random variable describing $\mathbf{IDEAL}_{g,\mathcal{S},\mathcal{Z}}(\kappa, x, \mathbf{r})$ when the random tapes $r_{\mathcal{Z}}$ and $r_{\mathcal{S}}$ are chosen uniformly.

**Encoding:** Consider a (semi-honest) adversary $\mathcal{A}_1$ that corrupts $P_1$ at the beginning of the execution. At the end of the execution, $\mathcal{A}_1$ first sends $\tau$ to $\mathcal{Z}$ where $\tau$ is the transcript of messages exchanged between $P_1$ and $P_2$. Next it (adaptively) corrupts $P_2$ and sends $(a_2, r_2)$ to $\mathcal{Z}$ where $a_2$ and $r_2$ are the respective input and randomness used by party $P_2$. Let $\mathcal{S}_1$ be the corresponding simulator as guaranteed by the properties of $\widetilde{\Pi}$.

1. $\widehat{f}_{\text{OFF}}(r)$: The offline encoding is obtained by running $\mathcal{S}_1$ with randomness $r_{\mathcal{S}_1}$ until it sends the first message to the environment. Recall that $\mathcal{S}_1$ statically corrupts $P_1$, where upon completing the execution, $\mathcal{S}_1$ sends the transcript of the messages to the environment. We define the output of $\widehat{f}_{\text{OFF}}(r)$ to be this output where the input $a_1$ of party $P_1$ is sampled uniformly at random. Notice that the offline part of the encoding does not depend on the input $x$ as required.
2. $\widehat{f}_{\text{ON}}(x, r)$: To obtain the online part, we continue the execution of $\mathcal{S}_1$ in the execution corresponding to the transcript $\tau$ generated by $\widehat{f}_{\text{OFF}}(r)$. Recall that after sending $\tau$, $\mathcal{S}_1$ adaptively corrupts $P_2$ and sends the input and random tape of $P_2$ to the environment. $\widehat{f}_{\text{ON}}(x, r)$ continues the emulation of $\mathcal{S}_1$, where upon corrupting party $P_2$ it feeds $\mathcal{S}_1$ with the input of $P_2$ as $a_2 = x \oplus a_1$ and $f(x)$ as the output. The simulation returns the view of $P_2$ and $\widehat{f}_{\text{ON}}(x, r)$ is set to $(a_2, r_2)$ where $r_2$ is the random tape of $P_2$ output by $\mathcal{S}_1$.

**Decoder:** The decoder $B$ on input $(z_{\text{OFF}}, z_{\text{ON}})$ recomputes the view of $P_2$ from the messages sent by $P_1$ to $P_2$ in $z_{\text{OFF}}$ and the input and randomness of $P_2$ in $z_{\text{ON}}$. It checks if the messages sent from $P_2$ to $P_1$ are consistent with what is in $z_{\text{OFF}}$ and finally outputs what $P_2$ outputs in the execution.

**Simulation:** Consider the (semi-honest) adversary $\mathcal{A}_2$ that statically corrupts $P_2$. At the end of the execution $\mathcal{A}_2$ sends $(\tau, (a_2, r_2))$ to $\mathcal{Z}$ where $\tau$ is the transcript of messages exchanged between $P_1$ and $P_2$ and $a_2$ and $r_2$ are the respective input and randomness used by party $P_2$. Let $\mathcal{S}_2$ be the corresponding simulator. Then the simulation algorithm of the randomized encoding $\mathcal{S}$ is defined as follows. Upon receiving $y = f(x)$, $\mathcal{S}$ invokes $\mathcal{S}_2$ where $P_2$'s input is set to a uniformly chosen random string $a_2$ and its output is set to $y$. Recall that $\mathcal{S}_2$ outputs $(\tau, (a_2, r_2))$ at the end of the execution. Then the output of $\mathcal{S}$ is defined by $(s_{\text{OFF}}, s_{\text{ON}})$ where $s_{\text{OFF}} = \tau$ and $s_{\text{ON}} = (a_2, r_2)$.

**Theorem 51** *Let $(\widehat{f}(x, r), \mathcal{S}, B)$ be as above. Then $\widehat{f}(x, r)$ is a randomized encoding of $f$ with computational privacy. Assuming the existence of enhanced trapdoor permutations, we obtain an encoding with offline complexity $C_\Pi + \rho_\Pi \kappa$ and online complexity $|x| + r_\Pi + \rho_\Pi \kappa$ where $C_\Pi$ is the communication complexity of $\Pi_g^{\text{OT}}$ in the OT-hybrid, $\rho_\Pi$ in the number of OT invocations made by $P_2$, $r_\Pi$ is the randomness complexity of $P_2$ in $\Pi_g^{\text{OT}}$ and $\kappa$ is the security parameter. If we instead rely on one-way functions we achieve an encoding with offline and online complexities $C_\Pi + \rho_\Pi \mathsf{poly}(\kappa)$ and $|x| + r_\Pi + \rho_\Pi \mathsf{poly}(\kappa)$, respectively.*

In [HV16] we discuss the relaxation to one-way functions and the proof.

*Complexity.* Finally, we measure the complexity of our encoding. Note first that for each OT call the offline encoding is a pair of image elements of the one-way permutation incurring $O(\kappa)$ overhead, while the online complexity is a preimage of length $\kappa$. Then the offline encoding of the overall construction is the communication complexity of $\widetilde{\Pi}$ which equals to the communication of $\Pi_g^{\mathrm{OT}}$, denoted by $C_\Pi$, together with the number of OT calls, denoted by $\rho_\Pi$, which overall yields $C_\Pi + \rho_\Pi O(\kappa)$. Moreover, the online encoding includes $P_2$'s input $a_2$ and randomness $r_2$ where the latter includes the randomness complexity of $\Pi_g^{\mathrm{OT}}$ and the complexity of the receiver's randomness for the OT invocations which is $|x| + r_\Pi + \rho_\Pi\kappa$. If we rely on one-way functions then the OT calls are incorporated as commitments and incur $\mathsf{poly}(\kappa)$ per invocation for the commitment as well as the decommitment algorithms.

## 5.1 Corollaries and Applications

Below, we demonstrate the power of the proceeding transformation by proving lower bounds and providing additional applications. We discuss instance-dependent commitment schemes in [HV16] as well as realizations for our RE.

**Input-Delayed Zero-Knowledge Proofs** In this section, we extend the basic construction of instance-dependent commitment schemes from our previous construction to additionally allow constructing input-delayed zero-knowledge proofs. We show how randomized-encoding that is secure against adaptive chosen inputs can be used to realize input-delayed zero-knowledge proofs. Then relying on the recent construction of such a randomized encoding [HJO$^+$15] we obtain a constant-rate input-delayed zero-knowledge proof, namely whose communication complexity is $O(s) + \mathsf{poly}(\kappa)$ where $s$ is the size of the circuit realizing the $\mathsf{NP}$-relation and $\kappa$ is the security parameter. We achieve this in two steps. First, we extend our notion of instance-dependent commitment scheme to one where the actual commitment scheme do not require the input statement. Then using such an instance-dependent commitment scheme we will show how to realize an input-delayed zero-knowledge proofs. We provide next definitions for the above primitives.

Our first notion is that of input-delayed instant-dependent commitment scheme. On a high-level, this primitive is a variant of the plain instant-dependent commitment scheme where the real and fake commitment algorithms do not require the knowledge of the input statement in the commit phase. The statement can be adaptively chosen based on the commit phase and will be required only in the decommit phase. Second, we will not require an $\mathsf{Adapt}$ algorithm that can explain a fake commitment as an honest commitment of any message by generating random coins for an honest committer that would have produced the same commitment. Instead, we will only require the slightly weaker property of the fake commitment being equivocable. Towards this, we will introduce a decommitment algorithm for the honest commitment that additionally takes as input the statement $x$ and produces a decommitment to the corresponding message $m$. The receiver then verifies the decommitment with respect to the statement $x$. Corresponding to the fake commitment algorithm, we now require an algorithm that, given the statement and the witness can reveal a commitment (i.e. produce decommitments) to any message $m$.

**Definition 52 (Input-delayed IDCS)** *Let $\mathcal{R}$ be an $\mathsf{NP}$ relation and $\mathcal{L}$ be the language associated with $\mathcal{R}$. A (non-interactive) instance dependent commitment scheme (IDCS) for $\mathcal{L}$ is a tuple of probabilistic polynomial-time algorithms $(\widetilde{\mathsf{Com}}, \widetilde{\mathsf{Decom}}, \widetilde{\mathsf{Ver}}, \widetilde{\mathsf{Com}}', \mathsf{Equiv})$, where:*

- $\widetilde{\mathsf{Com}}$ *is the commitment algorithm: For a message $m \in \mathcal{M}_n$, and a random string $r \in \{0,1\}^{p(n)}$, $\widetilde{\mathsf{Com}}(1^n, m; r)$ returns a commitment value $c$ where $n$ is the length of the input-instance and $p(\cdot)$ is a polynomial.*
- $\widetilde{\mathsf{Decom}}$ *is the decommitment algorithm that on input a statement $x$, commitment $c$, mesage $m$ and randomness $r$ outputs a decommitment $d$.*
- $\widetilde{\mathsf{Ver}}$ *is the verification algorithm that on input $x, m, c, d$ outputs $\mathsf{accept}$ or $\mathsf{reject}$.*
- $\widetilde{\mathsf{Com}}'$ *is a "fake" commitment algorithm: For a random string $r \in \{0,1\}^{q(n)}$, $\widetilde{\mathsf{Com}}'(1^n, r)$ returns a commitment value $c$ where $n$ is the length of the input instance and $q(\cdot)$ is a polynomial.*
- $\mathsf{Equiv}$ *is an equivocation algorithm: Let $x \in \mathcal{L}$ and $\omega \in \mathcal{R}_x$. For all $c$ and $r \in \{0,1\}^{q(|x|)}$ such that $\mathsf{Com}'(r) = c$, and for all $m \in \mathcal{M}_n$, $\mathsf{Equiv}(x, \omega, c, m, r)$ outputs $d$ such that $\widetilde{\mathsf{Ver}}(x, m, c, d)$ outputs $\mathsf{accept}$.*

The hiding property now requires that for any message $m$, an honest commitment and decommitment to $m$ be indistinguishable from a fake commitment and decommitment to $m$ even when the input statement is adaptively chosen after the commitment phase. The binding property on the other hand will require that for any commitment $c$ and a false statement $x \notin \mathcal{L}$, there exists no values $m, d$ and $m', d'$ such that $\widetilde{\mathsf{Ver}}(x, m, c, d) = \widetilde{\mathsf{Ver}}(x, m', c, d') = \mathsf{accept}$. Finally, in Figure 2 we describe our input-delayed zero-knowledge proof.

**Theorem 53** *Assume the existence of one-way functions. Then, the protocol presented in Figure 2 is an input-delayed zero-knowledge proof with soundness $1/2$ for any language in $\mathsf{NP}$.*

See [HV16] for the proof. Finally, we need to show how our input-delayed IDCS can be constructed from a robust randomized encoding that is secure against an adaptive chosen input. We begin with a randomized encoding for the following function $f$: $f(x, \omega) = (\mathcal{R}(x, \omega), x)$. Since the randomized encoding is secure against adaptive choice of inputs, the simulation algorithm of the RE is decomposed into two algorithms, namely the offline part $s_{\text{OFF}}$ and online part $s_{\text{ON}}$. Now, we can define our commitment algorithm as follows: A commitment to 0 returns the offline part of the encoding $\widehat{f}_{\text{OFF}}(r)$ whereas a commitment to 1 returns the offline part of the simulation $s_{\text{OFF}}(r')$ where $r$ and $r'$ are the randomness used for the algorithms. A decommitment to 0 requires revealing randomness showing that the commitment was generated honestly using $\widehat{f}_{\text{OFF}}(r)$ and a decommitment to 1 requires providing the online part $s_{\text{ON}}$ that along with the commitment decodes to $(1, x)$ where $x$ is the statement. Finally, the fake commitment algorithm is defined as a commitment to 0. Observe that both the honest and fake commitment algorithms do not depend on the input statement. This is enabled by the adaptive input security of the randomized encoding. The hiding property of the commitment for bit 0 holds directly, whereas the hiding property for the bit

**Input-Delayed Zero-Knowledge Proof for any Language $\mathcal{L} \in \mathsf{NP}$**

*Building block:* Input delayed IDCS $(\widetilde{\mathsf{Com}}, \widetilde{\mathsf{Decom}}, \widetilde{\mathsf{Ver}}, \widetilde{\mathsf{Com}}', \mathsf{Equiv})$ for $\mathcal{L}$.

*Inputs:* A circuit C that computes the function $f(x, \omega) = (\mathcal{R}(x, \omega), x)$.

*The protocol:*

1. $\mathcal{P} \to \mathcal{V}$: $\mathcal{P}$ invokes $\mathsf{com}_0 \leftarrow \widetilde{\mathsf{Com}}'(1^\kappa; r)$ and $\mathsf{com}_1 \leftarrow \widetilde{\mathsf{Com}}'(1^\kappa; r)$ and sends $(\mathsf{com}_0, \mathsf{com}_1)$ to the verifier.
2. $\mathcal{V} \to \mathcal{P}$: The verifier sends a random challenge $b \leftarrow \{01, 10\}$.
3. $\mathcal{P} \to \mathcal{V}$: Upon receiving the input statement $x$ and witness $\omega$,
   - If $b = 01$ then the prover sends the decommitments to $(\mathsf{com}_0, \mathsf{com}_1)$ by computing $\mathsf{Equiv}(x, \omega, \mathsf{com}_0, 0, r)$ and $\mathsf{Equiv}(x, \omega, \mathsf{com}_1, 1, r)$.
   - If $b = 10$ then the prover sends the decommitments to $(\mathsf{com}_0, \mathsf{com}_1)$ by computing $\mathsf{Equiv}(x, \omega, \mathsf{com}_0, 1, r)$ and $\mathsf{Equiv}(x, \omega, \mathsf{com}_1, 0, r)$.
4. The verifier checks that the decommitments are valid with respect to $x$.

**Fig. 2.** Input delayed zero-knowledge proof for any language $\mathcal{L} \in \mathsf{NP}$

1 follows from the simulation property of the randomized encoding. Binding on the other hand follows directly from the robustness property of the randomized encoding. The complete description is given in [HV16]. We note that the work of Hemenway et al. [HJO$^+$15] shows how to obtain a randomized encoding that is secure against adaptively chosen inputs. We show in [HV16] how to extend it to achieve the stronger robustness property. Combining their work with our construction, we have the following corollary.

**Corollary 54** *Assuming the existence of one-way functions. Then for any $\mathsf{NP}$-relation $\mathcal{R}$, there exists an input-delayed ZK proof with communication complexity $O(s \cdot \mathsf{poly}(k))$ where $s$ is the size of the circuit computing the $\mathsf{NP}$ relation.*

## 5.2 Commit-and-Prove Zero-Knowledge Proofs

In the "commit-and-prove" paradigm, the prover first commits to its witness and then proves that the statement, along with the decommitment value maintains the underlying $\mathsf{NP}$ relation. This paradigm has turned useful for constructing maliciously secure protocols [GMW87,CLOS02]. In this section we show how to design such an *input-delayed* proof, namely, where the statement is determined only at the last round and the underlying commitment scheme (in turn the one-way function) is used in a black-box way. Specifically, in this input-delaying flavour the witness is known ahead of time but not the statement, and hence not the $\mathsf{NP}$ relation.

As above, we employ a robust randomized encoding that is secure in the presence of adaptive choice of inputs, where the simulation algorithm is split into an offline and online phases, that computes the function $f_{\omega_0}(x, \omega_1) = (\mathcal{R}(x, \omega_0 \oplus \omega_1), x, \omega_1)$ where $\omega_0$ is hardwired into the circuit that computes this functionality. The reason we need to hardwire it is because the offline phase must be associated with this share. Whereas the other share $\omega_1$ is output by the circuit in order to enforce the usage of the right share.

*Achieving negligible soundness.* In order to improve the soundness parameter of our ZK proof we need to repeat the basic proof sufficiently many times in parallel, using fresh witness shares each time. This, however, does not immediately work as the dishonest prover may use different shares for each proof instance. In order to overcome this problem we use the [IKOS09] approach in order to add a mechanism that verifies the consistency of the shares. Namely, suppose we wish to repeat the basic construction in parallel $N = O(t)$ times where $t = O(\kappa)$ and $\kappa$ is the security parameter. Formally,

– The verifier picks a random $t$-subset $I$ of $[N]$. It also picks $t$ random challenge bit $\{ch_i\}_{i\in I}$ and commits to them.
– The prover then continues as follows:
  1. It first generates $N$ independent XOR sharings of $w$, say $\{(w_{i,0}, w_{i,1})\}_{i\in[N]}$.
  2. It generates the views of $2N$ parties $P_{i,0}$ and $P_{i,1}$ for $i \in [N]$ executing a $t$-robust $t$-private MPC protocol, where $P_{i,j}$ has input $w_{i,j}$, that realizes the functionality that checks if $w_{i,0} \oplus w_{i,1}$ are equal for all $i$. Let $V_{i,j}$ be view of party $P_{i,j}$.
  3. Next, it computes $N$ offline encodings of the following set of functions:

$$f_{w_{i,0},V_{i,0}}(x, w_{i,1}, V_{i,1}) = (b, x, w_{i,1}, V_{i,1})$$

     for $i \in [N]$, where $b = 1$ if and only if $\mathcal{R}(x, w_{i,0} \oplus w_{i,1})$ holds and the views $V_{i,0}$ and $V_{i,1}$ are consistent with each other.
  4. Finally, the prover sends:

$$\left\{f^{\rm OFF}_{w_{i,0}}(r_i), \mathsf{com}(r_i), \mathsf{com}(w_{i,0}), \mathsf{com}(w_{i,1}), \mathsf{com}(V_{i,0}), \mathsf{com}(V_{i,1})\right\}_{i\in[N]}.$$

– The verifier decommits to all its challenges.
– For every index $i$ in the $t$ subset the prover replies as follows:
  • If $ch_i = 0$ then it decommits to $r_i$, $w_{i,0}$ and $V_{i,0}$. The verifier then checks if the offline part was constructed correctly (as in our basic proof).
  • If $ch_i = 1$ then i sends $f^{\rm ON}_{w_{i,0}}(r_i, x, w_{i,1})$ and decommits $w_{i,1}$. The verifier then runs the decoder and checks if it obtains $(1, x, w_{i,0})$.

  Furthermore, for every index $i$, the prover decommits the views $V_{i,ch_i}$ for which the verifier checks if the MPC-in-the-head protocol was executed correctly.

**Theorem 55** *Assume the existence of one-way functions. Then, the above protocol is a commit-and-prove input-delayed zero-knowledge proof with negligible soundness for any language in* $\mathsf{NP}$.

## 6 Constructing Adaptive Zero-Knowledge Proofs

We describe next how to construct adaptive zero-knowledge proofs for all $\mathsf{NP}$ languages based on our instance-dependent commitment schemes from Sections 4 and 5.

Let $x$ denote a statement to be proven by the prover relative to some language $\mathcal{L}$ associated with relation $\mathcal{R}$. Then the prover generates a garbled circuit C that takes $(x, \omega)$ and outputs 1 only if $(x, \omega) \in \mathcal{R}$, and commits to this garbling and the secret key $\mathsf{sk}$ using the commitment scheme from Section 4. Next, upon receiving a challenge bit $b$ from the verifier, the prover continues as follow. If $b = 0$ then the prover decommits to the commitment of the secret key and the garbled circuit for which the verifier verifies the correctness of garbling. Else, if $b = 1$ then the prover decommits a "path" in the garbled circuit and provides an encoding for $\omega$ that evaluates the path to 1. Namely, we consider the concrete garbling construction by [Yao86,LP09] for which each evaluation induces a path of computation, where each gate evaluation requires the decryption of a single ciphertext out of four ciphertexts, where this ciphertext can be part of the decommitted information handed to the verifier when $b = 1$. The verifier then evaluates the garbling on this path and checks that the outcome if 1. We note that it is not clear how to generalize this property (where only part of the garbled circuit is decommitted) nor the following reconstructability property for the notion of randomized encodings.

Let $\mathsf{Garb} = (\mathsf{Grb}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ denote a garbling scheme as in Section 2.3. Then, we will require one more property that $\mathsf{Garb}$ should satisfy:

**Reconstructability:** Given any path of computation in the garbled circuit it is possible to reconstruct the rest of the garbled circuit as being honestly generated by $\mathsf{Grb}$.

We note that the [LP09] garbling scheme meets this notion. The description of our protocol can be found in Figure 3 and the proof of the following theorem in [HV16].

**Theorem 61** *Assume the existence of one-way functions. Then, the protocol presented in Figure 3 is an adaptively secure honest verifier zero-knowledge proof for any language in $\mathsf{NP}$ with soundness error* $1/2$.

We note that the communication complexity of our protocol is $O(\kappa s^2)$ where $\kappa$ is the security parameter and $s$ is the size of C. In the full version we extend this construction to achieve a linear-rate adaptive ZK proof and obtain the following theorem.

**Theorem 62** *Assume the existence of one-way functions. Then, for any $\mathsf{NP}$ relation $\mathcal{R}$ that can be verified by a circuit of size $s$ (using bounded fan-in gates), there exists an adaptive zero-knowledge proof with communication complexity $O(s) \cdot \mathsf{poly}(\kappa, \log s)$ where $\kappa$ is the security parameter.*

## References

[AIK04] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in $NC^0$. In *FOCS*, pages 166–175, 2004.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nc$^0$. *SIAM J. Comput.*, 36(4):845–888, 2006.

[AIK10] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. From secrecy to soundness: Efficient verification via secure computation. In *ICALP*, pages 152–163, 2010.

[AIKP15] Shweta Agrawal, Yuval Ishai, Dakshita Khurana, and Anat Paskin-Cherniavsky. Statistical randomized encodings: A complexity theoretic view. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, pages 1–13, 2015.

**Adaptive Zero-Knowledge Proof for Any Language $\mathcal{L} \in \mathsf{NP}$**

*Building block:* Instance-dependent commitment scheme $\mathsf{Com}$ for language $\mathcal{L}$.

*Inputs:* A circuit C as above and a public statement $x \in \mathcal{L}$ for both. A witness $\omega$ for the validity of $x$ for the prover $\mathcal{P}$.

*The protocol:*

1. $\boldsymbol{\mathcal{P}} \rightarrow \boldsymbol{\mathcal{V}}$: $\mathcal{P}$ generates $(\widetilde{\mathrm{C}}, \mathbf{dk}, \mathsf{sk}) \leftarrow \mathsf{Grb}(1^\kappa, \mathrm{C})$ and sends $\mathsf{Com}(\widetilde{\mathrm{C}}, \mathbf{dk})$ and $\mathsf{Com}(\mathsf{sk})$ to the verifier (where the commitments are computed using the real commitment algorithm).
2. $\boldsymbol{\mathcal{V}} \rightarrow \boldsymbol{\mathcal{P}}$: The verifier sends a random challenge bit $b \leftarrow \{0,1\}$.
3. $\boldsymbol{\mathcal{P}} \rightarrow \boldsymbol{\mathcal{V}}$:
   - If $b = 0$ then the prover decommits to $\widetilde{\mathrm{C}}, \mathbf{dk}$ and $\mathsf{sk}$. The verifier accepts if the decommitments are valid and that the garbling was honestly generated.
   - If $b = 1$ then the prover decommits to $\mathbf{dk}$ and further provides the decommitment for the encoding of $\omega$ and the path of computation in the commitment to $\widetilde{\mathrm{C}}$ that is evaluated during the computation of $\mathsf{Eval}(\widetilde{\mathrm{C}}, \widetilde{\omega})$. Namely, the prover invokes $\widetilde{\omega} := \mathsf{Enc}(\mathsf{sk}, \omega)$ and then decommits to the encoding of $\widetilde{\omega}$ within the commitment of $\mathsf{sk}$ (recall that this is possible due to the decomposability of the garbled scheme), as well as the path of computation. The verifier then invokes $\widetilde{y} := \mathsf{Eval}(\widetilde{\mathrm{C}}, \widetilde{\omega})$ and accepts if $\mathsf{Dec}(\mathbf{dk}, \widetilde{\omega})$ equals 1.

**Fig. 3.** Adaptive zero-knowledge proof for any language $\mathcal{L} \in \mathsf{NP}$

[AIKW13] Benny Applebaum, Yuval Ishai, Eyal Kushilevitz, and Brent Waters. Encoding functions with constant online rate or how to compress garbled circuits keys. In *CRYPTO*, pages 166–184, 2013.

[App14] Benny Applebaum. Key-dependent message security: Generic amplification and completeness. *J. Cryptology*, 27(3):429–451, 2014.

[Bea96] Donald Beaver. Correlated pseudorandomness and the complexity of private computations. In *STOC*, pages 479–488, 1996.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.

[BHHI10] Boaz Barak, Iftach Haitner, Dennis Hofheinz, and Yuval Ishai. Bounded key-dependent message security. In *EUROCRYPT*, pages 423–444, 2010.

[BHR12] Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In *CCS*, pages 784–796, 2012.

[BMO90] Mihir Bellare, Silvio Micali, and Rafail Ostrovsky. Stoc. pages 482–493, 1990.

[CCD87] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (abstract). In *CRYPTO*, page 462, 1987.

[CDD$^+$04] Ran Canetti, Ivan Damgård, Stefan Dziembowski, Yuval Ishai, and Tal Malkin. Adaptive versus non-adaptive security of multi-party protocols. *J. Cryptology*, 17(3):153–207, 2004.

[CDD$^+$15] Ignacio Cascudo, Ivan Damgård, Bernardo Machado David, Irene Giacomelli, Jesper Buus Nielsen, and Roberto Trifiletti. Additively homomorphic UC commitments with optimal amortized overhead. In *PKC*, pages 495–515, 2015.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *STOC*, pages 494–503, 2002.

[CPS$^+$15] Michele Ciampi, Giuseppe Persiano, Alessandra Scafuro, Luisa Siniscalchi, and Ivan Visconti. Improved OR composition of sigma-protocols. *IACR Cryptology ePrint Archive*, 2015:810, 2015.

[CvdGT95] Claude Crépeau, Jeroen van de Graaf, and Alain Tapp. Committed oblivious transfer and private multi-party computation. In *CRYPTO*, pages 110–123, 1995.

[Dam10] Ivan Damgård. On $\Sigma$-protocols. *http://www.cs.au.dk/ ivan/Sigma.pdf*, 2010.

[DHRS04] Yan Zong Ding, Danny Harnik, Alon Rosen, and Ronen Shaltiel. Constant-round oblivious transfer in the bounded storage model. In *TCC*, pages 446–472, 2004.

[DI06] Ivan Damgård and Yuval Ishai. Scalable secure multiparty computation. In *CRYPTO*, pages 501–520, 2006.

[EGL85] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.

[FKN94] Uriel Feige, Joe Kilian, and Moni Naor. A minimal model for secure computation (extended abstract). In *STOC*, pages 554–563, 1994.

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM J. Comput.*, 29(1):1–28, 1999.

[FS89] Uriel Feige and Adi Shamir. Zero knowledge proofs of knowledge in two rounds. In *CRYPTO*, pages 526–544, 1989.

[GGP10] Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *CRYPTO*, pages 465–482, 2010.

[GIS$^+$10] Vipul Goyal, Yuval Ishai, Amit Sahai, Ramarathnam Venkatesan, and Akshay Wadia. Founding cryptography on tamper-proof hardware tokens. In *TCC*, pages 308–326, 2010.

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. One-time programs. In *CRYPTO*, pages 39–56, 2008.

[GLOV12] Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *FOCS*, pages 51–60, 2012.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.

[GOSV14] Vipul Goyal, Rafail Ostrovsky, Alessandra Scafuro, and Ivan Visconti. Black-box non-black-box zero knowledge. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 515–524, 2014.

[GWZ09] Juan A. Garay, Daniel Wichs, and Hong-Sheng Zhou. Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. In *CRYPTO*, pages 505–523, 2009.

[HIKN08] Danny Harnik, Yuval Ishai, Eyal Kushilevitz, and Jesper Buus Nielsen. Ot-combiners via secure computation. In *TCC*, pages 393–411, 2008.

[HJO$^+$15] Brett Hemenway, Zahra Jafargholi, Rafail Ostrovsky, Alessandra Scafuro, and Daniel Wichs. Adaptively secure garbled circuits from one-way functions. *IACR Cryptology ePrint Archive*, 2015:1250, 2015.

[HR07] Iftach Haitner and Omer Reingold. A new interactive hashing theorem. In *CCC*, pages 319–332, 2007.

[HV16] Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On the power of secure two-party computation. *IACR Cryptology ePrint Archive*, 2016:74, 2016.

[IK00] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *FOCS*, pages 294–304, 2000.

[IK02] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *ICALP*, pages 244–256, 2002.

[IKOS07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 21–30, 2007.

[IKOS09] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM J. Comput.*, 39(3):1121–1152, 2009.

[IKPY16] Yuval Ishai, Eyal Kushilevitz, Manoj Prabhakaran, and Amit Sahai Ching-Hua Yu. Secure protocol transformations. To appear CRYPTO, 2016.

[IOS97] Toshiya Itoh, Yuji Ohta, and Hiroki Shizuya. A language-dependent cryptographic primitive. *J. Cryptology*, 10(1):37–50, 1997.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In *CRYPTO*, pages 572–591, 2008.

[IPS09] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Secure arithmetic computation with no honest majority. In *TCC*, pages 294–314, 2009.

[IW14] Yuval Ishai and Mor Weiss. Probabilistically checkable proofs of proximity with zero-knowledge. In *TCC*, pages 121–145, 2014.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *STOC*, pages 20–31, 1988.

[KO04] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In *CRYPTO*, pages 335–354, 2004.

[LP09] Yehuda Lindell and Benny Pinkas. A proof of security of Yao's protocol for two-party computation. *J. Cryptology*, 22(2):161–188, 2009.

[LS90] Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. In *CRYPTO*, pages 353–365, 1990.

[LZ11] Yehuda Lindell and Hila Zarosim. Adaptive zero-knowledge proofs and adaptively secure oblivious transfer. *J. Cryptology*, 24(4):761–799, 2011.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.

[NOVY98] Moni Naor, Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Perfect zero-knowledge arguments for *NP* using any one-way permutation. *J. Cryptology*, 11(2):87–108, 1998.

[OSV15] Rafail Ostrovsky, Alessandra Scafuro, and Muthuramakrishnan Venkitasubramaniam. Resettably sound zero-knowledge arguments from owfs - the (semi) black-box way. In *TCC*, pages 345–374, 2015.

[OV08] Shien Jin Ong and Salil P. Vadhan. An equivalence between zero knowledge and commitments. In *TCC*, pages 482–500, 2008.

[PSSW09] Benny Pinkas, Thomas Schneider, Nigel P. Smart, and Stephen C. Williams. Secure two-party computation is practical. In *ASIACRYPT*, pages 250–267, 2009.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, pages 162–167, 1986.