Strengthening Zero-Knowledge Protocols Using Signatures

Juan A. Garay[1], Philip MacKenzie[1], and Ke Yang[2]

[1] Bell Labs – Lucent Technologies, 600 Mountain Ave., Murray Hill, NJ 07974, USA.
{garay,philmac}@research.bell-labs.com
[2] Computer Science Dept., Carnegie Mellon University, Pittsburgh, PA 15213, USA.
yangke@cs.cmu.edu.

Abstract. Recently there has been an interest in zero-knowledge protocols with stronger properties, such as concurrency, unbounded simulation soundness, non-malleability, and universal composability. In this paper, we show a novel technique to convert a large class of existing honest-verifier zero-knowledge protocols into ones with these stronger properties in the common reference string model. More precisely, our technique utilizes a signature scheme existentially unforgeable against adaptive chosen-message attacks, and transforms any Σ-protocol (which is honest-verifier zero-knowledge) into an unbounded simulation sound concurrent zero-knowledge protocol. We also introduce *Ω-protocols*, a variant of Σ-protocols for which our technique further achieves the properties of non-malleability and/or universal composability.

In addition to its conceptual simplicity, a main advantage of this new technique over previous ones is that it avoids the Cook-Levin theorem, which tends to be rather inefficient. Indeed, our technique allows for very efficient instantiation based on the security of some efficient signature schemes and standard number-theoretic assumptions. For instance, one instantiation of our technique yields a universally composable zero-knowledge protocol under the Strong RSA assumption, incurring an overhead of a small constant number of exponentiations, plus the generation of two signatures.

1 Introduction

The concept of a zero-knowledge (ZK) proof, as defined by Goldwasser, Micali, and Rackoff [25], has become a fundamental tool in cryptography. Informally, if a prover proves a statement to a verifier in ZK, then the verifier gains no information except for being convinced of the veracity of that statement. In particular, whatever the verifier could do after the ZK proof, it could have done before the ZK proof, in some sense because it can "simulate" the proof itself. In early work, Goldreich, Micali and Wigderson [24] showed that any NP statement could be proven in (computational) ZK. In another early work, Goldreich, Micali and Wigderson [23] showed the usefulness of ZK proofs in multiparty protocols, in particular, in having the parties prove the correctness of their computations. There has been a great deal of work since then on all properties of ZK proofs. Here we focus on a few such properties, namely, concurrency, non-malleability, simulation soundness, and universal composability, with our main goal being to construct *efficient* protocols that achieve these properties.

The problem of concurrency was first discussed in Dwork, Naor and Sahai [17]. Informally, the problem arises when many verifiers are interacting with a prover. An adversary controlling all the verifiers may coordinate the timing of their messages so that a simulator would not be able to simulate the execution of the prover in polynomial time. Canetti *et al.* [7] showed that without additional assumptions, such as timing constraints or a common reference string, logarithmic rounds are necessary to achieve concurrent (black-box) ZK. Prabhakaran, Rosen, and Sahai [37] showed that logarithmic rounds suffice. On the other hand, Damgård [13] showed that concurrent, constant-round ZK protocols can be achieved in the common reference string model. Furthermore, Barak [1] showed that by using a non black-box simulator, constant-round, concurrent protocols can be constructed in the plain model.[3]

The problem of malleability was first pointed out by Dolev, Dwork and Naor [16]. Roughly speaking, the problem is that an adversary may be able to play a "man-in-the-middle" attack on a ZK protocol, playing the role of the verifier in a first protocol, and that of the prover in a second protocol, and such that using information from the first protocol he is able to prove something in the second protocol that he could not prove without that information. A ZK protocol that does not suffer from this problem is said to achieve *one-time non-malleability* (since the adversary only interacts with one prover). Dolev, Dwork and Naor give a construction of a one-time non-malleable ZK protocol that uses a polylogarithmic number of communication rounds. Katz [28] describes efficient protocols for one-time non-malleable proofs of plaintext knowledge for several encryption schemes. His protocols work in the common reference string model, and consist of three rounds and constant number of exponentiations. However, since the witness extractor uses "rewinding," the resulting protocols were only proven secure in a concurrent setting with the introduction of timing constraints. Barak [2] gives a construction of constant-round, one-time non-malleable ZK protocols in the plain model. His construction uses a non-blackbox proof of security and is not very efficient. Sahai [40] provides a definition for one-time non-malleability in the case of non-interactive ZK (NIZK) proofs. De Santis *et al.* [15] generalize this to *unbounded non-malleability* of NIZK proofs, where even any polynomial number of simulator-constructed proofs does not help an adversary to construct any new proof. (As they do, for the remainder of this paper we will simply refer to this property as *non-malleability*, leaving off the "unbounded" modifier.) Their definition is very strong in that (in some sense) it requires a witness to be extractable from the adversary. They give two constructions of non-malleable ZK proofs for any NP language. In fact, these proofs are non-interactive, and thus achieve concurrent (constant-round) ZK.

[3] His construction, however, only admits bounded concurrency, meaning that the number of sessions that the protocol can execute concurrently and still retain its zero-knowledge property is at most a *fixed* polynomial in the security parameter.

The notion of simulation soundness for NIZK proofs was introduced by Sahai [40] in the context of chosen-ciphertext security of the Naor-Yung [33] encryption scheme. Informally, an NIZK proof is one-time simulation sound if even after seeing a "simulated proof" (which could be of a false statement) generated by the simulator, the adversary cannot generate a proof for a false statement. Sahai notes that the Naor-Yung encryption scheme would be adaptive chosen-ciphertext secure if it used a one-time simulation-sound NIZK proof. De Santis *et al.* [15] further generalized this notion to *unbounded simulation soundness*. An NIZK proof is unbounded simulation sound if even after seeing any polynomial number of simulated proofs, the adversary cannot generate a proof of a false statement. The non-malleable NIZK protocols given in [15] are also unbounded simulation sound.

The notions of unbounded simulation soundness and non-malleability extend naturally to the case of interactive proof systems; we do this in Section 2.

Universal composability is a notion proposed by Canetti [5] to describe protocols that behave like ideal functionalities, and can be composed in arbitrary ways. Universal composability can be defined in either the adaptive model or the static model, denoting whether the adversary is allowed to adaptively corrupt parties, or must decide which parties to corrupt before the protocol starts, respectively. Universal composability is a very strong notion. For example, a universally composable ZK (UCZK) protocol is both non-malleable (at least in an intuitive sense) and concurrent.

Canetti [5] proved that UCZK protocols do not exist in the "plain" model, where there is no assumption about the system set-up. On the other hand, UCZK is possible in the common reference string model, which is the model we focus on in this paper. As pointed out by Canetti *et al.* [8], the non-malleable NIZK protocols of [15] are also UCZK protocols in the static corruption model. Since they use non-interactive proof techniques and general NP reductions, these protocols are not very efficient. Canetti and Fischlin [6] give a construction of a UCZK protocol for any NP language secure in the adaptive model. Basically, they use a standard three-round ZK protocol for Hamiltonian Cycle, except that they use universally composable commitments as a building block. Damgård and Nielsen [14] use the same general ZK protocol construction as Canetti and Fischlin, but with a more efficient UC commitment scheme.[4] Specifically, for a security parameter k, their UC commitment scheme allows commitment to k bits using a constant number of exponentiations and $O(k)$ bits of communication. Their most efficient UC commitment schemes are based on the p-subgroup assumption [34] or the decisional composite residuosity assumption (DCRA) [35]. Note that even with the more efficient UC commitment scheme, this approach to constructing UCZK protocols tends to be fairly inefficient, since a general NP reduction to Hamiltonian Cycle or SAT is used.

[4] In a later version of their paper, Damgård and Nielsen use SAT instead of Hamiltonian Cycle [14].

Our results. We show a new technique that allows us to convert certain types of honest-verifier ZK protocols into ZK protocols with the stronger properties described above, i.e., concurrency, unbounded simulation-soundness, non-malleability, and/or universal composability, in the common reference string model. More precisely, we can

1. transform any Σ-protocol [11] (which are special three-round, honest-verifier protocols where the verifier only sends random bits) into an unbounded simulation-sound ZK protocol; and
2. transform any Ω-protocol (which we introduce in this paper as a variant of Σ-protocols) into a non-malleable ZK protocol, and further into a universally-composable ZK protocol.

The main transformations (sufficient to achieve all results except for UCZK protocols secure in the adaptive model) use a signature scheme that is existentially unforgeable against adaptive chosen-message attacks [25], which exists if one-way functions exist [39], as well as a Σ-protocol to prove knowledge of a signature. Note that one-way functions can be used to construct commitments, and thus if one-way functions exist, Σ-protocols exist for any NP statement (say, through a Cook-Levin reduction, and a standard Σ-protocol for Hamiltonian Cycle). Hence the requirement of our main transformations is the existence of one-way functions. On the other hand, certain signature schemes, such as the Cramer-Shoup [12] scheme and the DSA scheme [30], admit very efficient Σ-protocols. Using these schemes (and at the price of specific number-theoretic assumptions), we are able to construct strengthened ZK protocols that are more efficient than all previously known constructions, since we can completely avoid the Cook-Levin theorem [10, 31]. To further achieve a UCZK protocol that is secure in the adaptive model, we also require a *simulation-sound trapdoor commitment* scheme, a new type of commitment scheme that we introduce and which may be of independent interest. This may be based on trapdoor permutations, but we are able to construct a more efficient version based on DSA.

We now sketch the intuition behind our technique. We first select two signature schemes, the second of which being a one-time signature scheme [20].[5] The common reference string will contain a randomly generated verification key vk for the first signature scheme, and hence neither the prover nor the verifier will know the corresponding signing key. We then take an HVZK protocol Π for an NP statement ϕ, and we modify it to Π^*, which consists of (1) a witness indistinguishable (WI) proof for the statement

> "Either ϕ is true, or I know the signature for the message vk' w.r.t. verification key vk,"

where vk' is a freshly generated verification key for the one-time signature scheme that is also sent to the verifier, and (2) a signature on the transcript of the WI

[5] The second signature scheme may be the same as the first, although for greater efficiency, a signature scheme that is specifically designed for one-time use may be employed.

proof using the secret key corresponding to vk'. We show that this is an unbounded simulation-sound ZK protocol, and we give some efficient instantiations.

Non-malleability is achieved by replacing the Σ-protocol with an Ω-protocol. We then show that an non-malleable ZK protocol can be easily augmented to obtain a universally-composable ZK protocol in the static model. Finally, to achieve a universally-composable ZK protocol in the adaptive model (with erasures), we start with the augmented non-malleable protocol (based on the Ω-protocol), and modify it using a simulation-sound trapdoor commitment scheme,

2 Preliminaries and Definitions

All our results will be in the *common reference string* (CRS) model, which assumes that there is a string uniformly generated from some distribution and is available to all parties at the start of a protocol. Note that this is a generalization of the *public random string* model, where a uniform distribution over fixed-length bit strings is assumed.

For a distribution Δ, we say $a \in \Delta$ to denote any element that has non-zero probability in Δ, i.e., any element in the support of Δ. We say $a \stackrel{R}{\leftarrow} \Delta$ to denote a is randomly chosen according to distribution Δ. For a set S, we say $a \stackrel{R}{\leftarrow} S$ to denote that a is uniformly drawn from S.

We will use signatures schemes that are existentially unforgeable against adaptive chosen-message attacks [26]. However, some of these may only be used for a single signature, and for these, more efficient *one-time* signature scheme constructions may be used [20].

2.1 Zero-knowledge proofs and proofs of knowledge

Here we provide definitions related to zero-knowledge proofs and proofs of knowledge. They are based on definitions of NIZK proofs from [15], but modified to allow interaction.

For a relation R, let $L_R = \{x : (x,w) \in R\}$ be the *language* defined by the relation. For any NP language L, note that there is a natural *witness relation* R containing pairs (x,w) where w is the witness for the membership of x in L, and that $L_R = L$. We will use k as the security parameter.

For two interactive machines A and B, we define $\langle A, B\rangle_{[\sigma]}(x)$ as the local output of B after an interactive execution with A using CRS σ, and common input x. The transcript of a machine is simply the messages on its input and output communication tapes. Two transcripts *match* if the ordered input messages of one are equivalent to the ordered output messages of the other, and vice-versa. We use the notation $tr \bowtie tr'$ to indicate tr matches tr'.

For some definitions below, we need to define security when an adversary is allowed to interact with more than one instance of a machine. Therefore it will be convenient to define a common *wrapper* machine that handles this "multi-session" type of interaction.[6] For an interactive machine A, we define $\boxed{A}$ to be

[6] This is similar to the "multi-session extension" concept in Canetti and Rabin [9].

a protocol wrapper for A, that takes two types of inputs on its communication tape:

- $(\textsc{start}, \pi, x, w)$: For this message $\boxed{A}$ starts a new interactive machine A with label π, common input x, private input w, a freshly generated random input r, and using the CRS of $\boxed{A}$.
- $(\textsc{msg}, \pi, m)$: For this message $\boxed{A}$ sends the message m to the interactive machine with label π (if it exists), and returns the output message of that machine.

We define the output of $\boxed{A}$ to be a tuple (x, tr, v), where x is the common input (from the START message), tr is the transcript (the input and output messages A) and v is the output of A. (In particular, if A is a verifier in a zero-knowledge protocol, this output will be 1 for accept, and 0 for reject.) We say $\boxed{A}_1$ is the wrapper of A that ignores all the subsequent START messages after seeing the first one. Effectively, $\boxed{A}_1$ is a "single-session" version of A.

We say two interactive machines B and C are *coordinated* if they have a single control, but two distinct sets of input/output communication tapes. For four interactive machines A, B, C, and D we define $(\langle A, B\rangle, \langle C, D\rangle)_{[\sigma]}$ as the local output of D after an interactive execution with C and after an interactive execution of A and B, all using CRS σ. Note that we will only be concerned with this if B and C are coordinated.

We note that all our ZK definitions use black-box, non-rewinding simulators, and our proofs of knowledge use non-rewinding extractors.

Definition 1. [Unbounded ZK Proof] $\Pi = (\mathcal{D}, \mathcal{P}, \mathcal{V}, \mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2))$ *is an* unbounded ZK proof *(resp.,* argument*)* system *for an NP language L with witness relation R if $\mathcal{D}$ is an ensemble of polynomial-time samplable distributions, $\mathcal{P}$, $\mathcal{V}$, and $\mathcal{S}_2$ are probabilistic polynomial-time interactive machines, and $\mathcal{S}_1$ is a probabilistic polynomial-time machine, such that there exist negligible functions α and β (the simulation error), such that for all k,*

Completeness *For all $x \in L$ of length k, all w such that $R(x, w) = 1$, and all $\sigma \in \mathcal{D}_k$ the probability that $\langle \mathcal{P}(w), \mathcal{V}\rangle_{[\sigma]}(x) = 0$ is less than $\alpha(k)$.*

Soundness *For all unbounded (resp., polynomial-time) adversaries $\mathcal{A}$, if $\sigma \xleftarrow{R} \mathcal{D}_k$, then for all $x \notin L$, the probability that $\langle \mathcal{A}, \mathcal{V}\rangle_{[\sigma]}(x) = 1$ is less than $\alpha(k)$.*

Unbounded ZK *For all non-uniform probabilistic polynomial-time interactive machines $\mathcal{A}$, we have that $|\Pr[\mathsf{Expt}_{\mathcal{A}}(k) = 1] - \Pr[\mathsf{Expt}^{\mathcal{S}}_{\mathcal{A}}(k) = 1]| \le \beta(k)$, where the experiments $\mathsf{Expt}_{\mathcal{A}}(k)$ and $\mathsf{Expt}^{\mathcal{S}}_{\mathcal{A}}(k)$ are defined as follows:*

$\mathsf{Expt}_{\mathcal{A}}(\kappa)$:	$\mathsf{Expt}^{\mathcal{S}}_{\mathcal{A}}(\kappa)$:
$\sigma \xleftarrow{R} \mathcal{D}_k$	$(\sigma, \tau) \leftarrow \mathcal{S}_1(1^k)$
Return $\langle \boxed{\mathcal{P}}, \mathcal{A}\rangle_{[\sigma]}$	Return $\langle \boxed{\mathcal{S}'(\tau)}, \mathcal{A}\rangle_{[\sigma]}$

where $\mathcal{S}'(\tau)$ runs as follows on common reference string σ, common input x and private input w: if $R(x, w) = 1$, $\mathcal{S}'(\tau)$ runs $\mathcal{S}_2(\tau)$ on common reference *string σ and common input x; otherwise $\mathcal{S}'(\tau)$ runs $\mathcal{S}_{\text{null}}$, where $\mathcal{S}_{\text{null}}$ is an interactive machine that simply aborts.*[7]

We point out that this definition only requires the simulator to simulate a valid proof, which is implemented by having $\mathcal{S}'$ have access to the witness w and only invoking $\mathcal{S}_2$ when w is valid.[8] However, $\mathcal{S}_2$ does not access the witness and will simulate a proof from the input x only.

Definition 2. [Same-String Unbounded ZK] $\Pi = (\mathcal{D}, \mathcal{P}, \mathcal{V}, \mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2))$ *is a* same-string unbounded ZK argument *system for an NP language L with witness relation R if Π is an unbounded ZK argument system for L with the additional property that the distribution of the reference string output by $\mathcal{S}_1(1^k)$ is exactly $\mathcal{D}_k$.*

We only define same-string unbounded ZK arguments since, as shown in [15], any protocol that is same-string unbounded ZK must be an argument, and not a proof.

The following defines unbounded simulation-sound zero-knowledge (USSZK). This has been useful in applications. In particular, as shown in [40], the one-time version suffices for the security of a (non-interactive) ZK protocol in the construction of adaptive chosen-ciphertext secure cryptosystems using the Naor-Yung [33] paradigm. We directly define the unbounded version, needed in other applications such as threshold password-authenticated key exchange [32].

Definition 3. [Unbounded Simulation-Sound ZK]
$\Pi = (\mathcal{D}, \mathcal{P}, \mathcal{V}, \mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2))$ *is an* unbounded simulation-sound ZK proof *(resp.,* argument*)* system *for an NP language L if Π is an unbounded ZK proof (resp., argument) system for L and furthermore, there exists a negligible function α such that for all k,*

Unbounded Simulation Soundness

For all non-uniform probabilistic polynomial-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, where $\mathcal{A}_1$ and $\mathcal{A}_2$ are coordinated, we have that $\Pr[\mathsf{Expt}_{\mathcal{A}}(k) = 1] \le \alpha(k)$, where $\mathsf{Expt}_{\mathcal{A}}(k)$ is defined as follows:

$$\boxed{\begin{array}{l} \mathsf{Expt}_{\mathcal{A}}(k): \\ \quad (\sigma, \tau) \leftarrow \mathcal{S}_1(1^k) \\ \quad (x, tr, b) \leftarrow (\langle \boxed{\mathcal{S}''(\tau)}, \mathcal{A}_1 \rangle, \langle \mathcal{A}_2, \boxed{\mathcal{V}}_1 \rangle)_{[\sigma]} \\ \quad \text{Let } Q \text{ be the set of transcripts of machines in } \boxed{\mathcal{S}''(\tau)} \\ \quad \text{Return 1 iff } b = 1,\ x \notin L, \text{ and for all } tr' \in Q,\ tr \not\bowtie tr' \end{array}}$$

where $\mathcal{S}''(\tau)$ runs as follows on CRS σ, common input x and private input w: $\mathcal{S}''(\tau)$ runs $\mathcal{S}_2(\tau)$ on CRS σ and common input x.

---

[7] Without loss of generality, we assume that if the input to $\mathcal{P}$ is not a witness for the common input, $\mathcal{P}$ simply aborts.

[8] $\mathcal{A}$ must supply a witness, since $\mathcal{P}$ is restricted to polynomial time, and thus may not be able to generate a witness itself. This may seem odd compared to definitions of standard ZK that assume an unbounded prover, but it does seem to capture the correct notion of unbounded ZK, and in particular does not allow $\mathcal{A}$ to test membership in L. See Sahai [40] for more discussion.

In the above definition, we emphasize that $\mathcal{S}_2$ may be asked to simulate *false* proofs for $x \notin L_R$, since $\mathcal{S}''$ does not check whether $(x, w) \in R$. The idea is that even if the adversary is able to obtain acceptable proofs on false statements, it will not be able to produce any new acceptable proof on a false statement.

The following defines non-malleable zero-knowledge (NMZK) proofs (resp., arguments) of knowledge. If a protocol is NMZK according to our definition, then this implies the protocol is also a NMZK in the explicit witness sense (as defined in [15]). Moreover, we show that the protocol is also UCZK in the model of static corruptions. Also note that simulation soundness is implied by this definition.

Definition 4. **[Non-malleable ZK Proof/Argument of Knowledge]** $\Pi = (\mathcal{D}, \mathcal{P}, \mathcal{V}, \mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2), \mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2))$ *is a* non-malleable ZK proof *(resp.,* argument*)* of knowledge system *for an NP language L with witness relation R if Π is an unbounded ZK proof (resp., argument) system for L and furthermore, $\mathcal{E}_1$ and $\mathcal{E}_2$ are probabilistic polynomial-time machines such that there exists a negligible function α (the knowledge error) such that for all k,*

Reference String Indistinguishability *The distribution of the first output of $\mathcal{S}_1(1^k)$ is identical to the distribution of the first output of $\mathcal{E}_1(1^k)$.*

Extractor Indistinguishability *For any $\tau \in \{0,1\}^*$, the distribution of the output of $\boxed{\mathcal{V}}_1$ is identical to the distribution of the restricted output of $\boxed{\mathcal{E}_2(\tau)}_1$, where the restricted output of $\boxed{\mathcal{E}_2(\tau)}_1$ does not include the extracted value.*

Extraction *For all non-uniform probabilistic polynomial-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, where $\mathcal{A}_1$ and $\mathcal{A}_2$ are coordinated machines, we have that $|\Pr[\mathsf{Expt}^{\mathcal{E}}_{\mathcal{A}}(k) = 1] - \Pr[\mathsf{Expt}_{\mathcal{A}}(k) = 1]| \le \alpha(k)$, where the experiments $\mathsf{Expt}_{\mathcal{A}}(k)$ and $\mathsf{Expt}^{\mathcal{E}}_{\mathcal{A}}(k)$ are defined as follows:*

$\mathsf{Expt}_{\mathcal{A}}(k)$:	$\mathsf{Expt}^{\mathcal{E}}_{\mathcal{A}}(k)$:
$(\sigma, \tau) \leftarrow \mathcal{S}_1(1^k)$	$(\sigma, \tau_1, \tau_2) \leftarrow \mathcal{E}_1(1^k)$
(x, tr, b)	$(x, tr, (b, w))$
$\leftarrow (\langle \boxed{\mathcal{S}''(\tau)}, \mathcal{A}_1 \rangle, \langle \mathcal{A}_2, \boxed{\mathcal{V}}_1 \rangle)[\sigma]$	$\leftarrow (\langle \boxed{\mathcal{S}''(\tau_1)}, \mathcal{A}_1 \rangle, \langle \mathcal{A}_2, \boxed{\mathcal{E}_2(\tau_2)}_1 \rangle)[\sigma]$
Let Q be the set of transcripts of machines in $\boxed{\mathcal{S}''(\tau)}$.	Let Q be the set of transcripts of machines in $\boxed{\mathcal{S}''(\tau_1)}$.
Return 1 iff $b = 1$ and for all $tr' \in Q$, $tr \not\bowtie tr'$	Return 1 iff $b = 1$, $(x, w) \in R$, and for all $tr' \in Q$, $tr \not\bowtie tr'$

where $\mathcal{S}''(\tau)$ runs as follows on CRS σ, common input x and private input w: $\mathcal{S}''(\tau)$ runs $\mathcal{S}_2(\tau)$ on CRS σ and common input x.

In the above definition, as in the definition of USSZK protocols, we emphasize that $\mathcal{S}_2$ may be asked to simulate *false* proofs for $x \notin L_R$, since $\mathcal{S}''$ does not check whether $(x, w) \in R$. The idea is that even if the adversary is able to obtain acceptable proofs on false statements, it will not be able to produce any new acceptable proof for which a witness cannot be extracted.

2.2 Σ-protocols

Here we overview the basic definitions and properties of Σ-protocols [11]

First we start with some definitions and notation. Let $R = \{(x, w)\}$ be a binary relation and assume that for some given polynomial $p(\cdot)$ it holds that $|w| \leq p(|x|)$ for all $(x, w) \in R$. Furthermore, let R be testable in polynomial time. Let $L_R = \{x : (x, w) \in R\}$ be the *language* defined by the relation.

Now we define a Σ-protocol (A, B) to be a three move interactive protocol between a probabilistic polynomial-time prover A and a probabilistic polynomial-time verifier B, where the prover acts first. The verifier is only required to send random bits as a challenge to the prover. For some $(x, w) \in R$, the common input to both players is x while w is private input to the prover. For such given x, let (a, c, z) denote the conversation between the prover and the verifier. To compute the first and final messages, the prover invokes efficient algorithms $a(\cdot)$ and $z(\cdot)$, respectively, using (x, w) and random bits as input. Using an efficient predicate $\phi(\cdot)$, the verifier decides whether the conversation is accepting with respect to x. The relation R, the algorithms $a(\cdot)$, $z(\cdot)$ and $\phi(\cdot)$ are public.

We will need to broaden this definition slightly, to deal with cheating provers. We will define $\hat{L}_R$ to be the input language, with the property that $L_R \subseteq \hat{L}_R$, and membership in $\hat{L}_R$ may be tested in polynomial time. We implicitly assume B only executes the protocol if the common input $x \in \hat{L}_R$.

All Σ-protocols presented here will satisfy the following security properties:

- *Weak special soundness*: Let (a, c, z) and (a, c', z') be two conversations, that are accepting for some given $x \in \hat{L}_R$. If $c \neq c'$, then $x \in L_R$. The pair of accepting conversations (a, c, z) and (a, c', z') with $c \neq c'$ is called a *collision*.
- *Special honest verifier zero knowledge* (SHVZK): There is a (probabilistic polynomial time) simulator M that on input $x \in L_R$ generates accepting conversations with a distribution that is computationally indistinguishable from when A and B faithfully execute the protocol on common input x. The simulator is special in the sense that it can additionally take a random string c as input, and output an accepting conversation for x where c is the challenge. In fact, we will assume the simulator has this special property for not only $x \in L_R$, but also any $x \in \hat{L}_R$.

Some of the Σ-protocols also satisfy the following property.

- *Special soundness*: Let (a, c, z) and (a, c', z') be two conversations, that are accepting for some given x, with $c \neq c'$. Then given x and those two conversations, a witness w such that $(x, w) \in R$ can be computed efficiently.

A simple but important fact (see [11]) is that if a Σ-protocol is HVZK, the protocol is *witness indistinguishable* (WI) [21].

In our results to follow, we need a particular, simple instance of the main theorem from [11]. Specifically, we use a slight generalization of a corollary in [11] which enables a prover, given two relations (R_1, R_2), values $(x_1, x_2) \in \hat{L}_{R_1} \times \hat{L}_{R_2}$, and corresponding 3-move Σ-protocols $((A_1, B_1), (A_2, B_2))$, to present a 3-move

$$
\begin{array}{rcl}
\textsf{prover} & & \textsf{verifier} \\
(vk', sk') \leftarrow \textsf{sig\_gen}_1(1^k) & \xrightarrow{\quad vk' \quad} & \\
 & \overset{\Sigma^R(x) \vee \Sigma^{R_{vk}}(vk')}{\rightleftarrows} & \\
s \leftarrow \textsf{sig\_sign}_1(sk', \text{transcript}) & \xrightarrow{\quad s \quad} & \textsf{sig\_verify}_1(vk', \text{transcript})
\end{array}
$$

Fig. 1. $\mathsf{USS}^R_{[vk]}(x)$: An unbounded simulation-sound ZK protocol for relationship R with CRS vk (drawn from the distribution $\mathsf{sig\_gen}_0(1^k)$), and common input x. The prover also knows the witness w such that $R(x, w) = 1$.

Σ-protocol (A_{or}, B_{or}) for proving the existence of a w such that either $(x_1, w) \in R_1$ or $(x_2, w) \in R_2$. We call this the "OR" protocol for $((A_1, B_1), (A_2, B_2))$,

For two Σ-protocols, (A_1, B_1) and (A_2, B_2), let $(A_1, B_1) \vee (A_2, B_2)$ denote the "OR" protocol for $((A_1, B_1), (A_2, B_2))$.

3 Unbounded Simulation-Sound ZK

We are now ready to present the first result achieved with our technique: An unbounded simulation-sound zero-knowledge protocol for a relation $R = \{(x, w)\}$. We assume that we have the following building blocks:

1. Σ^R: a Σ-protocol for the binary relation R.
2. $\mathsf{SIG}_0 = (\mathsf{sig\_gen}_0, \mathsf{sig\_sign}_0, \mathsf{sig\_verify}_0)$: a signature scheme secure against adaptive chosen-message attack.
3. $R_{vk} = \{(m, s) \mid \mathsf{sig\_verify}_0(vk, m, s) = 1\}$: a binary relation of message-signature pairs.
4. $\Sigma^{R_{vk}}$: a Σ-protocol with the special soundness property for the binary relation R_{vk}.
5. $\mathsf{SIG}_1 = (\mathsf{sig\_gen}_1, \mathsf{sig\_sign}_1, \mathsf{sig\_verify}_1)$: a one-time signature scheme secure against chosen-message attack.

The protocol $\mathsf{USS}^R_{[vk]}(x)$ is shown in Figure 1. It assumes the prover and verifier share a common input x to a Σ-protocol Σ^R, and the prover knows w such that $(x, w) \in R$. The CRS σ is the verification key vk of a signature scheme that is existentially unforgeable against adaptive chosen-message attacks. The prover generates a pair (vk', sk') for a one-time signature scheme, and sends vk' to the verifier. After this, vk' is the common input to a Σ-protocol $\Sigma^{R_{vk}}$ satisfying special soundness. Then the prover uses the OR construction for Σ-protocols to prove that either $x \in L_R$ or it knows a signature for vk' under verification key vk. (Note that since $\Sigma^{R_{vk}}$ satisfies special soundness, intuitively it is a proof of knowledge.) Finally, the prover signs the transcript with sk', and sends the resulting signature to the verifier.

Now we must describe $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ for $\mathsf{USS}^R_{[vk]}(x)$. $\mathcal{S}_1(1^k)$ first generates signature keys $(vk, sk) \leftarrow \mathsf{sig\_gen}_0(1^k)$ and outputs $(\sigma, \tau) = (vk, sk)$. $\mathcal{S}_2(sk)$ first checks that common input $x \in \hat{L}_R$. If not, it aborts. Otherwise it runs the protocol as normal, except generating $s' \leftarrow \mathsf{sig\_sign}_0(sk, vk')$, and using knowledge of s' to complete the Σ-protocol $\Sigma^R(x) \vee \Sigma^{R_{vk}}(vk')$.

Theorem 1. *The protocol* $\mathsf{USS}^R_{[vk]}(x)$ *is a USSZK argument.*

4 Non-malleable ZK

Our general NMZK construction will be similar to the USSZK construction above, but with a Σ-protocol replaced by an *Ω-protocol*, defined here.

4.1 Ω-protocols

An Ω-protocol $(A, B)_{[\sigma]}$ for a relation $R = \{(x, w)\}$ and CRS σ, is a Σ-protocol for relation R with the following additional properties.

1. For a given distribution ensemble $\mathcal{D}$, a common reference string σ is drawn from $\mathcal{D}_k$ and each function $a(\cdot)$, $z(\cdot)$, and $\phi(\cdot)$ takes σ as an additional input. (Naturally, the simulator M in the definition of Σ-protocols may also take σ as an additional input.)
2. There exists a polynomial-time extractor $\mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2)$ such that the reference string output by $\mathcal{E}_1(1^k)$ is statistically indistinguishable from $\mathcal{D}_k$. Furthermore, given $(\sigma, \tau) \leftarrow \mathcal{E}_1(1^k)$, if there exists two accepting conversations (a, c, z) and (a, c', z') with $c \neq c'$ for some given $x \in \hat{L}_R$, then $\mathcal{E}_2(x, \tau, (a, c, z))$ outputs w such that $(x, w) \in R$.[9]

Informally, one way to construct Ω-protocols is as follows. Our common reference string will consist of a random public key pk for a semantically-secure encryption scheme. Then for a given $(x, w) \in R$, we will construct an encryption e of w under key pk, and then construct a Σ-protocol to prove that there is a w such that $(x, w) \in R$ and that e is an encryption of w.

As with Σ-protocols, we will use the $\vee$ notation to denote an "OR" protocol, even if one or both of these protocols are Ω-protocols.

4.2 NMZK protocol

Let $\Omega^R_{[\sigma']}(x)$ be an Ω-protocol for a relation R with common reference string σ' and common input x. Let $\mathsf{NM}^R_{[vk,\sigma']}(x)$ be the $\mathsf{USS}^R_{[vk]}(x)$ protocol with $\Sigma^R(x)$

[9] Notice that this extraction property is similar to that of weak special soundness of Σ-protocols, where there exists an accepting conversation even for an invalid proof, but two accepting conversations guarantees that the proof is valid. Here, the extractor can always extract something from any conversation, but it might not be the witness if there is only one accepting conversation. However, having two accepting conversations sharing the same a guarantees that the extracted information is indeed a witness.

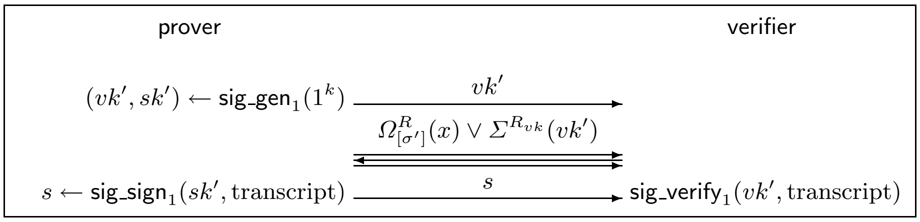

Fig. 2. $\mathsf{NM}^R_{[vk,\sigma']}(x)$: A non-malleable ZK protocol for relationship R with common reference string (vk, σ') where σ' is drawn from the distribution associated with $\Omega^R_{\sigma'}$, and common input x.

replaced by $\Omega^R_{[\sigma']}(x)$. (For every σ', the resultant protocol is also a Σ-protocol.) Let $\mathcal{E}_\Omega = (\mathcal{E}_{\Omega,1}, \mathcal{E}_{\Omega,2})$ be the extractor for $\Omega^R_{[\sigma']}(x)$. The protocol $\mathsf{NM}^R_{[vk,\sigma']}(x)$ is shown in Figure 2.

We now describe $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ for $\mathsf{NM}^R_{[vk,\sigma']}$. $\mathcal{S}_1(1^k)$ first generates signature keys $(vk, sk) \leftarrow \mathsf{sig\_gen}_0(1^k)$ and then sets $\sigma' \stackrel{R}{\leftarrow} \mathcal{D}_k$, where $\mathcal{D}$ is the distribution ensemble for $\Omega^R_{[\sigma']}$. Next, $\mathcal{S}_1(1^k)$ outputs $((vk, \sigma'), sk)$. $\mathcal{S}_2(sk)$ first checks that common input $x \in \hat{L}_R$. If not, it aborts. Otherwise it runs the protocol as normal, except generating $s' \leftarrow \mathsf{sig\_sign}_0(sk, vk')$, and using knowledge of s' to complete the protocol $\Omega^R_{[\sigma']}(x) \vee \Sigma^{R_{vk}}(vk')$.

Finally, we must describe $\mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2)$ for $\mathsf{NM}^R_{[vk,\sigma']}(x)$. $\mathcal{E}_1(1^k)$ generates signatures keys $(vk, sk) \leftarrow \mathsf{sig\_gen}_0(1^k)$, generates $(\sigma', \tau') \leftarrow \mathcal{E}_{\Omega,1}(1^k)$, and then outputs $((vk, \sigma'), sk, \tau')$. $\mathcal{E}_2(\tau')$ simply runs as $\mathcal{V}$ until $\mathcal{V}$ outputs a bit b. If $b = 1$, $\mathcal{E}_2(\tau')$ takes the conversation (a, c, z) produced by $\Omega^R_{[\sigma']}(x)$, and generates $w \leftarrow \mathcal{E}_{\Omega,2}(x, \tau', (a, c, z))$. If $b = 0$, $\mathcal{E}_2(\tau')$ sets $w \leftarrow \perp$. Then $\mathcal{E}_2(\tau')$ outputs (b, w).

Theorem 2. *The protocol* $\mathsf{NM}^R_{[vk,\sigma']}(x)$ *is an NMZK argument of knowledge for the relation* R.

5 Universally Composable ZK

First we review the framework of universal composability [5]. Then we prove that any NMZK protocol with certain simple properties can be augmented to be UCZK in the model of static corruptions. This result implies as a corollary that a slight generalization of our protocol from the previous section can be augmented to be UCZK in this model. Then we give a new construction that is UCZK in the model of adaptive corruptions.

5.1 The universal composability framework

The universal composability paradigm was proposed by Canetti [5] for defining the security and composition of protocols. To define security one first specifies an *ideal functionality* using a trusted party that describes the desired behavior of the protocol. Then one proves that a particular protocol operating in a real-life model securely realizes this ideal functionality, as defined below. Here we briefly summarize the framework as defined in Canetti [5].

A (real-life) protocol π is defined as a set of n interactive Turing Machines $P_1, \ldots, P_n$, designating the n parties in the protocol. It operates in the presence of an environment $\mathcal{Z}$ and an adversary $\mathcal{A}$, both of which are also modeled as interactive Turing Machines. The environment $\mathcal{Z}$ provides inputs and receives outputs from honest parties, and may communicate with $\mathcal{A}$. $\mathcal{A}$ controls (and may view) all communication between the parties. We will assume that messages are authenticated, and thus $\mathcal{A}$ may not insert or modify messages between honest parties.[10] $\mathcal{A}$ also may corrupt parties, in which case it obtains the internal state of the party.

The ideal process with respect to a functionality $\mathcal{F}$, is defined for n parties $P_1, \ldots, P_n$, an environment $\mathcal{Z}$, and an (ideal-process) adversary $\mathcal{S}$. However, $P_1, \ldots, P_n$ are now dummy parties that simply forward (over secure channels) inputs received from $\mathcal{Z}$ to $\mathcal{F}$, and forward (again over secure channels) outputs received from $\mathcal{F}$ to $\mathcal{Z}$. Thus the ideal process is a trivially secure protocol with the input-output behavior of $\mathcal{F}$.

To formulate the universal composition theorem, Canetti [5] also introduces a a hybrid model, a real-life model with access to an ideal functionality $\mathcal{F}$. In particular, this $\mathcal{F}$-hybrid model functions like the real-life model, but where the parties may also exchange messages with an unbounded number of copies of $\mathcal{F}$, each copy identified via a unique *session identifier* (*sid*). The communication between the parties and each one of these copies mimics the ideal process, and in particular the hybrid adversary does not have access to the contents of the messages. See Canetti [5] for details of the universal composition theorem.

The zero-knowledge functionality. The (multi-session) ZK functionality is given in Figure 3. In the functionality, parameterized by a relation R, the prover sends to the functionality the input x together with a witness w. If $R(x, w)$ holds, then the functionality forwards x to the verifier. As pointed out in [5], this is actually a proof of knowledge in that the verifier is assured that the prover actually knows w. Note the two types of indices: the *sid*, which, as before, differentiates messages to $\hat{\mathcal{F}}_{\mathrm{ZK}}^{R}$ from messages sent to other functionalities, and *ssid*, the sub-session ID, which is unique per input message (or proof).

Recall that we will be designing and analyzing protocols in the common reference string model, and so they will be operating in the $\mathcal{F}_{\mathrm{CRS}}^{\mathcal{D}}$-hybrid model, where $\mathcal{F}_{\mathrm{CRS}}^{\mathcal{D}}$ is the functionality that, for a given security parameter k, chooses a string from distribution $\mathcal{D}_k$ and hands it to all parties.

[10] This feature could be added to an unauthenticated model using a message authentication functionality as described in [5].

Functionality $\hat{\mathcal{F}}_{\mathrm{ZK}}^{R}$

$\hat{\mathcal{F}}_{\mathrm{ZK}}^{R}$ proceeds as follows, running with security parameter k, parties $P_1, \ldots, P_n$, and an adversary $\mathcal{S}$:

– Upon receiving $(\mathsf{zk\text{-}prover}, sid, ssid, P_i, P_j, x, w)$ from P_i: If $R(x, w)$ then send $(\mathsf{ZK\text{-}PROOF}, sid, ssid, P_i, P_j, x)$ to P_j and $\mathcal{S}$ and halt. Otherwise, ignore.

Fig. 3. *The multi-session zero-knowledge functionality (for relation R)*

5.2 NMZK implies UCZK

Let Π be an NMZK protocol between a prover and verifier. We say Π is *augmentable* if the prover sends the first message, and this message contains the common input x, along with auxiliary data aux that may contain any arbitrary public values. (The reason for aux is discussed below.) We will show how to augment Π with additional information in each message to allow it to be used between two parties in the universal composability framework. This augmented protocol is denoted $\hat{\Pi}$, and is constructed as follows.

For an instance of $\hat{\Pi}$ run between parties P_i and P_j, set aux to $(ssid, P_i, P_j)$, where $ssid$ is defined in the previous section, P_i is the identity of the prover, and P_j is the identity of the verifier.[11] Then the ℓth prover message is formatted as $(\mathsf{prv}_\ell, sid, ssid, P_i, \mathsf{prv\text{-}data}_\ell)$, where prv_ℓ is the label for the ℓth prover message, and $\mathsf{prv\text{-}data}_\ell$ is the data field containing the ℓth message sent by the prover in Π. Analogously, the ℓth verifier message is formatted as $(\mathsf{ver}_\ell, sid, ssid, P_j, \mathsf{ver\text{-}data}_\ell)$, where ver_ℓ is the label for the ℓth verifier message, and $\mathsf{ver\text{-}data}_\ell$ is the data field containing the ℓth message sent by the verifier in Π. Finally, before accepting, the verifier checks that aux corresponds to the values $(ssid, P_i, P_j)$ outside the prover data field, and that aux was not used previously.

Theorem 3. *Let $\Pi = (\mathcal{D}, \mathcal{P}, \mathcal{V}, \mathcal{S}_\Pi = (\mathcal{S}_{\Pi,1}, \mathcal{S}_{\Pi,2}), \mathcal{E}_\Pi = (\mathcal{E}_{\Pi,1}, \mathcal{E}_{\Pi,2}))$ be an augmentable NMZK protocol for a relation R. Then the augmented protocol $\hat{\Pi}$ securely realizes functionality $\hat{\mathcal{F}}_{\mathrm{ZK}}^{R}$ in the $\mathcal{F}_{\mathrm{CRS}}^{\mathcal{D}}$-hybrid model, assuming static corruptions.*

We say a protocol $\hat{\Pi}$ is a *UCZK protocol* for R if it securely realizes functionality $\hat{\mathcal{F}}_{\mathrm{ZK}}^{R}$ in the $\mathcal{F}_{\mathrm{CRS}}^{\mathcal{D}}$-hybrid model, for some $\mathcal{D}$.

Corollary 1. *Let Π be protocol $\mathsf{NM}^{R}_{[vk,\sigma']}(x)$ from Figure 2 with the addition of the common input x and $\mathsf{aux} = (ssid, P_i, P_j)$ in the first message. Then the augmented protocol $\hat{\Pi}$ is a UCZK protocol for R, assuming static corruptions.*

[11] This auxiliary data aux is necessary since NMZK allows copying proofs exactly, but the ZK functionality does not, and thus we need some way to make every proof distinct.

5.3 UCZK: Adaptive corruptions

Our basic idea to deal with adaptive corruptions is to take the augmentable version of the NMZK protocol from Corollary 1, denoted $\mathsf{NM}^{R}_{[vk,\sigma']}(x;\mathsf{aux})$, and apply to it the technique proposed by Damgård [13] and Jarecki and Lysyanskaya [27] in which a *trapdoor commitment* is used to commit to the first message of a Σ-protocol, and then this commitment is opened when sending the third message. Informally, a trapdoor commitment is a commitment scheme with the additional property that there is a secret trapdoor such that knowing the trapdoor allows a committer to decommit to an arbitrary value. More precisely, $\mathsf{TC} = (\mathsf{TCgen}, \mathsf{TCcom}, \mathsf{TCver}, \mathsf{TCkeyver}, \mathsf{TCfake})$ is a trapdoor commitment scheme if it satisfies the properties of completeness, binding, perfect secrecy, and trapdoorness. The first three properties are the same as in any unconditionally-hiding commitment scheme. The trapdoor property says (informally) that $\mathsf{TCgen}(1^k)$ outputs a secret key (the trapdoor) along with the public key, and that using this secret key and a commitment/decommitment pair (c, d) associated with a value v, (i.e., $(c, d) \leftarrow \mathsf{TCcom}(pk, v)$), the function TCfake can for any value v' output a decommitment d' that is a valid decommitment of c resulting in v' (i.e., $\mathsf{TCver}(pk, c, v', d') = 1$).

However, for technical reasons, a "plain" trapdoor commitment does not provide the properties we need to deal with adaptive corruptions, and so we define a stronger type of trapdoor commitment scheme, which we call a *simulation-sound trapdoor commitment* (SSTC) scheme.[12] Roughly speaking, an SSTC scheme is a trapdoor commitment scheme with an extra input *id* to the commitment protocol, which guarantees that a commitment made by the adversary using input *id* is binding, even if the adversary has seen any commitment using input *id* opened (using a simulator that knows a trapdoor) once to any arbitrary value, and moreover, any commitment using $id' \neq id$ opened (again using the simulator) an unbounded number of times to any arbitrary values.

Now let Π be a three move interactive proof protocol with common input x, auxiliary input aux, witness w, common reference string σ, and prover random bits r. Similarly to Σ-protocols, we use the notation $a_{\Pi}(\cdot)$, $z_{\Pi}(\cdot)$, and $\mathsf{verify}_{\Pi}(\cdot)$ to denote the algorithms for computing the two messages of the prover, and verifying the proof, respectively. Using this notation, the protocol $\mathsf{UC}^{R}_{[pk^*,vk,\sigma']}(x;\mathsf{aux})$ is shown in Figure 4.

Theorem 4. *Let Π' be the protocol $\mathsf{UC}^{R}_{[pk^*,vk,\sigma']}(x;\mathsf{aux})$, where $\mathsf{aux} = (ssid, P_i, P_j)$. Then the augmented protocol $\hat{\Pi}'$ securely realizes functionality $\hat{\mathcal{F}}^{R}_{\mathrm{ZK}}$ in the $\mathcal{F}^{\mathcal{D}}_{\mathrm{CRS}}$-hybrid model where erasing is allowed, assuming adaptive corruptions.*

[12] Universally-composable commitments [6, 8] would also suffice, and can be constructed using trapdoor permutations. However, this construction is not as efficient as the SSTC scheme in this paper.

prover		verifier
$(x, \mathsf{aux}, a) \leftarrow a_\Pi(x, \mathsf{aux}, w, r, \sigma)$		
$(a^*, r^*) \leftarrow \mathsf{TCcom}(pk^*, a, \mathsf{aux})$	$\xrightarrow{x,\, \mathsf{aux},\, a^*}$	
		$c \stackrel{R}{\leftarrow} \{0,1\}^k$
	$\xleftarrow{c}$	
$z \leftarrow z_\Pi(x, \mathsf{aux}, w, r, c, \sigma)$		
$\mathsf{erase}(r, w)$	$\xrightarrow{z,\, a,\, r^*}$	$\mathsf{TCkeyver}(pk^*, 1^k)$
		$\mathsf{TCver}(pk^*, a^*, a, \mathsf{aux}, r^*)$
		$\mathsf{verify}_\Pi(x, \mathsf{aux}, a, c, z, \sigma)$

Fig. 4. $\mathsf{UC}^R_{[pk^*, vk, \sigma']}(x; \mathsf{aux})$: A UCZK protocol for R with common reference string (pk^*, vk, σ') drawn from $\mathcal{D}_{pk}(\mathsf{TC}) \times \mathcal{D}_{vk}(\mathsf{SIG}_0) \times \mathcal{D}_\sigma(\Omega^R)$, common input x, and auxiliary input aux where $\Pi = \mathsf{NM}^R_{[vk, \sigma']}(x; \mathsf{aux})$:

6 Efficient Instantiations

Here we briefly discuss some efficient instantiations of our constructions.

Signature schemes First we note that for our constructions we can use a more general version of the Σ-protocol for proving knowledge of signatures, as follows. Consider the binary relation $R_{vk} = \{(m, s)\}$ for a signature scheme SIG. We say a polynomial-time computable function f is a *partial knowledge function* of SIG, if there exists a probabilistic polynomial-time machine M such that every m and vk, $\{s_1 : s_1 \leftarrow M(m, vk)\}$ and $\{s_1 : s \leftarrow \mathsf{sig\_sign}(vk, m); s_1 \leftarrow f(m, vk, s)\}$ have the same distribution. Intuitively, a partial knowledge function carries part of the information about the signature, yet can be efficiently sampled without even knowing one. If a signature scheme SIG has a partial knowledge function f, then the relation $R'_{vk} = \{((m, s_1), s) : (m, s) \in R_{vk} \wedge s_1 = f(m, vk, s)\}$ can replace R_{vk} in the constructions for $\mathsf{USS}^R_{[vk]}$, $\mathsf{NM}^R_{[vk,\sigma']}(x)$, and $\mathsf{UC}^R_{[pk^*, vk, \sigma']}(x)$, with $\mathcal{P}$ sending a randomly sampled s_1 (partial knowledge) before running the Σ-protocol $\Sigma^R(x) \vee \Sigma^{R'_{vk}}(vk', s_1)$. We say R'_{vk} is a *partial signature relation* for SIG.

It can be shown that the Cramer-Shoup signature scheme [12] and the DSA signature scheme [30] both admit efficient Σ-protocols for proving knowledge of signatures using this more general definition, and thus can be plugged into our constructions. We discuss the detailed constructions in the full version.

Efficient Ω-protocols In the full version we describe an efficient Ω-protocol for proving knowledge of a discrete logarithm. This protocol is based on the Decisional Composite Residuosity assumption and the Strong RSA assumption. In the full version we also describe a generalized version of Ω-protocols, and an efficient generalized Ω-protocol for proving plaintext knowledge of ElGamal encryptions.

References

1. B. Barak. How to Go Beyond the Black-box Simulation Barrier. In *42nd IEEE Symp. on Foundations of Computer Sci.*, 106–115, 2001.
2. B. Barak. Constant-Round Coin-Tossing With a Man in the Middle or Realizing the Shared Random String Model. In *43rd IEEE Symp. on Foundations of Computer Sci.*, 345–355, 2002
3. N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology – EUROCRYPT '97* (LNCS 1233), 480–494, 1997.
4. D. Boneh. The decision Diffie-Hellman problem. In *Proceedings of the Third Algorithmic Number Theory Symp.* (LNCS 1423), 48–63, 1998.
5. R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd IEEE Symp. on Foundations of Computer Sci.*, 136–145, 2001.
6. R. Canetti and M. Fischlin. Universally composable commitments. In *Advances in Cryptology – CRYPTO 2001* (LNCS 2139), 19–40, 2001.
7. R. Canetti, J. Kilian, E. Petrank and A. Rosen. Concurrent zero-knowledge requires $\tilde{\Omega}(\log n)$ rounds. In *33rd ACM Symp. on Theory of Computing*, 570–579, 2001.
8. R. Canetti, Y. Lindell, R. Ostrovsky and A. Sahai. Universally composable two-party computation. In 34th ACM Symp. on Theory of Computing, 494–503, 2002. Full version in *ePrint archive*, Report 2002/140. `http://eprint.iacr.org/`, 2002.
9. R. Canetti and T. Rabin. Universal Composition with Joint State In *ePrint archive*, Report 2002/047, `http://eprint.iacr.org/`, 2002.
10. S. A. Cook. The complexity of theorem-proving procedures. In *3rd IEEE Symp. on Foundations of Computer Sci.*, 151–158, 1971.
11. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology – CRYPTO '94* (LNCS 839), pages 174–187, 1994.
12. R. Cramer and V. Shoup. Signature scheme based on the strong RSA assumption. In *ACM Trans. on Information and System Security* 3(3):161-185, 2000.
13. I. Damgård. Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In *Advances in Cryptology – EUROCRYPT 2000* (LNCS 1807), 418–430, 2000.
14. I. Damgård and J. Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In *Advances in Cryptology – CRYPTO 2002* (LNCS 2442), 581–596, 2002. Full version in *ePrint Archive*, report 2001/091. `http://eprint.iacr.org/`, 2001.
15. A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano and A. Sahai. Robust non-interactive zero knowledge. In *Advances in Cryptology – CRYPTO 2001* (LNCS 2139), 566–598, 2001.
16. D. Dolev, C. Dwork and M. Naor. Non-malleable cryptography. *SIAM J. on Comput.*, 30(2):391–437, 2000. Also in *23rd ACM Symp. on Theory of Computing*, 542–552, 1991.
17. C. Dwork, M. Naor and A. Sahai. Concurrent zero-knowledge. In *30th ACM Symp. on Theory of Computing*, 409–418, 1998.
18. C. Dwork and A. Sahai. Concurrent Zero-Knowledge: Reducing the Need for Timing Constraints. In *Advances in Cryptology – CRYPTO '98* (LNCS 1462), 442–457, 1998.
19. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. on Information Theory*, 31:469–472, 1985.

20. S. Even, O. Goldreich, and S. Micali. On-line/Off-line digital signatures. *J. Cryptology* 9(1):35-67 (1996).
21. U. Feige and A. Shamir. Witness Indistinguishable and Witness Hiding Protocols. In *22nd ACM Symp. on Theory of Computing*, 416–426, 1990.
22. FIPS 186. Digital signature standard. Federal Information Processing Standards Publication 186, U.S. Dept. of Commerce/NIST, National Technical Information Service, Springfield, Virginia, 1994.
23. O. Goldreich, S. Micali and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *19th ACM Symp. on Theory of Computing*, 218–229, 1987.
24. O. Goldreich, S. Micali and A. Wigderson. Proofs that yield nothing but their validity or All languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.
25. S. Goldwasser, S. Micali and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, February 1989.
26. S. Goldwasser, S. Micali and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17:281–308, 1988.
27. S. Jarecki and A. Lysyanskaya. Adaptively Secure Threshold Cryptography: Introducing Concurrency, Removing Erasures. In *Advances in Cryptology – EUROCRYPT '00* (LNCS 1807), 221–242, 2000.
28. J. Katz. Efficient and Non-Malleable Proofs of Plaintext Knowledge and Applications. In *ePrint Archive*, Report 2002/027, `http://eprint.iacr.org/`, 2002.
29. J. Kilian and E. Petrank. Concurrent and resettable zero-knowledge in polylogarithmic rounds. In *33rd ACM Symp. on Theory of Computing*, 560–569, 2001.
30. D. W. Kravitz. Digital signature algorithm. U.S. Patent 5,231,668, 27 July 1993.
31. L. A. Levin. Universal sorting problems. *Problemy Peredaci Informacii*, 9:115–116, 1973. In Russian. Engl. trans.: *Problems of Information Transmission* 9:265–266.
32. P. MacKenzie, T. Shrimpton, and M. Jakobsson. Threshold password-authenticated key exchange. In *Advances in Cryptology – CRYPTO 2002* (LNCS 2442), 385–400, 2002.
33. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM Symp. on Theory of Computing*, 427–437, 1990.
34. T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Advances in Cryptology – EUROCRYPT '98* (LNCS 1403), 380–318, 1998.
35. P. Paillier. Public-key cryptosystems based on composite degree residue classes. In *Advances in Cryptology – EUROCRYPT '99* (LNCS 1592), 223–238, 1999.
36. T. P. Pedersen. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *Advances in Cryptology – CRYPTO '91* (LNCS 576), 129–140, 1991.
37. M. Prabhakaran, A. Rosen and A. Sahai. Concurrent zero knowledge with logarithmic round-complexity, In *ePrint Archive, Report 2002/055*, `http://eprint.iacr.org/`, 2002. Also in *43rd IEEE Symp. on Foundations of Computer Sci.*, 366–375, 2002.
38. L. Reyzin. Zero-knowledge with public keys. Ph.D. Thesis, MIT, 2001.
39. J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd ACM Symp. on Theory of Computing*, 387–394, 1990.
40. A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th IEEE Symp. on Foundations of Computer Sci.*, 543–553, 1999.