# Concurrent Zero Knowledge without Complexity Assumptions*

Daniele Micciancio[1]**, Shien Jin Ong[2]***, Amit Sahai[3]†, and Salil Vadhan[2]‡

[1] University of California, San Diego, La Jolla CA 92093, USA
daniele@cs.ucsd.edu
[2] Harvard University, Cambridge MA 02138, USA
{shienjin, salil}@eecs.harvard.edu
[3] University of California, Los Angeles, Los Angeles CA 90095, USA
sahai@cs.ucla.edu

**Abstract.** We provide *unconditional* constructions of *concurrent* statistical zero-knowledge proofs for a variety of non-trivial problems (not known to have probabilistic polynomial-time algorithms). The problems include Graph Isomorphism, Graph Nonisomorphism, Quadratic Residuosity, Quadratic Nonresiduosity, a restricted version of Statistical Difference, and approximate versions of the (**coNP** forms of the) Shortest Vector Problem and Closest Vector Problem in lattices.

For some of the problems, such as Graph Isomorphism and Quadratic Residuosity, the proof systems have provers that can be implemented in polynomial time (given an **NP** witness) and have $\tilde{O}(\log n)$ rounds, which is known to be essentially optimal for black-box simulation.

To the best of our knowledge, these are the first constructions of concurrent zero-knowledge proofs in the plain, asynchronous model (i.e., without setup or timing assumptions) that do not require complexity assumptions (such as the existence of one-way functions).

## 1 Introduction

In the two decades since their introduction [2], zero-knowledge proofs have taken on a central role in the study of cryptographic protocols, both as a basic building block for more complex protocols and as a testbed for understanding important new issues such as composability (e.g., [3]) and concurrency (e.g., [4]). The "classic" constructions of zero-knowledge proofs came primarily in two flavors. First, there were direct constructions of zero-knowledge proofs for specific problems, such as Quadratic Residuosity [2] and Graph Isomorphism [5]. Second, there were general constructions of zero-knowledge proofs for entire classes of

* A full version of this paper is available [1].

** Supported by NSF grant 0313241 and an Alfred P. Sloan Research Fellowship.

*** Supported by ONR grant N00014-04-1-0478.

† Supported by NSF ITR and Cybertrust programs, an equipment grant from Intel, and an Alfred P. Sloan Foundation Fellowship.

‡ Supported by NSF grants CNS-0430336 and CCR-0205423.

problems, such as all of **NP** [5].[4] Both types of results have played an important role in the development of the field.

The general results of the second type show the wide applicability of zero knowledge, and are often crucial in establishing general feasibility results for other cryptographic problems, such as secure multiparty computation [8, 5] and CCA-secure public-key encryption [9–11]. However, they typically are too inefficient to be used in practice. The specific results of the first type are often much more efficient, and are therefore used in (or inspire) the construction of other efficient cryptographic protocols, e.g., identification schemes [12] and again CCA-secure public-key encryption [13–15]. Moreover, the specific constructions typically do not require any unproven complexity assumptions (such as the existence of one-way functions), and yield a higher security guarantee (such as *statistical* zero-knowledge proofs).[5] The fact that the proof systems are unconditional is also of conceptual interest, because they illustrate the nontriviality of the notion of zero knowledge even to those who are unfamiliar with (or who do not believe in the existence of) one-way functions.[6]

*Concurrent zero knowledge.* In recent years, a substantial effort has been devoted to understanding the security of cryptographic protocols when many executions are occurring concurrently (with adversarial scheduling). As usual, zero-knowledge proofs led the way in this effort, with early investigations of concurrency for relaxations of zero knowledge dating back to Feige's thesis [22], and the recent interest being sparked by the work of Dwork, Naor, and Sahai [4], which first defined the notion of concurrent zero knowledge. Research on concurrent zero knowledge has been very fruitful, with a sequence of works leading to essentially tight upper and lower bounds on round complexity for black-box simulation [23–28], and partly motivating the first non-black-box-simulation zero-knowledge proof [29]. However, these works are primarily of the *second* flavor mentioned in the first paragraph. That is, they are general feasibility results, giving protocols for all of **NP**. As a result, these protocols are fairly inefficient (in terms of computation and communication), rely on unproven complexity assumptions, and only yield computational zero knowledge (or, alternatively, computational soundness).

There have been a couple of works attempting to overcome these deficiencies. Di Crescenzo [30] gave unconditional constructions of concurrent zero-knowledge

---

[4] See the textbook [6] and survey [7] by Oded Goldreich for a thorough introduction to zero-knowledge proofs.

[5] Of course, this partition into two types of zero-knowledge protocols is not a precise one. For example, there are some efficient zero-knowledge proofs for specific problems that use complexity assumptions (e.g., [16] and there are some general results that are unconditional (e.g., [17–19]).

[6] It should be noted that the results of [20, 21] show that the existence of a zero-knowledge proof for a problem outside **BPP** implies some weak form of one-way function. Still, appreciating something like the perfect zero-knowledge proof system for Graph Isomorphism [5] only requires believing that there is no *worst-case* polynomial-time algorithm for Graph Isomorphism, as opposed to appreciating notions of average-case complexity as needed for standard one-way functions.

proofs in various timing models. That is, his protocols assume that the honest parties have some synchronization and may employ delays in the protocol, and thus do not work in the standard, asynchronous model (and indeed he states such a strengthening as an open problem). Micciancio and Petrank [31] gave an efficient (in terms of computation and communication) transformation from honest-verifier zero-knowledge proofs to concurrent zero-knowledge proofs. However, their transformation relies on the Decisional Diffie–Hellman assumption, and yields only computational zero knowledge.

*Our Results.* We give the first unconditional constructions of concurrent zero-knowledge proofs in the standard, asynchronous model. Our proof systems are statistical zero knowledge and statistically sound (i.e. they are interactive proofs, not arguments [32]). Specifically, our constructions fall into two categories:

1. Efficient proof systems for certain problems in **NP**, including Quadratic Residuosity, Graph Isomorphism and a restricted form of quadratic nonresiduosity for Blum integers, which we call Blum Quadratic Nonresiduosity. These proof systems all have prover strategies that can be implemented in polynomial time given an **NP** witness and have $\widetilde{O}(\log n)$ rounds, which is essentially optimal for black-box simulation [27].
2. Inefficient proof systems for other problems, some of which are not known to be in **NP**. These include Quadratic Nonresiduosity, Graph Nonisomorphism, the approximate versions of the complements of the Closest Vector Problem and Shortest Vector Problem in lattices, and a restricted version of Statistical Difference (the unrestricted version is complete for statistical zero knowledge [33]). These proof systems have a polynomial number of rounds, and do not have polynomial-time prover strategies. These deficiencies arise from the fact that our construction begins with a public-coin, honest-verifier zero-knowledge proof for the problem at hand, and the only such proofs known for the problems listed here have a polynomial number of rounds and an inefficient prover strategy.

*Techniques.* One of the main tools for constructing zero-knowledge proofs are commitment schemes, and indeed the only use of complexity assumptions in the construction of zero-knowledge proofs for all of **NP** [5] is to obtain a commitment scheme (used by the prover to commit to the **NP** witness, encoded as, e.g., a 3-coloring of a graph). Our results rely on a relaxed notion of commitment, called an *instance-dependent commitment scheme*,[7] which is implicit in [35] and formally defined in [36, 34, 19]. Roughly speaking, for a language $L$ (or, more generally, a promise problem), a instance-dependent commitment scheme for $L$ is a commitment protocol where the sender and receiver algorithms also depend on the instance $x$. The security requirements of the protocol are relaxed so that the hiding property is only required when $x \in L$, and the binding property is only required when $x \notin L$ (or vice-versa).

[7] Previous works [34, 19] have referred to this as "problem-dependent" commitment scheme, but this new terminology of "instance-dependent" seems more accurate.

As observed in [36], many natural problems, such as Graph Isomorphism and Quadratic Residuosity, have simple, unconditional instance-dependent commitment schemes. This is useful because in many constructions of zero-knowledge proofs (such as that of [5]), the hiding property of the commitment scheme is only used to establish the zero-knowledge property and the binding property of the commitment scheme is only used to establish soundness. Since, by definition, the zero-knowledge property is only required when the input $x$ is in the language, and the soundness condition is only required when $x$ is not in the language, it suffices to use a instance-dependent commitment scheme. Specifically, if a language $L \in \mathbf{NP}$ (or even $L \in \mathbf{IP}$) has a instance-dependent commitment scheme, then $L$ has a zero-knowledge proof [36] (see also [34, 19]).

Existing constructions of *concurrent* zero-knowledge proofs [24, 27, 28] also rely on commitment schemes (and this is the only complexity assumption used). Thus it is natural to try to use instance-dependent commitments to construct them. However, these protocols use commitments not only from the prover to the verifier, but also from the verifier to the prover. Naturally, for the latter type of commitments, the roles of the hiding and binding property are reversed from the above — the hiding property is used to prove soundness and the binding property is used to prove (concurrent) zero knowledge. Thus, it seems that we need not only a instance-dependent commitment as above, but also one where the security properties are reversed (i.e. binding when $x \in L$, and hiding when $x \notin L$).

Our first observation is that actually we only need to implement the commitment schemes from the verifier to the prover. This is because the concurrent zero-knowledge proof system of Prabhakaran, Rosen and Sahai [28] is constructed by a general compiler that converts *any* public-coin zero-knowledge proof into a concurrent zero-knowledge proof, and this compiler only uses commitments from the verifier to the prover. (Intuitively, the verifier commits to its messages in an initial "preamble" stage, which is designed so as to allow concurrent simulation.) Since all the problems we study are unconditionally known to have public-coin zero-knowledge proofs, we only need to implement the compiler. So we are left with the task finding instance-dependent commitments that are binding when $x \in L$ and hiding when $x \notin L$. Thus, for the rest of the paper, we use this as our definition of instance-dependent commitment.

This idea works directly for some problems, such as Graph Nonisomorphism and Quadratic Nonresiduosity. For these problems, we have instance-dependent commitments with the desired security properties, and thus we can directly use these commitments in the compiler of [28]. Unfortunately, for the complement problems, such as Graph Isomorphism and Quadratic Residuosity, we only know of instance-dependent commitments that are hiding when $x \in L$, and binding when $x \notin L$.

Thus, for some of our results, we utilize a more sophisticated variant of instance-dependent commitments, due to Bellare, Micali, and Ostrovsky [35]. Specifically, they construct something like a instance-dependent commitment scheme for the Graph Isomorphism problem, but both the hiding and binding properties are non-standard. For example, the binding property is as follows: they show that if $x \in L$ and the sender can open a commitment in two different ways, then it is possible for the sender to extract an **NP** witness for $x \in L$. Thus we call these *witness-binding commitments*. Intuitively, when we use such commitments, we prove concurrent zero knowledge by the following case analysis: either the verifier is bound to its commitments, in which case we can simulate our proof system as in [28], *or* the simulator can extract a witness, in which case it can be simulated by running the honest prover strategy. In reality, however, the analysis does not break into such a simple case analysis, because the verifier may break the commitment scheme in the middle of the protocol. Thus we require that, in such a case, an already-begun simulation can be "continued" once we are given an **NP** witness. Fortunately, the classic (stand-alone) proof systems for Graph Isomorphism and Quadratic Residuosity turn out to have the needed "witness-completable simulation" property.

An additional contribution of our paper is to provide abstractions and generalizations of all of the above tools that allow them to be combined in a modular way, and may facilitate their use in other settings. First, we show how the "preamble" of the Prabhakaran–Rosen–Sahai concurrent zero-knowledge proof system [28] can be viewed as a way to transform any commitment scheme into one that is "concurrently extractable," in the sense that we are able to simulate the concurrent execution of many sessions between an adversarial sender and the honest receiver in a way that allows us to extract the commitments of the sender in every session. This may be useful in constructing other concurrently secure protocols (not just proof systems). Second, we provide general definitions of witness-binding commitment schemes as well as witness-completable zero-knowledge proofs as possessed by Graph Isomorphism and Quadratic Residuosity and as discussed above.

*Perspective.* The recent works of Micciancio and Vadhan [34] and Vadhan [19] hypothesized that every problem that has a statistical (resp., computational) zero-knowledge proof has a instance-dependent commitment scheme.[8] There are several pieces of evidence pointing to this possibility:

1. A restricted form of a complete problem for statistical zero knowledge has a instance-dependent commitment scheme [34].
2. If instance-dependent commitments exist for all problems with statistical zero-knowledge proofs, then instance-dependent commitments exist for all of problems with (general, computational) zero-knowledge proofs [19].

[8] Actually, the works of [34] and [19] refer to instance-dependent commitments where the hiding property holds on yes instances and the binding property on no instances, which is opposite of what we use. For statistical zero knowledge, this does not matter because the class of problems having statistical zero-knowledge proofs is closed under complement [17]. But for computational zero knowledge, it means that outline presented here might yield a concurrent zero-knowledge *argument* system rather than a proof system.

3. Every problem that has (general, computational) zero-knowledge proofs also has inefficient instance-dependent commitments. These commitments are inefficient in the sense that the sender algorithm is not polynomial-time computable [19]. Unfortunately we cannot use these commitments in our protocols in this paper, because our verifier plays the role of the sender.

If the above hypothesis turns out to be true, then our work suggests that we should be able prove that *any* problem that has a zero-knowledge proof has a concurrent zero-knowledge protocol: simply plug the hypothesized instance-dependent commitment scheme into our constructions. (We do not claim this as a theorem because in this paper, we restrict our attention to instance-dependent commitment schemes that are noninteractive and perfectly binding for simplicity, but the hypothesis mentioned above make no such restriction.)

*Outline.* Section 2 details some nonstandard notations that are used in this paper. In Sect. 3, we abstract the preamble stage in the Prabhakaran-Rosen-Sahai concurrent zero-knowledge protocol [28, Sect. 3.1], showing how it transforms any noninteractive commitment scheme into one satisfying a desirable extraction property. In Sect. 4, we apply this transformation to *instance-dependent* commitments, and thereby obtaining some of our concurrent zero-knowledge proofs. In Sect. 5, we extend this transformation to problems with *witness-binding* commitments, and thereby obtaining concurrent zero-knowledge proofs for Quadratic Residuosity and Graph Isomorphism. Many details and proofs are contained in the full version of this paper [1].

## 2 Preliminaries

For the most part, we use standard notations found in the theoretical cryptography and complexity theory literature. In the next few paragraphs, we highlight several nonstandard notations used.

*Transcript and output of interactive protocols.* For an interactive protocol $(A, B)$, let $\langle A, B\rangle(x)$ denote the random variable representing the output of $B$ after interaction with $A$ on common input $x$. In addition, let $\mathrm{view}_B^A(x)$ denote the random variable representing the content of the random tape of $B$ together with the messages received by $B$ from $A$ during the interaction on common input $x$.

*Committed-verifier zero knowledge.* Prabhakaran, Rosen and Sahai [28], in their works on concurrent zero knowledge, showed that adding a $\widetilde{O}(\log n)$-round preamble to a specific form of zero-knowledge protocol (the Hamiltonicity protocol) results in a concurrent zero-knowledge proof system, assuming the existence of a collection of claw-free functions. Alon Rosen, in his PhD thesis, noted that the preamble can be added to a more general form of zero-knowledge protocol, which he informally defines as *challenge-response zero knowledge* [37, Sect. 4.8.1]. We formalize this notion and call it *committed-verifier zero knowledge*.

**Definition 1 (committed-verifier zero knowledge).** *A* committed-verifier *$V_m$, where $m = (m_1, m_2, \ldots, m_k)$, is a deterministic verifier that always sends $m_i$ as its i-th round message.*

*An interactive proof $(P, V)$ for (promise) problem $\Pi$ is* perfect (resp., statistical, computational) committed-verifier zero knowledge (CVZK) *if there exists a probabilistic polynomial-time simulator $S$ such that for all committed verifier $V_m$, the ensembles $\{\mathrm{view}^P_{V_m}(x)\}_{x \in \Pi_Y}$ and $\{S(x, m)\}_{x \in \Pi_Y}$ are perfectly (resp., statistically, computationally) indistinguishable.*

This CVZK property is closely related to notion of *honest-verifier zero knowledge* (HVZK) in that any CVZK protocol is also trivially HVZK. Conversely, any *public-coin* HVZK protocol can be converted into a *public-coin* CVZK protocol by allowing the prover to send random coins $m'$ before the verifier's public-coin message $m$, and making the prover respond to $m' \oplus m$ (instead of just $m$).

**Lemma 2.** *Promise problem $\Pi$ has* public-coin *(perfect/statistical/computational) CVZK proofs if and only if it has* public-coin *(perfect/statistical/computational) HVZK proofs.*

## 3 Concurrently-Extractable Commitment Scheme

### 3.1 Overview

A key component in our concurrent zero-knowledge protocols is a commitment scheme with a *concurrent extractability* property. We call this scheme *concurrently-extractable commitment (CEC) scheme.* The notion of concurrent extractability informally means that we are able to simulate the concurrent execution of many sessions between an adversarial sender and the honest receiver in a way that allows us to extract the commitments of the sender in every session.

This notion of concurrent extractability is inspired by the rewinding and simulation strategy of the Prabhakaran-Rosen-Sahai (PRS) [28] concurrent zero-knowledge protocol. The PRS protocol essentially consists of two stages, the preamble (first) stage and the main (second) stage [28, Sect. 3.1]. The concurrent zero knowledge feature of the protocol comes from the preamble stage, in which the verifier is required to commit to the messages that it will use in the main stage. Our goal in this section is to modularize the PRS protocol by abstracting this key feature (preamble stage) that allows for concurrent security.

### 3.2 Definitions

*Standard commitment schemes.* A standard (interactive) commitment scheme typically consists of a sender $S$, a receiver $R$ and a verification algorithm $\mathsf{Verify}$. A message bit $m \in \{0, 1\}$ is given as private input to $S$, and the common input to both is $1^n$, where $n$ is the security parameter. After the interaction $(S(m), R)(1^n)$, $R$ outputs a commitment string $c$ and $S$ outputs a decommitment pair $(m, d)$. (Without loss of generality, we can assume that $c$ is $R$'s view of the interaction and $d$ is $S$'s coin tosses.) The verification algorithm $\mathsf{Verify}$ checks that $(m, d)$ is a valid decommitment of $c$ by accepting if it is, and rejecting otherwise.

*Commitment schemes with partial verification.* To extend standard commitments to concurrently extractable ones, we require an additional verification procedure denoted as $\mathsf{Partial\text{-}Verify}$, which is needed for the special binding property (see Definition 6).

**Definition 3.** *A* commitment scheme with partial verification *consists of probabilistic polynomial-time algorithms* $(S, R, \mathsf{Verify}, \mathsf{Partial\text{-}Verify})$ *such that the following conditions hold.*

1. *After the interaction* $(S(m), R)(1^n)$*,* $R$ *outputs a commitment string* $c$ *and* $S$ *outputs a decommitment pair* $(m, d)$*.*
2. *For all* $(c, (m, d)) \leftarrow (S(m), R)(1^n)$*, we have that* $\mathsf{Verify}(c, m, d) = 1$*.*
3. *For all* $c$*,* $m$ *and* $d$*,* $\mathsf{Verify}(c, m, d) = 1$ *implies* $\mathsf{Partial\text{-}Verify}(c, m, d) = 1$*.*

A decommitment $(m, d)$ to $c$ with $\mathsf{Verify}(c, m, d) = 1$ is called a *full decommitment*, whereas if we have only that $\mathsf{Partial\text{-}Verify}(c, m, d) = 1$, it is called a *partial decommitment*. Note that a standard commitment scheme is a special case of the above definition by imposing $\mathsf{Partial\text{-}Verify} = \mathsf{Verify}$.

*Remark 4.* Our above notion of a commitment scheme with partial verification shares some similarities with *mercurial commitments*, a notion recently defined in [38]. For our notion, we have a single kind of commit phase that has two kinds of decommitments, a full decommitment and a partial decommitment. For mercurial commitments, the *hard commitments* correspond to our single commit phase, and thus has two kinds of decommitments; standard decommitments and *tease*. Standard decommitments and tease correspond to full decommitments and partial decommitments, respectively. Mercurial commitments also have a notion of *soft commitments* (that cannot be opened with standard decommitments, but can be teased to any value), which we do not require. Mercurial commitments were defined as a primitive for constructing *zero-knowledge sets* [39].

*Statistical hiding and perfect binding.* Definition 3 only refers to the syntax of a commitment scheme, and does not impose any security requirements (e.g., hiding and binding). For that, we have the following two definitions.

**Definition 5 (hiding).** *A commitment scheme with partial verification* $(S, R, \mathsf{Verify}, \mathsf{Partial\text{-}Verify})$ *is* statistically hiding *if for every adversarial receiver* $R^*$*, the ensembles* $\{\langle S(0), R^* \rangle(1^n)\}_{n\in\mathbb{N}}$ *and* $\{\langle S(1), R^* \rangle(1^n)\}_{n\in\mathbb{N}}$ *are statistically indistinguishable.*

The above definition is restricted to statistically hiding since for the purposes of our paper, we will only need to consider statistically hiding commitments. It is straightforward to extend Definition 5 to encompass perfect and computational hiding. Next, we define the *perfectly binding* property for commitment schemes with partial verification. This perfectly binding notion will be used throughout Sect. 4.

**Definition 6 (binding).** *A commitment scheme with partial verification* $(S, R, \mathsf{Verify}, \mathsf{Partial\text{-}Verify})$ *is* perfectly binding *if for every commitment* $c$*, there do* not *exist decommitments* $(m, d)$ *and* $(m', d')$ *such that* $m \neq m'$ *and* $\mathsf{Verify}(c, m, d) = \mathsf{Partial\text{-}Verify}(c, m', d') = 1$.

Intuitively the above definition says that a partial decommitment of $c$ to a message $m$ is a proof that $c$ can only be full decommitted to $m$. Also, observe that Definition 6 implies that the scheme is binding with respect to Verify alone. That is, there do not exist $c$, $(m, d)$ and $(m', d')$ with $m \neq m'$ and $\mathsf{Verify}(c, m, d) = \mathsf{Verify}(c, m', d') = 1$. But the scheme need not be binding with respect to Partial-Verify alone. Hence, the binding property specified in Definition 6 is a natural extension of the binding property of standard commitments (where Partial-Verify = Verify).

*Concurrent simulatability with extractability.* The commitment scheme with partial verification (as in Definition 3) will be used as a building block for our concurrent zero-knowledge protocols in Sects. 4 and 5. For these concurrent zero-knowledge protocols, the prover $P$ and adversarial verifier $V^*$ will play the role of the receiver $R$ and concurrent adversarial sender $\widehat{S}$, respectively. Therefore, we will need to simulate the concurrent interaction between $R$ and $\widehat{S}$, but it turns out this alone is not sufficient. We will also need the simulator to determine partial decommitments of $\widehat{S}$ in every completed session that it has simulated. This property is called *concurrent extractability*, a notion we formalize next.

**Definition 7.** *A commitment scheme with partial verification* $(S, R, \mathsf{Verify}, \mathsf{Partial\text{-}Verify})$ *is* concurrently extractable *if there exists a probabilistic polynomial-time simulator* Sim *such that for every* $Q \leq \mathrm{poly}(n)$*, and for every concurrent adversary* $\widehat{S}$ *that executes at most* $Q$ *concurrent sessions, we have:*

1. (Statistical simulation) *The output of* $\mathsf{Sim}^{\widehat{S}}(1^n, 1^Q)$ *is statistically indistinguishable to the output of* $\widehat{S}$ *in the concurrent interaction* $\langle R, \widehat{S}\rangle(1^n)$.
2. (Concurrent extractability) *Whenever* Sim *queries* $\widehat{S}$ *on a transcript* $T$*, for every completed session* $s$ *in* $T$ *with a commitment* $c[s]$*, it provides partial decommitment* $(m[s], d[s])$ *such that* $\mathsf{Partial\text{-}Verify}(m[s], c[s], d[s]) = 1$.

*For short, we call this a* concurrently-extractable commitment scheme. *Also,* Sim *is called the* concurrently-extracting simulator.

Note that we require that the concurrent extractability property hold for all adversaries $\widehat{S}$, even computationally unbounded ones. The only limitation on $\widehat{S}$ is that it executes at most polynomial sessions, which is a natural restriction since it is infeasible to simulate a superpolynomial number of sessions in polynomial time. In addition, the simulator is only required to provide partial decommitments for every completed session. This suffices because a valid partial decommitment $(m, d)$ of a commitment $c$ effectively binds it to the message $m$ if we insist on a full decommitment later on (see Definition 6).

### 3.3 Construction of Concurrently-Extractable Commitments

A circuit $\mathsf{Com}\colon \{0,1\} \times \{0,1\}^n \to \{0,1\}^n$ can be viewed as a generic (noninteractive) commitment scheme, with $n$ being the security parameter. The commitment to a message bit $m$ is $\mathsf{Com}(m; r)$, where $r \leftarrow \{0,1\}^n$ is a uniformly chosen random key. Likewise, the decommitment of $c$ to a bit $m$ is a pair $(m, r)$ such that $c = \mathsf{Com}(m; r)$. Note that this definition only refers to the syntax of a commitment scheme and does not impose any security requirements (i.e., hiding and binding).

The next lemma states that we can transform any generic commitment scheme $\mathsf{Com}\colon \{0,1\} \times \{0,1\}^n \to \{0,1\}^n$ into a new scheme with the concurrent extractability property. This new scheme is essentially the preamble stage of the PRS concurrent ZK protocol [28], with the sender (verifier) using $\mathsf{Com}$ to commit in the $\widetilde{O}(\log n)$ rounds of interaction, and the receiver (prover) just sending random coins.

**Lemma 8.** *For any generic noninteractive commitment scheme* $\mathsf{Com}\colon \{0,1\} \times \{0,1\}^n \to \{0,1\}^n$*, there is a* concurrently-extractable *commitment scheme* $C_{\mathsf{Com}} = (S_{\mathsf{Com}}, R_{\mathsf{Com}}, \mathsf{Verify}_{\mathsf{Com}}, \mathsf{Partial\text{-}Verify}_{\mathsf{Com}})$ *(taking the circuit* $\mathsf{Com}$ *as auxiliary input), such that:*

1. *If* $\mathsf{Com}$ *is perfectly binding, then* $C_{\mathsf{Com}}$ *is perfectly binding.*
2. *If* $\mathsf{Com}$ *is statistically hiding, then* $C_{\mathsf{Com}}$ *is statistically hiding.*
3. $(S_{\mathsf{Com}}, R_{\mathsf{Com}})$ *has* $\widetilde{O}(\log n)$ *rounds of interaction.*

*We denote* $\mathsf{CEC\text{-}Sim}_{\mathsf{Com}}$ *as the concurrently-extracting simulator for* $C_{\mathsf{Com}}$.

*Committing to multi-bit messages.* The concurrently-extractable commitment scheme obtained from Lemma 8 is for a single-bit message; to commit to a $\ell$-bit message, we independently repeat the scheme $\ell$ times in parallel. It is important to note that even if we do so, all the properties required in Definition 7 still hold. (Concurrent extractability follows because parallel repetition is a special case of concurrent interaction.) Later in Sect. 4, it will be more convenient to think of $\widehat{S}$ as committing to an $\ell$-bit message per session, rather than $\ell$ senders committing to a single-bit message each.

Finally, when $S$ commits to multi-bit messages, it can full-decommit in multiple steps, one for each committed bit. This is because the full decommitment for each bit of the message is independent of the others.

## 4 Unconditional Concurrent Zero-Knowledge Proofs for Problems with Instance-Dependent Commitments

In this section, we demonstrate a generic technique for transforming certain stand-alone public-coin zero-knowledge protocols into concurrent zero-knowledge protocols. In doing so, we construct unconditional concurrent zero-knowledge proofs for non-trivial problems like QUADRATIC NONRESIDUOSITY, GRAPH NON-ISOMORPHISM, a variant of STATISTICAL DIFFERENCE and approximate lattice problems.

The main tool used in the transformation is a *instance-dependent commitment scheme*, formally defined in Definition 9. Later in Sect. 5, we demonstrated a modified transformation that works for certain problems possessing *witness-binding commitments*.

### 4.1 Instance-Dependent Commitments

In order to prevent the adversarial verifier from deviating widely from the original protocol specification, the previous constructions of concurrent zero-knowledge protocols require the verifier to commit to certain messages in advance [23, 25, 28]. While these commitments can be constructed from one-way functions [40, 41], proving the existence of one-way functions remains a major open problem in complexity theory.

To achieve concurrent security without relying on unproven assumptions, we observe that the standard verifier's commitments used in [28] can be replaced by *instance-dependent commitments* [36] (cf., [34]). A instance-dependent commitment, roughly speaking, is a commitment protocol that takes the problem instance $x$ as an additional input, is binding on the YES instances ($x \in \Pi_{\mathrm{Y}}$), and is hiding on the NO instances ($x \in \Pi_{\mathrm{N}}$). Standard commitments, by contrast, are required to always be both hiding and binding regardless of the problem instance.

Because the hiding and binding properties of instance-dependent commitments depend on the problem instance, we can construct instance-dependent commitments that are both perfectly binding (on the YES instances) and statistically hiding (on the NO instances).[9] We give a simplified, noninteractive definition of instance-dependent commitments that suffices for our applications in this section.

**Definition 9 (noninteractive instance-dependent commitment).** *Promise problem $\Pi = (\Pi_{\mathrm{Y}}, \Pi_{\mathrm{N}})$ has a* instance-dependent commitment *if there exists a polynomial-time algorithm* $\mathsf{PD\text{-}Com}$ *such that the following holds.*

1. *Algorithm* $\mathsf{PD\text{-}Com}$ *takes as input the problem instance $x$, a bit $b$, and a random key $r$, and produces a commitment $c = \mathsf{PD\text{-}Com}_x(b; r)$. The running time of* $\mathsf{PD\text{-}Com}$ *is bounded by a polynomial in $|x|$, hence without loss of generality we can assume that $|c| = |r| = \mathrm{poly}(|x|)$.*
2. *(perfectly binding on YES instances) For all $x \in \Pi_{\mathrm{Y}}$, the distributions* $\mathsf{PD\text{-}Com}_x(0)$ *and* $\mathsf{PD\text{-}Com}_x(1)$ *have disjoint supports. That is, there does not exist strings $r$ and $r'$ such that* $\mathsf{PD\text{-}Com}_x(0; r) = \mathsf{PD\text{-}Com}_x(1; r')$.

[9] By contrast, standard commitments *cannot* be both statistically binding and statistically hiding.

3. *(statistically hiding on* NO *instances) For all $x \in \Pi_{\mathrm{N}}$, the commitments to* 0 *and* 1 *are statistically indistinguishable. In other words, the distributions* $\mathsf{PD\text{-}Com}_x(0)$ *and* $\mathsf{PD\text{-}Com}_x(1)$ *are statistically indistinguishable (w.r.t. $|x|$, the length of the instance).*

*The commitment $c$ can be decommitted to by sending the committed bit $b$ and random key $r$. Since both parties have access to the problem instance $x$, this decommitment can be verified by checking that $c = \mathsf{PD\text{-}Com}_x(b; r)$.*

### 4.2 Main Results

Before presenting the our unconditional concurrent zero-knowledge protocol, we state our main results for this section.

**Theorem 10.** *If promise problem $\Pi$ has a public-coin CVZK proof system $(P_0, V_0)$ (in the sense of Definition 1) and also a instance-dependent commitment, then $\Pi$ has a proof system $(P, V)$ with the following properties:*

1. *If $(P_0, V_0)$ is statistical (resp., computational) zero knowledge, then $(P, V)$ is* concurrent *statistical (resp., computational) zero knowledge.*
2. *Prover $P$ is black-box simulatable in* strict *polynomial time.*
3. *The round complexity of $(P, V)$ increases only by an additive factor of $\widetilde{O}(\log n)$, with $n$ being the security parameter, compared to the original protocol $(P_0, V_0)$.*
4. *The completeness of $(P, V)$ is exactly the same as that of $(P_0, V_0)$, while the soundness error increases by only a negligible additive term (as a function of $n$).*
5. *The prover strategy $P$ can be implemented in probabilistic polynomial-time with oracle access to $P_0$. In particular, if $P_0$ is efficient, so is $P$.*

We provide an outline of the proof of Theorem 10 in Sects. 4.3 and 4.4. Several natural problems that Theorem 10 applies to are listed below.

**Corollary 11.** *The following problems have concurrent statistical zero-knowledge proofs:*

- *The statistical difference problem* $\mathrm{SD}^1_{1/2}$.
- *The languages* QUADRATIC NONRESIDUOSITY *and* GRAPH NONISOMORPHISM.
- *The lattice problems* co-GAPCVP$_\gamma$ *and* co-GAPSVP$_\gamma$, *for $\gamma = \Omega(\sqrt{(n/\log n)})$.*

*Proof.* All the problems listed—$\mathrm{SD}^1_{1/2}$, QUADRATIC NONRESIDUOSITY, GRAPH NONISOMORPHISM, co-GAPCVP$_\gamma$ and co-GAPSVP$_\gamma$, for $\gamma = \Omega(\sqrt{(n/\log n)})$—have honest-verifier statistical zero-knowledge proofs [2, 5, 42, 33], which can be made public-coin by [17]. In addition, they all have instance-dependent commitments [36, 34].

The above corollary does not guarantee a polynomial-time prover strategy (with auxiliary input) nor round efficiency. The reason is that the public-coin honest-verifier zero-knowledge proof systems known for these problems do not have a polynomial-time prover nor a subpolynomial number of rounds. For Blum Quadratic Nonresiduosity,[10] however, we can start with the noninteractive statistical zero-knowledge proof[11] of [43], whose prover is polynomial time (given the factorization of the modulus), and obtain the following:

**Corollary 12.** *The language* Blum Quadratic Nonresiduosity *has a concurrent statistical zero-knowledge proof systems with $\widetilde{O}(\log n)$ rounds and a prover that can be implemented in polynomial time given the factorization of the input modulus.*

We note that we do not expect to obtain efficient provers for Graph NonIsomorphism or $\mathrm{SD}^1_{1/2}$, since these problems are not known to be in **NP** (or **MA**), which is a prerequisite for an efficient-prover proof system. However, Quadratic Nonresiduosity is in **NP** (the factorization of the input is a witness), as are co-GapCVP$_\gamma$ and co-GapSVP$_\gamma$ for larger approximation factors $\gamma = \Omega(\sqrt{n})$ [44], so we could hope to obtain an efficient prover. The bottleneck is finding *public-coin* honest-verifier zero-knowledge proofs with a polynomial-time prover for these problems.

### 4.3 Our Concurrent Zero-Knowledge Protocol

A high-level description of our unconditional concurrent zero-knowledge protocol is as follows: We begin with a public-coin CVZK protocol. We make it concurrent zero knowledge by forcing the verifier to commit in advance to its (public-coin) messages in the CVZK protocol using concurrently-extractable commitments $C_{\mathsf{Com}}$ provided for by Lemma 8. However, $C_{\mathsf{Com}}$ still requires a generic noninteractive commitment scheme $\mathsf{Com}$; for this, we plug-in the instance-dependent commitment scheme $\mathsf{PD\text{-}Com}_x$.

Now, let us formally describe our concurrent zero-knowledge protocol. Let $(P_0, V_0)$ be a public-coin CVZK proof system for $\Pi$ with $q(|x|)$ rounds on common input $x$. Denote the messages sent by $V_0$ in the protocol as $m = (m_1, \ldots, m_q)$, and let $\ell \stackrel{\text{def}}{=} |m|$ be the verifier-to-prover communication complexity. Let $\mathsf{PD\text{-}Com}_x \colon \{0,1\} \times \{0,1\}^n \to \{0,1\}^n$, where $n = \mathrm{poly}(|x|)$, be a instance-dependent commitment for $\Pi$. The full description of our concurrent zero-knowledge protocol $(P, V)$ is next.

[10] The problem Blum Quadratic Nonresiduosity is a variant of quadratic residuosity restricted to Blum integers.

[11] Noninteractive zero knowledge implies (in fact is equivalent to) 2-round public-coin honest-verifier zero knowledge since the honest verifier just sends the common random string in the first round, and the prover sends the single-message proof in the second round.

**Protocol 13** *Our unconditional concurrent zero-knowledge protocol $(P, V)$ for problem $\Pi$ with instance-dependent commitments.*

**Input:** Instance $x$ of $\Pi$.

**Preamble stage (using instance-dependent commitments)**

Let $C_{\mathsf{PD\text{-}Com}_x} = (S_x, R_x, \mathsf{Verify}_x, \mathsf{Partial\text{-}Verify}_x)$ be the concurrently-extractable commitment scheme provided for by Lemma 8 by substituting $\mathsf{Com} = \mathsf{PD\text{-}Com}_x$.

$V$ : Select a random message $m = (m_1, \ldots, m_q) \leftarrow \{0,1\}^\ell$.

$V \rightarrow P$: Send the message `"start session"`.

$V \leftrightarrow P$: Run the following instance-dependent CEC schemes $(S_x(m_1), R_x)(1^n), \cdots, (S_x(m_q), R_x)(1^n)$ in parallel, with the verifier $V$ acting as $S_x$ and the prover $P$ as $R_x$.

Let the output of $R_x$ be the commitments $(c_1, \ldots, c_q)$, and be the output of $S_x$ be the decommitments $((m_1, d_1), \ldots, (m_q, d_q))$. Note that neither $P$ nor $V$ sends the outputs of $R_x$ or $S_x$ to the other party at this stage.

**Main stage (stand-alone zero-knowledge protocol)**

$V \rightarrow P$: Send the message `"start main stage"`.

$P$ : Select randomness $r_{P_0} \leftarrow \{0,1\}^*$ for the original prover $P_0$.

For $t = 1, \ldots, q$, do the following:

$V \rightarrow P$: Decommit to $m_t$ by sending full decommitment $(m_t, d_t)$ of $c_t$.

$P \rightarrow V$: Verify the decommitment received is valid by checking if $\mathsf{Verify}(c_t, m_t, d_t) = 1$. If so, answer as the original prover $P_0$ would, that is, send $\pi_t = P_0(x, m_1, \ldots, m_t; r_{P_0})$. Otherwise, halt and abort.

Verifier $V$ accepts if the original verifier $V_0$ accepts on $(m_1, \pi_1, \ldots, m_q, \pi_q)$.

From Protocol 13 and Lemma 8, we can easily derive the prover efficiency, round complexity and completeness claims of Theorem 10. For soundness, observe that since $\mathsf{PD\text{-}Com}_x$ is statistically hiding, $C_{\mathsf{PD\text{-}Com}_x}$ is also statistically hiding (by Lemma 8). Hence, the soundness of Protocol 13 only decreases by a negligible amount because a cheating prover will not know the committed messages of the verifier until the verifier decommits to $m_t$ (in round $t$ of the main stage).

We show that Protocol 13 is concurrent zero-knowledge by highlighting the main ideas behind its simulation in the next subsection.

### 4.4 Our Simulator

Observe that the prover's strategy can be broken into two parts, $P_{\text{pre}}$ and $P_{\text{main}}$, denoting the preamble stage and main stage, respectively. Both $P_{\text{pre}}$ and $P_{\text{main}}$ use independent randomness. The simulation procedure for our concurrent zero-knowledge protocol (Protocol 13) is broken into three main steps.

1. First, we analyze the concurrent interaction of $P$ and $V^*$ in the context of concurrently-extractable commitment schemes (provided for by Lemma 8, substituting $\mathsf{Com} = \mathsf{PD\text{-}Com}_x$). To do so, we define a new adversarial sender $\widehat{S}$ that takes $V^*$ and $P_{\text{main}}$ as oracles and only returns the preamble messages of $V^*$. The preamble stage prover $P_{\text{pre}}$ acts as the honest receiver $R_x$. By Definition 7 and Lemma 8, we can simulate the output of $\widehat{S}$ (after interaction with $P_{\text{pre}}$), while having the additional property of being able to extract the commitments.
   By virtue of the way we defined $\widehat{S}$, its output after concurrently interacting with $P_{\text{pre}}$ is equivalent to the output of $V^*$ after concurrently interacting with $P$. Nevertheless, this simulation is inefficient because $\widehat{S}$ uses an oracle for $P_{\text{main}}$.
2. Since we can extract partial decommitments, we are able to determine the verifier's main stage messages in advance.[12] Hence, we can replace the adaptive queries to $P_{\text{main}}$ by a single query made to a new oracle, called $\mathcal{O}_P$, at the start of each main stage.
3. However, $\mathcal{O}_P$ is still not an efficiently implementable oracle. In the final step, we replace oracle $\mathcal{O}_P$ with a committed-verifier zero knowledge (CVZK) simulator $S_{\text{CVZK}}$ to obtain an efficient simulation strategy.

## 5 Unconditional Concurrent Zero-Knowledge Proofs for Problems with Witness-Binding Commitments

Here we extend the techniques in Sect. 4 to obtain unconditional concurrent statistical zero-knowledge proofs for certain problems like Quadratic Residuosity and Graph Isomorphism. These problems are not known to have instance-dependent commitments (in the sense of Definition 9), but have a variant of instance-dependent commitments called *witness-binding commitments* (see Sect. 5.1). Informally, these commitments are not guaranteed to be perfectly binding but breaking the binding property of these commitments is as hard as finding a witness.

Using these witness-binding commitments, we proceed to transform them into ones with the concurrently extractability property. (In Sect. 3.3 we did a similar transformation for standard instance-dependent commitments.) Our concurrent zero-knowledge protocol combines the witness-binding concurrently-extractable commitments with an underlying stand-alone ZK protocol.

[12] The binding property in the sense of Definition 6 allows us to determine the committed message in any valid full decommitment by just knowing a partial decommitment.

Recall that in Sect. 4, we required the stand-alone protocol to be committed-verifier zero knowledge (CVZK), as in Definition 1. However, since we are using only witness-binding commitments, we require the underlying stand-alone protocol to have a stronger property that we call *witness-completable* CVZK (see Sect. 5.2). The additional witness-completable property, informally stated, gives our simulator the ability to complete the simulation even when the verifier sends a message different from its committed one, if we provide our simulator with a valid witness at that time. This is important because the binding property of witness-binding commitments can be broken, but if that is the case, the simulator can obtain a witness that it can use to complete the simulation.

### 5.1 Witness-Binding Commitments

Based on the techniques used in Sect. 4, the first natural step towards constructing concurrent zero-knowledge protocols would be to construct instance-dependent commitments. Consider the naive commitment scheme for Graph Isomorphism specified as follows: Let $(G_0, G_1)$ be an instance of the problem. To commit to bit $b$, send a random isomorphic copy of $G_b$. This commitment is perfectly hiding on the YES instances (when $G_0 \cong G_1$) and perfectly binding on the NO instances (when $G_0 \not\cong G_1$). However, this is exactly the opposite of what we require in a instance-dependent commitment (see Definition 9). In fact, every problem satisfying Definition 9 is in **coNP**, but Graph Isomorphism is not known to be in **coNP**.

**Protocol 14** *Witness-binding commitment scheme for* Graph Isomorphism *(implicit in [35]).*

To commit to bit $b$ using problem instance $(G_0, G_1)$, proceed as follows.

**Index generation stage**

$R \to S$: Let $H_1$ be a random isomorphic copy of $G_0$, and send $H_1$. That is, $H_1 = \sigma(G_0)$ for a random permutation $\sigma$ of the vertices of $G_0$. In addition, both parties set $H_0 = G_0$.

**Commitment stage**

$S \to R$: To commit to bit $b$, send $F$, a random isomorphic copy of $H_b$.

**Decommitment stage**

$S \to R$: To decommit, send $b$ together with the isomorphism between $H_b$ and $F$.

**Verification stage**

After the decommitment stage, the receiver $R_x$ proves that $H_1$, sent in the index generation stage, is isomorphic to $G_0$ by sending the isomorphism $\sigma$ between $G_0$ and $H_1$.

To overcome this apparent difficulty, the above commitment scheme (Protocol 14) makes use of additional index generation and verification stages to do instance-dependent commitments. It can be shown that this witness-binding commitment scheme is perfectly hiding on every instance (in particular the NO instances) if $H_1$ is generated correctly, that is if $H_1 \cong G_0$. On the YES instances, the scheme is "computationally binding" in that breaking the scheme is as hard as finding an **NP**-witness (an isomorphism between $G_0$ and $G_1$). More precisely, we can extract the witness if we use a *simulated* index generation stage, where $H_1$ is taken to be a random isomorphic copy of $G_1$ (which is distributed identically to the actual index generation).

This scheme can be generalized to a number of other **NP** languages, and a formal definition capturing the notion of witness-binding commitments is in the full version of this paper [1]. In addition, we note that QUADRATIC RESIDUOSITY has a similarly structured witness-binding commitment scheme (based on Protocol 14 and its 3-round perfect zero-knowledge proof system [2]).

### 5.2 Witness-Completable CVZK

Recall that witness-completable CVZK (wCVZK) is a strengthening of the notion of CVZK (Definition 1) in that our simulator, when given a valid witness, must have the ability to complete the simulation even when the verifier sends a message different from its committed one. The formal definition of wCVZK is the full version of this paper [1].

The 3-round perfect zero-knowledge protocols for both QUADRATIC RESIDUOSITY [2] and GRAPH ISOMORPHISM [5] turns out to have the witness-completable property, as desired.

### 5.3 Main Results

Our main result for this section can be summarized in a very similar manner as Theorem 10 in Sect. 4.2. The main differences are (1) the promise problem $\Pi$ needs to have a witness-binding commitment scheme and a 3-round, public-coin, wCVZK proof system (instead of instance-dependent commitment scheme and CVZK proof system), and (2) our new simulation runs in *expected* polynomial time instead of strict polynomial time. With that, we obtain the following theorem.

**Theorem 15.** *Both languages* GRAPH ISOMORPHISM *and* QUADRATIC RESIDUOSITY *have concurrent statistical zero-knowledge proof systems with $\widetilde{O}(\log n)$ rounds and efficient provers. The simulator for both protocols runs in* expected *polynomial time.*

Note that the round complexity of $\widetilde{O}(\log n)$ for the concurrent zero-knowledge protocols of both GRAPH ISOMORPHISM and QUADRATIC RESIDUOSITY is essentially optimal for black-box simulation [27].

### 5.4 Our Modified Concurrent Zero-Knowledge Protocol

Since we are dealing with witness-binding commitments, we have to modify Protocol 13 in Sect. 4.3. Our modified concurrent zero-knowledge protocol is similar in structure with the main difference being that instead of just the preamble stage and the main stage, it also an index generation stage before the preamble stage and a verification stage after the main stage (for implementing the corresponding stages of the witness-binding commitment scheme). The full description of our modified protocol is in the full version of this paper [1].

### 5.5 Our Simulator

Recall the three main steps of the simulation procedure in Sect. 4.4.

1. Analyze the concurrent interaction of $P$ and $V^*$ in the context of the concurrently-extractable commitment schemes. Specifically, define a new adversarial sender $\widehat{S}$ that takes $V^*$ and $P_{\text{main}}$ as oracles and only returns the preamble messages of $V^*$, and simulate its interaction while extracting its commitments.
2. Replace the adaptive queries to $P_{\text{main}}$ by a single query made to a new oracle, called $\mathcal{O}_P$, at the start of each main stage.
3. Replace oracle $\mathcal{O}_P$ with a CVZK simulator $S_{\text{CVZK}}$ to obtain an efficient simulation strategy.

For the simulation of our modified concurrent zero-knowledge protocol, we keep Step 1 the same, but in Step 2 observe that the prover responses provided by $\mathcal{O}_P$ depends on the witness $w$ given the to prover. Hence, we denote it more precisely as $\mathcal{O}_{P(w)}$. In Step 3, we simulate the answers from $\mathcal{O}_{P(w)}$ with our wCVZK simulator. However, our wCVZK simulator needs a witness $w$ in order to continue the simulation when the verifier's $V^*$ response does not match the expectation of our simulator.

This can only happen if $V^*$ breaks the binding of the witness-binding commitment. And when that happens, our simulator is able to obtain a witness $w$, which it can then feed to the wCVZK simulator to continue the simulation. Actually, a subtlety is that the witness-binding commitment allows us to extract a witness only if we *simulate* the index generation stage, whereas here we need to run the actual index generation in order to complete the verification stage. Thus, if needed, we run a separate offline process to extract a witness, and this is what causes our simulator to run in expected polynomial time. For details, see the full version of this paper [1].

Acknowledgements. We thank Alexander Healy, Manoj Prabhakaran and Alon Rosen for helpful discussions.

## References


1. Micciancio, D., Ong, S.J., Sahai, A., Vadhan, S.: Concurrent zero knowledge without complexity assumptions. Technical Report 05-093, Electronic Colloquium on Computational Complexity (2005) http://eccc.uni-trier.de/eccc-reports/2005/TR05-093/.
2. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing **18**(1) (1989) 186–208
3. Goldreich, O., Krawczyk, H.: On the composition of zero-knowledge proof systems. SIAM Journal on Computing **25**(1) (1996) 169–192
4. Dwork, C., Naor, M., Sahai, A.: Concurrent zero-knowledge. In: Proc. 30th STOC. (1998) 409–418
5. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. Journal of the ACM **38**(1) (1991) 691–729
6. Goldreich, O.: Foundations of cryptography. Volume 1. Cambridge University Press, Cambridge, UK (2001)
7. Goldreich, O.: Zero-knowledge twenty years after its invention. http://www.wisdom.weizmann.ac.il/~oded/zk-tut02.html (2002)
8. Yao, A.C.: How to generate and exchange secrets. In: Proc. 27th FOCS. (1986) 162–167
9. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attack. In: Proc. 22nd STOC. (1990) 427–437
10. Dolev, D., Dwork, C., Naor, M.: Nonmalleable cryptography. SIAM Journal on Computing **30**(2) (2001) 391–437
11. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: Proc. 40th FOCS. (1999) 543–553
12. Feige, U., Fiat, A., Shamir, A.: Zero-knowledge proofs of identity. Journal of Cryptology **1**(2) (1988) 77–94
13. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Proc. CRYPTO '98. (1998) 13–25
14. Elkind, E., Sahai, A.: A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive, Report 2002/042 (2002) http://eprint.iacr.org/.
15. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing **33**(1) (2004) 167–226
16. Gennaro, R., Micciancio, D., Rabin, T.: An efficient non-interactive statistical zero-knowledge proof system for quasi-safe prime products. In: Proc. of the 5th ACM Conference on Computer and Communications Security. (1998) 67–72
17. Okamoto, T.: On relationships between statistical zero-knowledge proofs. Journal of Computer and System Sciences **60**(1) (2000) 47–108
18. Goldreich, O., Sahai, A., Vadhan, S.: Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In: Proc. 30th STOC. (1998) 399–408
19. Vadhan, S.: An unconditional study of computational zero knowledge. In: Proc. 45th STOC. (2004) 176–185
20. Ostrovsky, R.: One-way functions, hard on average problems, and statistical zero-knowledge proofs. In: Proceedings of the Sixth Annual Structure in Complexity Theory Conference. (1991)

21. Ostrovsky, R., Wigderson, A.: One-way functions are essential for non-trivial zero-knowledge. In: Second Israel Symposium on Theory of Computing Systems. (1993) 3–17
22. Feige, U.: Alternative models for zero knowledge interactive proofs. PhD thesis, Weizmann Institute of Science, Israel (1990)
23. Richardson, R., Kilian, J.: On the concurrent composition of zero-knowledge proofs. In: Proc. EUROCRYPT '99. (1999) 415–431
24. Kilian, J., Petrank, E., Rackoff, C.: Lower bounds for zero knowledge on the Internet. In: Proc. 39th FOCS. (1998) 484–492
25. Kilian, J., Petrank, E.: Concurrent and resettable zero-knowledge in poly-logarithm rounds. In: Proc. 33rd STOC. (2001) 560–569
26. Rosen, A.: A note on the round-complexity of concurrent zero-knowledge. In: Proc. CRYPTO '00. (2000) 451–468
27. Canetti, R., Kilian, J., Petrank, E., Rosen, R.: Black-box concurrent zero-knowledge requires (almost) logarithmically many rounds. SIAM Journal on Computing **32**(1) (2003) 1–47
28. Prabhakaran, M., Rosen, A., Sahai, A.: Concurrent zero knowledge with logarithmic round-complexity. In: Proc. 43rd FOCS. (2002) 366–375
29. Barak, B.: How to go beyond the black-box simulation barrier. In: Proc. 42nd FOCS. (2001) 106–115
30. Di Crescenzo, G.: Removing complexity assumptions from concurrent zero-knowledge proofs. In: Proc. 6th COCOON. (2000) 426–435
31. Micciancio, D., Petrank, E.: Simulatable commitments and efficient concurrent zero-knowledge. In: Proc. EUROCRYPT '03. (2003) 140–159
32. Brassard, G., Chaum, D., Crepeau, C.: Minimum disclosure proofs of knowledge. Journal of Computer and System Sciences **37**(2) (1988) 156–189
33. Sahai, A., Vadhan, S.: A complete problem for statistical zero knowledge. Journal of the ACM **50**(2) (2003)
34. Micciancio, D., Vadhan, S.: Statistical zero-knowledge proofs with efficient provers: lattice problems and more. In: Proc. CRYPTO '03. (2003) 282–298
35. Bellare, M., Micali, S., Ostrovsky, R.: Perfect zero-knowledge in constant rounds. In: Proc. 22nd STOC. (1990) 482–493
36. Itoh, T., Ohta, Y., Shizuya, H.: A language-dependent cryptographic primitive. Journal of Cryptology **10**(1) (1997) 37–49
37. Rosen, A.: The Round-Complexity of Black-Box Concurrent Zero-Knowledge. PhD thesis, Weizmann Institute of Science, Israel (2003)
38. Chase, M., Healy, A., Lysyanskaya, A., Malkin, T., Reyzin, L.: Mercurial commitments with applications to zero-knowledge sets. In: Proc. EUROCRYPT '05. (2005) 422–439
39. Micali, S., Rabin, M.O., Kilian, J.: Zero-knowledge sets. In: Proc. 44th FOCS. (2003) 80–91
40. Naor, M.: Bit commitment using pseudorandomness. Journal of Cryptology **4**(2) (1991) 151–158
41. Hastad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM Journal on Computing **28**(4) (1999) 1364–1396
42. Goldreich, O., Goldwasser, S.: On the limits of nonapproximability of lattice problems. Journal of Computer and System Sciences **60**(3) (2000) 540–563
43. Blum, M., De Santis, A., Micali, S., Persiano, G.: Noninteractive zero-knowledge. SIAM Journal on Computing **20**(6) (1991) 1084–1118
44. Aharonov, D., Regev, O.: Lattice problems in NP $\cap$ coNP. In: Proc. 45th FOCS. (2004) 362–371