Online/Offline OR Composition of Sigma Protocols

Michele Ciampi[1], Giuseppe Persiano[2], Alessandra Scafuro[3], Luisa Siniscalchi[1], and Ivan Visconti[1]

[1] DIEM, University of Salerno, ITALY
{mciampi,lsiniscalchi,visconti}@unisa.it
[2] DISA-MIS, University of Salerno, ITALY
giuper@gmail.com
[3] Boston University and Northeastern University, USA
scafuro@bu.edu

Abstract. Proofs of partial knowledge allow a prover to prove knowledge of witnesses for k out of n instances of NP languages. Cramer, Schoenmakers and Damgård [10] provided an efficient construction of a 3-round public-coin witness-indistinguishable (k, n)-proof of partial knowledge for any NP language, by cleverly combining n executions of Σ-protocols for that language. This transform assumes that all n instances are fully specified before the proof starts, and thus directly rules out the possibility of choosing some of the instances after the first round.
Very recently, Ciampi et al. [6] provided an improved transform where one of the instances can be specified in the last round. They focus on $(1, 2)$-proofs of partial knowledge with the additional feature that one instance is defined in the last round, and could be *adaptively* chosen by the verifier. They left as an open question the existence of an efficient $(1, 2)$-proof of partial knowledge where no instance is known in the first round. More in general, they left open the question of constructing an efficient (k, n)-proof of partial knowledge where knowledge of *all* n instances can be postponed. Indeed, this property is achieved only by inefficient constructions requiring NP reductions [19].
In this paper we focus on the question of achieving *adaptive-input* proofs of partial knowledge. We provide through a transform the first efficient construction of a 3-round public-coin witness-indistinguishable (k, n)-proof of partial knowledge where *all* instances can be decided in the third round. Our construction enjoys *adaptive-input* witness indistinguishability. Additionally, the proof of knowledge property remains also if the adversarial prover selects instances adaptively at last round as long as our transform is applied to a proof of knowledge belonging to the widely used class of proofs of knowledge described in [21,9]. Since knowledge of instances and witnesses is not needed before the last round, we have that the first round can be precomputed and in the online/offline setting our performance is similar to the one of [10].
Our new transform relies on the DDH assumption (in contrast to the transforms of [10,6] that are unconditional).

Keywords: Σ-protocols, WI, PoKs, delayed and adaptive input.

1 Introduction

Proofs of knowledge (PoKs) are ubiquitous in cryptographic protocols. When enjoying additional features such as honest-verifier zero knowledge (HVZK), witness indistinguishability (WI) or zero knowledge (ZK), they are used as building blocks in basically any protocol for secure computation. As such, the degree of security and efficiency achieved by the underlying PoKs directly (and dramatically) impacts on the security and efficiency of the larger protocol. For instance, very efficient WI PoKs for specific languages, such as Discrete Log and DDH, have been instrumental for constructing efficient maliciously secure two-party computation (see [17] and references within). Furthermore, stronger security notions of PoKs, such as soundness, WI and ZK in presence of *adaptive-input* selection, are useful for constructing round-efficient protocols ([25,18]).

Proofs of partial knowledge. In [10], Cramer et al. showed how to construct *efficient* PoKs for compound statements starting from Σ-protocols. More precisely, the compound statement consists of n instances, and the goal is to prove knowledge of witnesses for at least k of the n instances. As such, these proofs are named "*proofs of partial knowledge*" in [10]. The transform of [10] cleverly combines n parallel executions of PoKs that are Σ-protocols in an efficient 3-round public-coin perfect WI (k, n)-proof of partial knowledge. A similar result was given in [26].

Note that, if efficiency is not a concern, proofs of partial knowledge were already possible (with *computational* WI, though) thanks to the general construction of Lapidot and Shamir (LaSh) [19]. Proving compound statements via LaSh constructions however requires expensive NP reductions. On the other hand, LaSh PoKs provide a stronger security guarantee: honest players use the instances specified in the statements only in the last round, and security holds even if the adversarial verifier (resp., prover) chooses the instances adaptively after having seen the first (resp., second) round. LaSh's construction is therefore an *adaptive-input* WI proof of partial knowledge for all NP. As mentioned above, this property can be instrumental to save at least one round of communication, when the proof of partial knowledge is used in a larger protocol. The construction shown in [10], instead, although efficient, does not provide any form of adaptivity, as all the n instances must be fully specified before the protocols starts. As a consequence, the better efficiency of [10] can be paid in additional rounds compared to [19] when the construction is used in larger applications.

The proof of partial knowledge of [6]. A very recent work by Ciampi et al. [6] makes a first preliminary step towards closing the gap between [19] and [10]. [6] proposes a different transform for WI proofs of partial knowledge that gives some adaptiveness at the price of generality. Namely, their technique yields to a $(1, 2)$-proof of partial knowledge where the knowledge of one of the two instances can be postponed to the last round. In more details, they show a PoK for a statement "$x_0 \in L_0 \vee x_1 \in L_1$" such that x_0 and x_1 are not immediately needed (in contrast to [10]). The honest prover needs x_0 to run the 1st round while x_1 is needed only in the 3rd round along with a witness for either x_0 or x_1. The verifier needs to see x_0 and x_1 only at the end, in order to accept/reject the proof. Ciampi et al. [6] defined the property of delayed input requiring that the *honest* prover does not need to know the instance to start the protocol. In other words, the need of the input is delayed to the very last round. For clarity, we stress that a delayed-input protocol is not necessarily secure against inputs that have been adaptively chosen. Indeed, their technique yields a proof of partial knowledge that is delayed input for one of the instances but is not adaptively secure against malicious provers (although it is adaptive-input WI). The security achieved by their transform is sufficient for their target applications.

The open question and its importance. The above preliminary progress leaves open the following fascinating question: can we design an efficient transform that yields an adaptive-input WI (k, n)-proof of partial knowledge where *all* n instances are known only in the last round?

Previous efficient transforms require the a-priori knowledge of all instances or of one out of two instances, even if the corresponding languages admit efficient delayed-input Σ-protocols. For the sake of concreteness, assume one wants to prove knowledge of the discrete logarithm of at least one of g^{x_0} or g^{x_1}. There exists a very efficient Σ-protocol Σ^{dl} due to Schnorr [27], for proving knowledge of one discrete log and that also enjoys the delayed-input property, i.e., the prover can compute the first round without knowing the instance g^x. However, when we apply known transforms to combine Σ^{dl}, the resulting protocol loses the delayed-input property, as it will still need either both instances g^{x_0} and g^{x_1}, if using [10], or at least one g^{x_0}, to be specified in advance if using [6].

1.1 Our Results

In this work we study the above open question and give various positive answers.

Σ-Protocols and adaptive-input selection. We shed light on the relation between delayed-input Σ-protocols and adaptive-input Σ-protocols. Recall that a Σ-protocols enjoys a special soundness[4] property, which means that given two accepting transcripts for the same statement having the same first round, one can efficiently extract a witness for that statement.

We show that in general Σ-protocols are delayed-input but are not adaptive-input sound; that is, they are not sound if the malicious prover can choose the statements adaptively. Indeed, in Section 4.1 we show how a malicious prover, based on the second round played by the verifier, can craft a false statement that will make the verifier accept (and the extractor of special soundness fail even when the statement is true). The attack applies to very popular Σ-protocols like Schnorr's protocol for discrete logarithm (DLog), the protocol for proving equality of DLogs for Diffie-Hellman (DH) tuples and the protocol of [22] for

[4] In literature special soundness is often generalized to $\ell > 2$ accepting transcripts with the bound of ℓ being polynomial in the security parameter.

proving knowledge of committed messages. These protocols all fall into a well known class of protocols studied by Cramer in [9] and Maurer in [21].

The above issue was already noticed in [1] for the case of non-interactive zero-knowledge arguments obtained from Σ-protocols by applying the Fiat-Shamir transform [14]. Indeed there are in literature some incorrect use of the Fiat-Shamir transform where the instance is not given in input to the random oracle. As a consequence an adversarial prover can first create a transcript and then can try to find an instance not in the language such that the transcript is accepting. Of course in the random-oracle model the above issue has the trivial fix consisting of giving the instance as input to the random oracle to generate the challenge. This fix is meaningless in the standard model that is the focus of our work.

We then analyze the transform of [6], that is delayed-input with respect to one instance only. We observe that when [6] combines protocols belonging to the class of [9,21], it also suffers from the same attack, when the malicious prover is allowed to adaptively choose his input. Therefore the transform of [6] is not adaptive-input sound. We stress however, that in the applications targeted in [6] the input that is specified only in the last round is chosen by the verifier. As such, for their applications they do not need any form of adaptive-input soundness, but only adaptive-input witness-indistinguishability (which they achieve). Moreover, the special soundness of their transform preserves security w.r.t. adaptive-input selection. Summing up, [6] correctly defines and achieves delayed-input Σ-protocols and adaptive-input WI and uses it in the applications. However adaptive-input special soundness is not defined and not achieved in their work.

Adaptive-input special-sound Σ-protocols. In light of the above discussion, a natural question is whether we can upgrade the security of the class of Σ-protocols that are delayed input, but not adaptive-input sound. Towards this, we first clarify the conceptual gap between adaptive-input selection and the adaptiveness considered in [6] by defining formally adaptive-input special soundness. Then we show a compiler that takes as input any delayed-input Σ-protocol belonging to the class specified in [9,21], and outputs a Σ-protocol, that is adaptive-input sound, i.e., it is sound even when the malicious prover adaptively chooses his input in the last round. The main idea behind this compiler is to force the prover to send correctly the first round of the Σ-protocol through another parallel run of the Σ-protocol. This allows for the extraction of any witness in the proof of knowledge. The compiler is shown in Section 4.2. We also show (in Section 5) that nevertheless, [6]'s transform preserves the adaptivity of the Σ-protocols that are combined. Namely, on input Σ-protocols that are already adaptive-input special sound and WI, the [6]'s transform outputs a $(1, 2)$-proof of partial knowledge that is an adaptive-input proof of knowledge as well.

Adaptive-input (k, n)-proofs of partial knowledge. The main contribution of this paper is a new transform that yields the first efficient (k, n)-proofs of partial knowledge where *all* n instances can be specified in the last round.

Our new transform takes as input a delayed-input Σ-protocol for a relation $\mathcal{R}$, and outputs a 3-round public-coin WI special-sound (k, n)-proof of partial knowledge for the relation $(\mathcal{R} \vee \cdots \vee \mathcal{R})$ where no instance is known at the beginning. The security of our transform is based on the DDH assumption. The WI property of the resulting protocol holds also with respect to adaptive-input selection, while the PoK property holds also in case of adaptive-input selection only if the underlying Σ-protocol is adaptive-input special sound.

We also show a transform that admits instances taken from different relations. Interestingly, this construction makes use as subprotocol of the first construction where instances are taken from the same relation.

1.2 Our Technique

We provide a technique for composing a delayed-input Σ-protocol for a relation $\mathcal{R}$ into a delayed-input Σ-protocol for the (k, n)-proof of partial knowledge for relation $(\mathcal{R} \vee \ldots \vee \mathcal{R})$. For a better understanding of our technique, it is instructive to see why the transform of [10] (resp., [6]) requires that all n (resp., 1 out of 2) instances are specified before the protocol starts.

Limitations of previous transforms. Let $\Sigma_{\mathcal{R}}$ be a delayed-input Σ-protocol, and let $(\mathcal{R} \vee \ldots \vee \mathcal{R})$ be the relation for which we would like to have a (k, n)-proof of partial knowledge. The technique of [10] works as follows. The prover P, on input the instances $(x_1 \in \mathcal{R} \vee \ldots \vee x_n \in \mathcal{R})$, runs protocols $\Sigma_{\mathcal{R}}, \ldots, \Sigma_{\mathcal{R}}$ in parallel. P gets only k witnesses for k different instances but it needs to somehow generate an accepting transcript for *all* instances. How to prove the remaining $n - k$ instances without having the witness? The idea of [10] consists simply in letting the prover generate the $n - k$ transcripts (corresponding to the instances for which he did not get the witnesses) using the HVZK simulator S associated to the Σ-protocol. Additionally [10] introduces a mechanism that allows the prover to control the value of exactly $(n - k)$ of the challenges played by V, so that the prover can force the transcripts computed by the simulator in $(n - k)$ positions.

So, why does the transform of [10] need *all* instances to be known already in the 1st round? The answer is that P needs to run S already in the 1st round, and S expects the instance as input. Similar arguments apply for [6] as it requires that 1 instance out of 2 is known already in the 1st round.

The core idea of our technique. Previous transforms fail because the prover runs the HVZK simulator to compute the 1st round of some of the transcripts of $\Sigma_{\mathcal{R}}$. Our core idea is to provide mechanisms allowing P to postpone the use of the simulator to the 3rd round. The main challenge is to implement mechanisms that are very efficient and preserve soundness and WI of the composed Σ-protocol. We stress that we want to solve the open problems in full, and thus none of the instances are known at the beginning of the protocol. To be more explicit, in the 1st round, the prover starts with the following statement $(? \in L_{\mathcal{R}} \vee \ldots \vee ? \in L_{\mathcal{R}})$.

Assume we have a (k, n)-equivocal commitment scheme that allows the prover to compute n commitments such that k of them are binding and the remaining $n - k$ are equivocal, and the verifier cannot distinguish between the two types of commitment, where the k positions that are binding must be chosen already in the commitment phase (a similar tool is constructed in [24]). With this gadget in hand, we can construct a delayed-input (k,n)-proof of partial knowledge $\Sigma^{\mathsf{OR}}_{k,n}$ as follows. Let (a,c,z) denote generically the 3 messages exchanged during the execution of a Σ-protocol $\Sigma_{\mathcal{R}}$.

In the 1st round, P honestly computes a_i for the i-th execution of $\Sigma_{\mathcal{R}}$. Here we are using the fact that $\Sigma_{\mathcal{R}}$ is delayed-input, and thus a_i can be computed without using the instance. Then he commits to $a_1,\dots,a_n$ using the (k,n)-equivocal commitment scheme discussed above, where the k binding positions are randomly chosen. Thus, the 1st round of protocol $\Sigma^{\mathsf{OR}}_{k,n}$ consists of n commitments. In the 2nd round V simply sends a single challenge c according to $\Sigma_{\mathcal{R}}$. In the 3rd round, P obtains the n instances $x_1,\dots,x_n$ and k witnesses. At this point, for the instances x_i for which he did not receive the witness, he will use the HVZK simulator to compute an accepting transcript $(\tilde{a}_i, c, \tilde{z}_i)$ and then equivocate the $(n-k)$ equivocal commitments so that they decommit to the new generated $\tilde{a}_i$. For the k remaining instances he will honestly compute the 3rd round using the committed input a_i. Intuitively, soundness follows from the fact that k commitments are binding, and from the soundness of $\Sigma_{\mathcal{R}}$. WI follows from the hiding of the equivocal commitment scheme and the HVZK property of $\Sigma_{\mathcal{R}}$. Note that in this solution we are crucially using the fact that we are composing the *same* Σ-protocol so that P can use any of the a_i committed in the 1st round to compute an honest transcript. This technique thus falls short as soon as we want to compose arbitrary Σ-protocols together. Nevertheless, this transformation turns to be useful for the case of different Σ-protocols.

(k,n)-equivocal commitment scheme. A (k,n)-equivocal commitment scheme allows a sender to compute n commitments $\mathtt{com}_1,\dots,\mathtt{com}_n$ such that k of them are binding and $n-k$ are equivocal. We will use the language DH of DH tuples and we will call non-DH a tuple that is not a DH tuple. We will implement a (k,n)-equivocal commitment scheme very efficiently under the DDH assumption as follows. In the commitment phase, the sender computes n tuples $T_1=(g_1,A_1,B_1,X_1),\dots,T_n=(g_n,A_n,B_n,X_n)$ and proves that k out of n tuples are *not* in DH (i.e., they are non-DH tuples). We show that this can be done using the classical [10] (k,n)-proof of partial knowledge that can be obtained starting with a Σ-protocol Σ^{ddh} for DH. We then use the well known [12,4,5,17] fact that Σ-protocols can be used to construct an instance-dependent trapdoor commitment scheme, where the sender can equivocate if he knows the witness for the instance. Thus, each tuple T_i can be used to compute an instance-dependent trapdoor commitment $\mathtt{com}_i$ using Σ^{ddh}. $\mathtt{com}_i$ will be equivocal if T_i was indeed a DH tuple, it will be binding otherwise. Because the sender proves that k tuples are not in DH, it holds that there are at least k binding commitment. Hiding follows from the WI property of [10] and the HVZK of Σ^{ddh}. Commitment and decommitment can be completed in 3 rounds.

The case of different Σ-protocols. We now consider the case where we want to compose $\Sigma_1,\dots,\Sigma_n$ for possibly different relations. Our (k,n)-equivocal commitment does not help here because each a_i is specific to protocol Σ_i, and cannot be arbitrarily mixed and matched once the k witnesses are known.

For this case we thus use a different trick. We ask the prover to commit to each a_i twice, once using a binding commitment and once using an equivocal commitment. This again can be very efficiently implemented from the DDH assumption as follows. For each i, P generates tuples T_i^0 and T_i^1, that are such that at most one can be a DH tuple. It then commits to a_i twice using the instance-dependent trapdoor commitment associated to tuple T_i^0 and tuple T_i^1. Because at most one of the two tuples is a DH tuple, at most one of the commitments of a_i can be later equivocated. Thus the 1st round of our transformation consists of 2 commitments of a_i for $1 \leq i \leq n$. In the 3rd round, when P receives instances $x_1, \dots, x_n$ and k witnesses, he proceeds at follows. For each i, if P knows the witness for x_i, he will open the binding commitment for position i, and compute z_i using the honest prover procedure of Σ_i. Instead, if P does not have a witness for x_i, he will compute a new $\tilde{a}_i, z_i$ using the simulator on input x_i, c and open the equivocal commitment in position i. At the end, for each position i, one commitment has remained unopened.

This mechanism allows an honest prover to complete the proof with the knowledge of only k witnesses. However, what stops a malicious prover to always open the equivocal commitments and thus complete the proof without knowing any of the witnesses? We avoid this problem by requiring P to prove that, among the n tuples corresponding to the unopened commitments, at least k out of n tuples are DH tuples. This directly means that k of the opened commitments were constructed over non-DH tuples, and therefore are binding.

Now note that proving this theorem requires an (k, n)-proof of partial knowledge in order to implement Σ^{ddh}, where the instance to prove, i.e., the tuple that will be unopened, is known only in the 3rd round when P knows for which instances he is able to open a binding commitment. Here we crucially use the (k, n)-proof of partial knowledge for the same Σ-protocol developed above making sure to first run our compiler that strengthen Σ^{ddh} with respect to statements adaptively selected by a malicious prover.

1.3 Comparison with the State of the Art

In Table 1 we compare our results with the relevant related work. We consider [19], a 3-round public-coin WIPoK that is *fully adaptive-input* and that works for any NP language. We also consider [10] that proposed efficient 3-round public-coin WI proofs of partial knowledge (though, without supporting any adaptivity). Finally, we consider [6] since it was the only work that faced the problem of combining together efficiency and some form of delayed-input instances. The last row refers to our main result that allows to postpone knowledge of all the instances to the last round. The 2nd column refers to the computational assumptions needed by [19] (i.e., one-way permutations) and our main result (i.e., DDH assumption). The 3rd column specifies the type of WI depending on the adaptive selection of the instances from the adversarial verifier. The

4th column specifies the soundness depending on the adaptive selection of the instances from the adversarial prover.

	Assumption	Adaptive WI	Adaptive PoK	NP Reduction
LaSh90 [19]	OWP	k out of n (all adaptive)	k out of n (all adaptive)	Yes
CDS94 [10]	/	/	/	No
CPSSV16 [6]	/	1 out of 2 (1 adaptive)	/	No
This Work (main result)	DDH	k out of n (all adaptive)	k out of n (all adaptive)	No

Table 1: Comparison with previous work.

1.4 Online/Offline Computations

Our result has the advantage that the prover can compute the first round without knowing instances and witnesses. The first round is therefore an *offline phase*. When the prover interacts with the verifier (*online phase*) he sends the first round precomputed and computes only the third round of the protocol. We stress that [10] requires to know the instances already to compute the first round. Furthermore the work of [19] allows the prover to compute the first round offline but in the online phase the prover must perform an NP reduction.

In Table 2[5] we compare the effort of the prover in the online phase in our work and in [10,19]. We consider a prover that proves knowledge of discrete logarithms for 1 instance out of 2 instances (1st column) and a prover that proves knowledge of discrete logarithms for k instances out of n instances (2nd column). As we noted, above in the online phase of [19] the prover computes an NP reduction (2nd row). For our construction and the one of [10] we count the number of modular[6] exponentiations that are computed in the online phase (3rd and 4th rows). Below we briefly describe how we have computed the above costs. In [10] the number of exponentiations is $2n - k$. This comes from the fact that the first round of Schnorr's Σ-protocol requires one exponentiation while the simulator requires two exponentiations. In [10] the simulator is executed $2(n-k)$ times and moreover k exponentiations are needed to run the prover of Schnorr's protocol.

In the (1,2)-proof of partial knowledge of [6], the 1st round requires 3 exponentiations. Indeed, in the 1st round of [6] the prover runs Schnorr's simulator and computes the 1st round of Schnorr's Σ-protocol. The 3rd round of [6] has a different analysis depending on which witness is used. When the prover of [6]

[5] The actual amount of computations significantly depends on the precise versions of the subprotocols used in the construction. The adaptive-input special-sound versions of the subprotocols are more expensive than their non-adaptive counterparts.

[6] We will omit the word *modular* from now on.

	$(1, 2)$ DLogs	(k, n) DLogs
LaSh90 [19]	NP-reduction	NP-reduction
CDS94 [10]	3 exps	$2n - k$ exps
CPSSV16 [6]	4 exps	/
This Work (main result)	2 exps [4 exps]	$2(n - k)$ exps [$4(n - k)$ exps]

Table 2: Comparison with previous work proving knowledge of discrete logarithms. The table illustrates the computations of the prover in the online phase.

uses the witness for an adaptively chosen instance, then there is no addition exponentiation. Otherwise, another execution of Schnorr's simulator is required. For this reason, in the worst case the 3rd round of [6] costs two exponentiations. Note that in the execution of the construction of [6] 4 exponentiations are performed in the online phase, since only the 1st round of Schnorr's Σ-protocol can be precomputed.

The final row corresponds to our main result and shows the general case of k instances out of n. Our construction involves $10n - k$ exponentiations. Indeed a commitment computed according to the commitment scheme described previously based on DH tuples costs 4 exponentiations. In our construction in the 1st round we sample $n-k$ DH tuples and k non-DH, sampling a DH/non-DH tuple costs 3 exponentiations, so this operation costs $3n$. Also in the 1st round we compute $n-k$ equivocal commitments and k binding commitments, and this sums up to $2n+2k$ modular exponentiations. Furthermore the prover computes the 1st round of Schnorr's Σ-protocol n times and this costs n exponentiations. Moreover it has to run [10] to prove knowledge of witnesses for k instances out of n instances, and this costs $2n - k$ exponentiations. The only operations that involve exponentiations at the third round are the $n - k$ executions of the simulator of Schnorr's Σ-protocol. Therefore the online phase costs $2(n - k)$.

The adaptive-input special-sound version of our construction costs $13n - 3k$ exponentiations. Consider that in the adaptive-input special-sound version of Schnorr's Σ-protocol an execution of the simulator costs 4 exponentiations. Moreover computing the 1st round involves 2 exponentiations. Hence the first round of our adaptive-input special-sound construction involves $6n+k$ exponentiations and the online phase costs $4(n - k)$ exponentiations.

The exponentiations in square brackets specify the cost of our main result when Schnorr's Σ-protocol is transformed into an adaptive-input special-sound Σ-protocol. The analysis for the case of 1 out of 2 is similar with $k = 1$ and $n = 2$ but in this case, in the offline phase, we do not consider the cost of [10] since the correctness of the pair of tuples can be self-verified.

2 Preliminaries

We use λ as security parameter. $A(x)$ denotes the probability distribution of the output of a probabilistic algorithm A when running with x as input. We will use $A(x;r)$ to denote the randomness r used by A. PPT stands for probabilistic polynomial time.

If $\mathcal{R}$ is a subset of $\{0,1\}^\star \times \{0,1\}^\star$ for which membership of (x,w) to $\mathcal{R}$ can be decided in time polynomial in $|x|$ then we say that $\mathcal{R}$ is a polynomial-time relation and w is a witness for the instance x. Given a polynomial-time relation $\mathcal{R}$, $L_\mathcal{R}$ defined as $L_\mathcal{R} = \{x|\exists w : (x,w) \in \mathcal{R}\}$ is an NP language. For generality, we define $\hat{L}_\mathcal{R}$ to be the *input language* that includes both $L_\mathcal{R}$ and all well formed instances that do not have a witness, as already done in [15]. It follows that $L_\mathcal{R} \subseteq \hat{L}_\mathcal{R}$ and membership in $\hat{L}_\mathcal{R}$ can be tested in polynomial time. In proof systems for relation $\mathcal{R}$, the verifier runs the protocol only if the common input x belongs to $\hat{L}_\mathcal{R}$, while it rejects immediately common inputs not in $\hat{L}_\mathcal{R}$.

Given two interactive machines M_0 and M_1, we denote by $\langle M_0(x_0), M_1(x_1)\rangle(x_2)$ the output of M_1 when running on input x_1 with M_0 running on input x_0, both running on common input x_2.

Definition 1. *A pair $(\mathcal{P},\mathcal{V})$ of PPT interactive machines is a complete protocol for an NP-language L with relation $\mathcal{R}$ if the following property holds:*

- *Completeness. For every* common input $x \in L$ *and* witness w *such that* $(x,w) \in \mathcal{R}$, *it holds that* $\text{Prob}\,[\ \langle \mathcal{P}(w), \mathcal{V}\rangle(x) = 1\ ] = 1$.

Definition 2. *a complete protocol $(\mathcal{P},\mathcal{V})$ is a proof system for an NP-language L with relation $\mathcal{R}$ if the following property holds:*

- *Soundness. For every interactive machine $\mathcal{P}^\star$ there exists a negligible function ν such that for every $x \notin L$:* $\text{Prob}\,[\ \langle \mathcal{P}^\star, \mathcal{V}\rangle(x) = 1\ ] \leq \nu(|x|)$.

A proof system $(\mathcal{P},\mathcal{V})$ is public coin *if $\mathcal{V}$ sends only random bits.*

Definition 3 ([11]). *Let $k : \{0,1\}^* \to [0,1]$ be a function. A protocol $(\mathcal{P},\mathcal{V})$ is a* proof of knowledge *for the relation $\mathcal{R}$ with* knowledge error k *if the following properties are satisfied:*

- *Completeness: if $\mathcal{P}$ and $\mathcal{V}$ follow the protocol on input x and private input w to $\mathcal{P}$ where $(x,w) \in \mathcal{R}$, then $\mathcal{V}$ always accepts.*
- *Knowledge soundness: there exists a constant $c > 0$ and a probabilistic oracle machine* Extract*, called the* extractor*, such that for every interactive prover $\mathcal{P}^\star$ and every input x, the machine* Extract *satisfies the following condition. Let $\epsilon(x)$ be the probability that $\mathcal{V}$ accepts on input x after interacting with $\mathcal{P}^\star$. If $\epsilon(x) > k(x)$, then upon input x and oracle access to $\mathcal{P}^\star$, the machine* Extract *outputs a string w such that $(x,w) \in \mathcal{R}$ within an expected number of steps bounded by $|x|^c/(\epsilon(x) - k(x))$.*

A *transcript* τ of an execution of a public-coin protocol $\Pi = (\mathcal{P},\mathcal{V})$ for statement x consists of the sequence of messages exchanged $\mathcal{P}$ and $\mathcal{V}$. We say that τ is *accepting* if $\mathcal{V}$ outputs 1. Two accepting transcripts (a,c,z) and (a',c',z') for a 3-round public coin proof system with the same common input constitute a *collision* iff $a = a'$ and $c \neq c'$. The one message sent by the verifier $\mathcal{V}$ in a 3-round public coin proof system is called the *challenge*.

Σ-protocols. The most common form of proof system used in practice consists of 3-round protocols referred to as Σ-protocols. For several useful languages there exist efficient Σ-protocols, and they are easy to work with as already shown in many transforms [12,22,28,3,2,29,23,20,8].

Definition 4. *A 3-round public-coin protocol $\Pi = (\mathcal{P}, \mathcal{V})$ is a Σ-protocol for an* NP*-language L with polynomial-time relation $\mathcal{R}$ iff the following additional properties are satisfied:*

- *Completeness. When $\mathcal{P}, \mathcal{V}$ execute the protocol on input x and private input w to $\mathcal{P}$ where $(x, w) \in \mathcal{R}$, the verifier $\mathcal{V}$ always accepts.*
- *Special Soundness. There exists an efficient algorithm* Extract *that, on input x and a collision for x, outputs a witness w such that $(x, w) \in \mathcal{R}$.*
- *Special Honest Verifier Zero Knowledge (special HVZK, SHVZK). There exists a PPT* simulator *algorithm S that, on input an instance $x \in L$ and challenge c, outputs (a, z) such that (a, c, z) is an accepting w.r.t. x. Moreover, the distribution of the output of S on input (x, c) is perfectly[7] indistinguishable from the distribution of the transcript obtained when $\mathcal{V}$ sends c as challenge and $\mathcal{P}$ runs on common input x and any private input w such that $(x, w) \in \mathcal{R}$.*

A security parameter 1^λ for a Σ-protocol represents challenge length. Therefore we have that a Σ-protocol with a sufficiently large security parameter 1^λ is also a proof system.

Theorem 1 ([10]). *Every Σ-protocol is Perfect WI.*

Theorem 2 ([11]). *Let Π be a Σ-protocol for a relation $\mathcal{R}$ with security parameter λ. Then Π is a proof of knowledge with knowledge error $2^{-\lambda}$.*

From the above theorem we have that every Σ-protocol with a sufficiently long challenge is a proof of knowledge with negligible knowledge error. We observe that in the proof of the above theorem only completeness and special soundness of the Σ-protocol are used. Therefore the theorem regardless of HVZK. Furthermore, using the same proof approach used in the security proof of this theorem we can consider a relaxed notion of special soundness, t-special soundness, requiring $t \geq 2$ transcripts to extract the witness, with $t = \texttt{poly}(\lambda)$. This is still sufficient to obtain a proof of knowledge with negligible soundness error when the challenge is sufficiently long.

Therefore in this work when interested in proving the proof of knowledge property we will without loss of generality just prove t-special soundness for a polynomially bounded t and completeness.

Definition 5 (Delayed-Input Σ-protocol [6]). *A Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ for a relation $\mathcal{R}$ is* delayed-input *if $\mathcal{P}$ computes the first round having as input only the security parameter 1^λ and $\ell = |x|$.*[8]

[7] In this work we stick with the requirement of perfect SHVZK for Σ-protocols. Various other papers in literature considered also special *computational* HVZK.

[8] For simplicity in the rest of the paper we do not specify anymore that the algorithms $\mathcal{P}, \mathcal{V}$ take as input ℓ when the instance x is not known.

2.1 Adaptive-Input Special Soundness and Proof of Knowledge

The special soundness of a Σ-protocol strictly requires the statement $x \in L$ to be unchanged in the 2 accepting transcripts. We introduce a stronger notion referred to as *adaptive-input special soundness*. Roughly speaking, we require that it is possible to extract witnesses from a collision even if the two accepting 3-round transcripts are for two different instances. It is easy to see that adaptive-input special soundness implies extraction against provers that choose the theorem to be proved after seeing the challenge.

Definition 6. *A Σ-protocol Π for relation $\mathcal{R}$ enjoys* adaptive-input special soundness *if there exists an efficient algorithm* $\mathsf{AExtract}$ *that, on input accepting 3-round transcripts (a, c_1, z_1) for input x_1 and (a, c_2, z_2) for input x_2, outputs witnesses w_1 and w_2 such that $(x_1, w_1) \in \mathcal{R}$ and $(x_2, w_2) \in \mathcal{R}$.*

In this work we also define a protocol $\Pi = (\mathcal{P}, \mathcal{V})$ that is adaptive-input proof of knowledge. The adaptive-input proof of knowledge property is the same as the proof of knowledge property, with the difference that the adversarial prover $\mathcal{P}^\star$ can choose the statement when the last round is played. We require that the instance x given in output by $\mathsf{AExtract}$ must be perfect indistinguishable from an instance x' given in output by $\mathcal{P}^\star$ in an execution of Π with $\mathcal{V}$. The previous discussion about proving the proof of knowledge property from ℓ-special soundness also applies when proving adaptive-input proof of knowledge from adaptive-input ℓ-special soundness.

2.2 Adaptive-Input Witness Indistinguishability

The notion of *adaptive-input WI* formalizes security of the prover with respect to an adversarial verifier $\mathcal{A}$ that adaptively chooses the input instance to the protocol; that is, after seeing the first message of the prover. More specifically, for a delayed-input 3-round complete protocol Π, we consider game $\mathsf{ExpAWI}_{\Pi,\mathcal{A}}$ between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ in which the instance x and two witnesses w_0 and w_1 for x are chosen by $\mathcal{A}$ *after* seeing the first message of the protocol played by the challenger. The challenger then continues the game by randomly selecting one of the two witnesses, w_b, and by computing the third message by running the prover's algorithm on input the instance x, the selected witness w_b and the challenge received from the adversary. The adversary wins the game if she can guess which of the two witnesses was used by the challenger.

We now define the adaptive-input WI experiment $\mathsf{ExpAWI}_{\Pi,\mathcal{A}}(\lambda, \mathsf{aux})$. This experiment is parameterized by a delayed-input 3-round complete protocol $\Pi = (\mathcal{P}, \mathcal{V})$ for a relation $\mathcal{R}$ and by PPT adversary $\mathcal{A}$. The experiment has as input the security parameter λ and auxiliary information aux for $\mathcal{A}$.

$\mathsf{ExpAWI}_{\Pi,\mathcal{A}}(\lambda, \mathsf{aux})$:

1. $\mathcal{C}$ randomly selects coin tosses r and runs $\mathcal{P}$ on input $(1^\lambda; r)$ to obtain a;
2. $\mathcal{A}$, on input a and aux, outputs instance x, witnesses w_0 and w_1 such that $(x, w_0), (x, w_1) \in \mathcal{R}$, challenge c and internal state $\mathtt{state}$;

3. $\mathcal{C}$ randomly selects $b \leftarrow \{0,1\}$ and runs $\mathcal{P}$ on input (x, w_b, c) to obtain z;
4. $b' \leftarrow \mathcal{A}((a,c,z), \mathsf{aux}, \mathtt{state})$;
5. if $b = b'$ then output 1 else output 0.

We set $\mathsf{AdvAWI}_{\Pi,\mathcal{A}}(\lambda, \mathsf{aux}) = \left|\text{Prob}\left[\ \mathsf{ExpAWI}_{\Pi,\mathcal{A}}(\lambda, \mathsf{aux}) = 1\ \right] - \frac{1}{2}\right|$.

Definition 7 (Adaptive-Input Witness Indistinguishability). *A delayed-input 3-round complete protocol Π is* adaptive-input WI *if for any PPT adversary $\mathcal{A}$ there exists a negligible function ν such that for any* $\mathsf{aux} \in \{0,1\}^*$ *it holds that* $\mathsf{AdvAWI}_{\Pi,\mathcal{A}}(\lambda, \mathsf{aux}) \leq \nu(\lambda)$.

About DDH. The DDH assumption posits the hardness of distinguishing a randomly selected DH tuple from a randomly selected non-DH tuple with respect to a *group generator* algorithm IG. For sake of concreteness, we consider a specific group generator that, on input 1^λ, randomly selects a λ-bit prime p such that $q = (p-1)/2$ is also prime and outputs the (description of the) order q group $\mathcal{G}$ of the quadratic residues modulo p along with a random generator g of $\mathcal{G}$.

2.3 A Σ-Protocol for Partial Knowledge of DH/Non-DH Tuples

Let $\mathcal{G}$ be a cyclic group of order p. We say that $T = (g, A, B, X) \in \mathcal{G}^4$ is *oneNDH* if there exits $\alpha, \beta \in Z_p$ such that $A = g^\alpha, B = g^\beta, X = g^{\alpha\beta+1}$. In this section we describe a Σ-protocol for proving that at least k out of n tuples are oneNDH. The Σ-protocol is based on the one of [10] and we stress that, just as in [10], the Σ-protocol is perfect WI.

Formally, for $1 \leq k \leq n-1$, we construct Σ-protocol $\Pi^{nddh}_{k,n} = (\mathcal{P}_{k,n}, \mathcal{V}_{k,n})$ for the polynomial-time relation

$$\mathsf{NDH}_{k,n} = \big\{ \big(((g_1, A_1, B_1, X_1), \ldots, (g_n, A_n, B_n, X_n)), (\alpha_{i_1}, \ldots, \alpha_{i_k}, \beta_{i_1}, \ldots, \beta_{i_k}) \big) : \\ 1 \leq i_1 < \cdots < i_k \leq n \wedge\ A_{i_j} = g_{i_j}^{\alpha_{i_j}} \wedge B_{i_j} = g_{i_j}^{\beta_{i_j}} \wedge X_{i_j} = g_{i_j}^{\alpha_{i_j}\beta_{i_j}+1},\ \text{for } j = 1, \ldots, k \big\}$$

of the sequences of the n-tuples such that at least k of them are oneNDH. The prover $\mathcal{P}_{k,n}$ and the verifier $\mathcal{V}_{k,n}$ of $\Pi^{nddh}_{k,n}$, on input n tuples $(g_1, A_1, B_1, X_1), \ldots$
$\ldots, (g_n, A_n, B_n, X_n)$ constructs tuples (g_i, A_i, B_i, Y_i) setting $Y_i = X_i/g_i$,
for $i = 1, \ldots, n$.

Then prover and verifier start Σ-protocol Σ^{ddh} of [10] for proving that at least k of n constructed tuples are DH.

Theorem 3. *For every n and $1 \leq k \leq n-1$, $\Pi^{ddh}_{k,n}$ is a Σ-protocol for the polynomial-time relation* $\mathsf{NDH}_{k,n}$ *with perfect WI.*

Proof. The perfect WI property follows from the perfect WI of [10]. The proof is then completed by the following two simple observations. If at least k of the input tuples are oneNDH then at least k of the *constructed* tuples (g_i, A_i, B_i, Y_i) are DH and the prover has a witness of this fact. On the other hand, if fewer than k of the input tuples are oneNDH then the transformed tuples contain fewer than k DH tuples.

2.4 Commitments from Σ-protocols

We define the notion of an Instance-Dependent Trapdoor Commitment scheme associated with a polynomial-time relation $\mathcal{R}$ and show a construction that uses Σ-protocols and fits this definition.

Definition 8 (Instance-Dependent Trapdoor Commitment scheme). *Let $\mathcal{R}$ be a polynomial-time relation. An* Instance-Dependent Trapdoor Commitment *(a* IDTC*, in short) scheme for $\mathcal{R}$ with message space M is a quadruple of PPT algorithms* $(\mathsf{Com}, \mathsf{Dec}, (\mathsf{Fake}_1, \mathsf{Fake}_2))$ *where* Com *is the randomized* commitment *algorithm that takes as input an instance $x \in \hat{L}_{\mathcal{R}}$ (with $|x| = \mathtt{poly}(\lambda)$) and a message $m \in M$ and outputs* commitment $\mathtt{com}$ *and* decommitment $\mathtt{dec}$*.* Dec *is the* verification *algorithm that takes as input $(x, \mathtt{com}, \mathtt{dec}, m)$ and decides whether m is the decommitment of* $\mathtt{com}$*.*

$(\mathsf{Fake}_1, \mathsf{Fake}_2)$ *are randomized algorithms.* Fake_1 *takes as input an instance x, a witness w s.t. $(x, w) \in \mathcal{R}$ ($|x| = \mathtt{poly}(\lambda)$) and outputs* commitment $\mathtt{com}$*, and* equivocation information $\mathtt{rand}$. Fake_2 *takes as input x, w, m, and* $\mathtt{rand}$*, and outputs* $\mathtt{dec}$ *s.t.* Dec*, on input $(x, \mathtt{com}, \mathtt{dec}, m)$, accepts m as decommitment of* $\mathtt{com}$*.*

An Instance-Dependent Trapdoor Commitment scheme has the following properties:

- **Correctness***: for all $x \in \hat{L}_{\mathcal{R}}$, all $m \in M$, it holds that*
$$\text{Prob}\left[\ (\mathtt{com}, \mathtt{dec}) \leftarrow \mathsf{Com}(x, m) : \mathsf{Dec}(x, \mathtt{com}, \mathtt{dec}, m) = 1\ \right] = 1.$$
- **Binding***: if $x \notin L$ then for every commitment* $\mathtt{com}$ *there exists at most one message m s.t. $\mathsf{Dec}(x, \mathtt{com}, \mathtt{dec}, m) = 1$ for any value* $\mathtt{dec}$*.*
- **Hiding***: for every receiver $\mathcal{A}$, for every auxiliary information* aux*, for all $x \in L_{\mathcal{R}}$ and all for $m_0, m_1 \in M$, it holds that*
$$\text{Prob}\left[\ b \leftarrow \{0,1\}; (\mathtt{com}, \mathtt{dec}) \leftarrow \mathsf{Com}(1^\lambda, x, m_b) : b = \mathcal{A}(\mathsf{aux}, x, \mathtt{com}, m_0, m_1)\ \right] \le \frac{1}{2}.$$
- **Trapdoorness***: the following two families of probability distributions are perfect indistinguishable (namely the two probability distributions coincide for all (x, w, m) such that $(x, w) \in \mathcal{R}$ and $m \in M$):*
$$\{(\mathtt{com}, \mathtt{rand}) \leftarrow \mathsf{Fake}_1(x, w); \mathtt{dec} \leftarrow \mathsf{Fake}_2(x, w, m, \mathtt{rand}) : (\mathtt{com}, \mathtt{dec})\}$$
$$\{(\mathtt{com}, \mathtt{dec}) \leftarrow \mathsf{Com}(x, m) : (\mathtt{com}, \mathtt{dec})\}.$$

IDTC from Σ-protocol. Our construction follows similar constructions of [11,17,13]. Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a Σ-protocol for polynomial-time relation $\mathcal{R}$ with the associated NP-language $L_{\mathcal{R}}$ and challenge length λ. Let S be the special HVZK simulator for Π and let (x, w) be s.t. $(x, w) \in \mathcal{R}$. Now we show an IDTC $\mathsf{CS}^\Pi = (\mathsf{Com}^\Pi, \mathsf{Dec}^\Pi, (\mathsf{Fake}_1^\Pi, \mathsf{Fake}_2^\Pi))$.

- Com^Π takes as input instance x and message $m \in \{0,1\}^\lambda$, sets $(\mathtt{com}, \mathtt{dec}) \leftarrow S(x, m)$ and outputs $(\mathtt{com}, \mathtt{dec})$.

- Dec^{Π} takes as input instance x and transcript $(\mathtt{com}, m, \mathtt{dec})$, runs $\mathcal{V}$ on input the instance and the transcript and returns $\mathcal{V}$'s output.
- Fake_1^{Π} takes as input instance x and witness w, samples random string ρ and runs $\mathcal{P}$ on input $(1^\lambda, x, w; \rho)$ to get the 1st message a of Π. Fake_1^{Π} sets $\mathtt{rand} = \rho$, $\mathtt{com} = \mathsf{a}$ and outputs $(\mathtt{com}, \mathtt{rand})$.
- Fake_2^{Π} takes as input $x, w, m, \mathtt{rand}$ and runs $\mathcal{P}$ on input $(1^\lambda, x, w, m, \mathtt{rand})$ to get the 3rd message z of Π. Fake_1^{Π} sets $\mathtt{dec} = \mathsf{z}$ and outputs $\mathtt{dec}$.

Theorem 4. CS^{Π} *is an IDTC.*

Proof. The security proof relies only on the properties of Π. Correctness follows from the completeness of Π. Binding follows from the special soundness of Π. Hiding and Trapdoorness follow from the SHVZK and the completeness of Π.

3 Adaptive-Input (k, n)-Proof of Partial Knowledge

In this section we describe in details our new transform for compound statements. For the high-level overview the reader is referred to Sec. 1.2.

Let $\mathcal{R}$ be a polynomial-time relation admitting a delayed-input Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$. Recall that delayed-input means that the prover does not need the instances of the statement to play the 1st round.

We describe a compiler that on input Π for $\mathcal{R}$ outputs a delayed-input WIPoK $\Pi^k = (\mathcal{P}^k, \mathcal{V}^k)$ for the (k, n)-threshold relation $\mathcal{R}_k$ defined as follows

$$\begin{aligned}\mathcal{R}_k = \big\{ \big((x_1, \ldots, x_n), (w_{i_1}, \ldots w_{i_k})\big) : 1 \leq i_1 < \cdots < i_k \leq n \\ \text{and } (x_{i_j}, w_{i_j}) \in \mathcal{R}, \text{ for } j = 1, \ldots, k \text{ and } x_j \in \hat{L}_{\mathcal{R}}, \text{ for } j = 1, \ldots, n \big\}.\end{aligned}$$

The main tools involved in our construction are the protocol $\Pi_{k,n}^{nddh}$ described in Section 2.3, and an IDTC scheme described in Section 2.4. More precisely the IDTC scheme is constructed using a Σ-protocol for DDH. Therefore given a tuple $T = (g, A, B, X)$ (either DH or non-DH), a message m and a randomness r, we can compute $(\mathtt{com}, \mathtt{dec})$ using the scheme described in Section 2.4. If T is a DH tuple, with $A = g^\alpha$, then α represents the trapdoor for the commitment $\mathtt{com}$ and $\mathtt{dec}$ is equal to $\perp$. In this case given a randomness r, $\mathtt{com}$, the tuple T and α for every message m it is possible to compute $\mathtt{dec}$ such that a receiver accepts $\mathtt{com}$ as a commitment of the message m.

1st round. $\mathcal{P}^k \Rightarrow \mathcal{V}^k$:

1. Set $(\mathcal{G}, p, g) \leftarrow \mathsf{IG}(1^\lambda)$.
2. Randomly choose tuples $T_1 = (g_1, A_1, B_1, X_1), \ldots, T_n = (g_n, A_n, B_n, X_n)$ of elements of $\mathcal{G}$ under the constraint that exactly k are oneNDH and $n-k$ are DH, along with $\alpha_1, \ldots, \alpha_n$ such that $A_i = g_i^{\alpha_i}$, for $i = 1, \ldots, n$.
3. Let $b_1, \ldots, b_k$ denote the indices of the k oneNDH tuples and $\widetilde{b}_1, \ldots, \widetilde{b}_{n-k}$ denote the indices of the $n-k$ DH tuples.
4. Run the prover of $\Pi_{k,n}^{nddh}$ on input $T = (T_1, \ldots, T_n)$, witnesses $(\alpha_{b_1}, \ldots, \alpha_{b_k})$ and randomness $r_{k,n}$ thus obtaining message $a_{k,n}$. Send $a_{k,n}$ to $\mathcal{V}^k$.

5. For $i = 1, \dots, n$:
   Compute the first round a_i of Π by running $\mathcal{P}$ with randomness r_i.
   Compute pair $(\mathtt{com}_i, \mathtt{dec}_i)$ of commitment and decommitment of a_i using T_i.
   Send $(T_i, \mathtt{com}_i)$ to $\mathcal{V}^k$.

2nd round. $\mathcal{V}^k \Rightarrow \mathcal{P}^k$: randomly select a challenge c and send it to $\mathcal{P}^k$.

3rd round. $\mathcal{P}^k \Rightarrow \mathcal{V}^k$:

1. Receive inputs $(x_1, \dots, x_n)$ and witnesses $(w_{d_1}, \dots, w_{d_k})$ for inputs $x_{d_1}, \dots, x_{d_k}$ (we denote by $\widetilde{d}_1, \dots, \widetilde{d}_{n-k}$ the indices of the inputs for which no witness has been provided).
2. Compute the third round of $\Pi^{nddh}_{k,n}$ using c as challenge to get $z_{n,k}$ and send it to $\mathcal{V}^k$.
3. Pick a random permutation σ of $\{1, \dots, k\}$ to associate each of the k oneNDH tuples $T_{b_1}, \dots, T_{b_k}$ with one of the k inputs $x_{d_1}, \dots, x_{d_k}$ for which a witness is available.
4. For $i = 1, \dots, k$:
   Set $j = d_{\sigma(i)}$ and $t_j = b_i$.
   Compute z_j by running $\mathcal{P}$ on input (x_j, w_j), a_{t_j}, randomness r_{t_j} and challenge c.
   Set $M_j = (j, t_j, \mathtt{dec}_{t_j}, a_{t_j}, z_j)$.
5. Pick a random permutation τ of $\{1, \dots, n-k\}$ to associate each of the $n-k$ DH tuples $T_{\widetilde{b}_1}, \dots, T_{\widetilde{b}_{n-k}}$ to one of the k inputs $x_{\widetilde{d}_1}, \dots, x_{\widetilde{d}_{n-k}}$ for which no witness is available.
6. For $i = 1, \dots, n-k$:
   Set $j = \widetilde{d}_{\tau(i)}$ and $t_j = \widetilde{b}_i$.
   Run simulator S on input x_j and c obtaining (a_j, z_j).
   Use trapdoor α_{t_j} to compute decommitment $\mathtt{dec}_{t_j}$ of $\mathtt{com}_{t_j}$ as a_j.
   Set $M_j = (j, t_j, \mathtt{dec}_{t_j}, a_j, z_j)$.
7. For $j = 1, \dots, n$: send M_j to $\mathcal{V}^k$.

$\mathcal{V}^k$ accepts if and only if all the following conditions are satisfied:

1. $(a_{n,k}, c, z_{n,k})$ is an accepting transcript for $\mathcal{V}^{nddh}_{k,n}$ with input T.
2. All t_j's are distinct.
3. For $j = 1, \dots, n$: $\mathtt{dec}_{t_j}$ is a valid decommitment of $\mathtt{com}_{t_j}$ with respect to T_{t_j}.
4. For $j = 1, \dots, n$: (a_j, c, z_j) is accepting for $\mathcal{V}$ with input x_j.

We will show now that Π^k is a (adaptive-Input) PoK and is adaptive-input WI for the relation $\mathcal{R}_k$.

3.1 (Adaptive-Input) Proof of Knowledge

Theorem 5. *Protocol Π^k is a proof of knowledge for $\mathcal{R}_k$.*

Proof. The completeness property follows from the completeness of protocols $\Pi^{nddh}_{k,n}$ and Π, and from the correctness and trapdorness property of the Instance-Dependent Trapdoor Commitment scheme used.

Now we proceed by proving that our protocol is $((n-1) \cdot k + 2)$-special sound and then, using the arguments of Section 2 about the proof of knowledge property of protocols that enjoy t-special soundness, we can conclude the proof claiming that Π^k is a proof of knowledge. There exists an efficient extractor that, for any sequence $(x_1, \ldots, x_n)$ of n inputs and for any set of $N = (n-1) \cdot k + 2$ accepting transcripts of Π^k that share the same first message and have different challenges, outputs the witnesses of k of the n inputs. The extractor is based on the following observations.

First of all, observe that, by the special soundness of $\Pi_{n,k}$, it is possible to extract the witness that k of the tuple $T_1, \ldots, T_n$ appearing in the first message are oneNDH. Let us denote by $b_1, \ldots, b_k$ the indices of the oneNDH tuples. This implies that commitments $\mathtt{com}_{b_1}, \ldots, \mathtt{com}_{b_k}$ that appear in the shared first round of the N transcripts will be opened to the same strings $a_{b_1}, \ldots, a_{b_k}$. We also observe that if two transcripts use the same input x_i with the same oneNDH tuple T_{b_i} then we can extract two transcripts of the Σ-protocol Π that share the same first message and have two different challenges. By the special soundness of Π there exists an extractor that efficiently extracts a witness. In other words, in order to be able to extract a witness for x_i, x_i has to be associated with the same oneNDH tuple in two distinct transcripts.

The extractor willing to get k witnesses considers the N transcripts one at the time and stops as soon as it reaches a *special* transcript C^l in which, for $j = 1, \ldots, k$, tuple T_{b_j} is associated with input x_{d_j} in C^l and in at least a transcript C^{l_j} with $l_j < l$. Clearly, once such a transcript is reached the extractor has obtained k witnesses. Now observe that a pair (oneNDH tuple, input x_i) can be used to eliminate at most one transcript. Moreover, there are $n \cdot k$ such pairs and the first transcript exhibits exactly k of these pairs. Therefore the set of N input transcripts contains at least one special transcript.

Theorem 6. *If Π is adaptive-input special sound then Π^k is an adaptive-input proof of knowledge for $\mathcal{R}_k$.*

Proof. We prove the following stronger statement. There exists an efficient algorithm that on input 2 accepting transcripts (a, c_1, z_1) (a, c_2, z_2) for Π^k, where

- the first one is accepting with respect to a sequence of n theorems $(x_1^1, \ldots, x_n^1)$,
- the second one is accepting with respect to a sequence of n (potentially different from the previous one) theorems $(x_1^2, \ldots, x_n^2)$,
- share the same first round and
- have different challenges,

outputs, for each of the two sequence, k witnesses (for a total of $2 \cdot k$ witnesses). The extractor is based on the following observations.

First of all, observe that, by the special soundness of protocol $\Pi_{n,k}$, it is possible to extract the witness certifying that k of the tuple $T_1, \ldots, T_n$ appearing in the first message are oneNDH let us denote by $b_1, \ldots, b_k$ the indices of the oneNDH tuples. This implies that commitments $\mathtt{com}_{b_1}, \ldots, \mathtt{com}_{b_k}$ that appear in the common first round of the N transcripts will be opened to the same strings $a_{b_1}, \ldots, a_{b_k}$.

To conclude the proof we observe that if two transcripts use the same oneNDH tuple T_{b_i} then we can obtain two transcripts of Σ-protocol Π that share the same first message and have two different challenges. By the adaptive-input special-soundness property of Π there exists an extractor that outputs a witness.

3.2 Adaptive-Input Witness Indistinguishability

Here we prove that Π^k is WI even when $\mathcal{A}$ can select instances and witnesses adaptively after receiving the first round. We have the following theorem.

Theorem 7. *Under the DDH assumption, if Π is SHVZK for $\mathcal{R}$ then Π^k is adaptive-input WI for relation $\mathcal{R}_k$.*

Proof. Let us fix a PPT adversary $\mathcal{A}$ and let us denote by X and W^0 and W^1 the instance and the witnesses of Π^k output by $\mathcal{A}$ at Step 2 of $\mathsf{ExpAWI}_{\Pi^k,\mathcal{A}}$. More precisely, we let $X = (x_1, \dots, x_n)$ be the sequence of n instances output by $\mathcal{A}$ and $W^0 = ((w_1^0, d_1^0), \dots (w_k^0, d_k^0))$ and $W^1 = ((w_1^1, d_1^1), \dots (w_k^1, d_k^1))$ the two sequences of witnesses. We remark that $(x_{d_i^b}, w_i^b) \in \mathcal{R}$ for $i = 1, \dots, k$ and $b = 0, 1$ and that $i \neq j$ implies that $d_i^0 \neq d_j^0$ and $d_i^1 \neq d_j^1$.

Let $m \leq k$ be the number of instances of Π in X for which W^1 contains a witness but W^0 does not. Obviously, since W^0 and W^1 contain witnesses for the same number k of instances of Π in X, it must be the case that m is also the number of instances of Π in X for which W^0 contains a witness and W^1 does not. We can rename the instances of X, so that W^0 and W^1 can be written as

$$W^0 = \big((w_1^0, m+1), \dots, (w_m^0, 2m), (w_{m+1}^0, 2m+1), (w_k^0, m+k)\big) \text{ and}$$

$$W^1 = \big((w_1^1, 1), \dots, (w_m^1, m), (w_{m+1}^1, 2m+1), \dots, (w_k^1, m+k)\big).$$

For our proof we now consider the case in which $m = 0$ and $m \neq 0$. When $m = 0$ we have that W^1 and W^0 contains witnesses for the same theorems (for Π). Therefore by the perfect-WI property of Π[9] we can claim that if $m = 0$ then $\mathsf{AdvAWI}_{\Pi^k,\mathcal{A}}(\lambda, \mathsf{aux}) = 0$. Now we consider the more interesting case where $m \neq 0$.

We define the intermediate sequences of witnesses $W_1, \dots, W_k$ in the following way.

1. For $i = 0, \dots, m$: W_i consists of witnesses

$$W_i = \big((w_1^1, 1), \dots, (w_i^1, i), (w_{i+1}^0, m+i+1), \dots \\ \dots, (w_m^0, 2m), (w_{m+1}^0, 2m+1), \dots, (w_{m+k}^0, m+k)\big).$$

   Note that W_i contains witnesses for $(x_1, \dots, x_i, x_{m+1+i}, \dots, x_{2m})$. Moreover, W_0 coincides with W^0 and in W_m the first m witnesses are from W^1 and the remaining are from W^0.

[9] We observe that Σ-protocols enjoy SHVZK and therefore by Theorem 1 we can claim that every Σ-protocol is also perfect WI.

2. For $i = m+1, \ldots, k$: W_i consists of witnesses

$$W_i = \big((w_1^1, 1), \ldots, (w_m^1, m), (w_{m+1}^1, 2m+1), \ldots \\ \ldots, (w_{m+i}^1, m+i), (w_{m+i+1}^0, m+i+1), \ldots, (w_{m+k}^0, m+k)\big).$$

   It is easy to see that W_k coincides with W^1.

For $i = 0, \ldots, k$, we define hybrid experiment $\mathcal{H}_i$ as the experiment in which the challenger $\mathcal{C}$ uses sequence of witnesses W_i to complete the third step of the experiment $\mathsf{ExpAWI}_{\Pi^k,\mathcal{A}}$. Clearly, $\mathcal{H}_0$ is the experiment $\mathsf{ExpAWI}_{\Pi^k,\mathcal{A}}$ when $\mathcal{C}$ picks $b = 0$ and $\mathcal{H}_k$ is the same experiment when $\mathcal{C}$ picks $b = 1$. We conclude the proof by showing that, for $i = 0, \ldots, k-1$, $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ are indistinguishable.

We start by proving indistinguishability of $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ for $i = 0, \ldots, m-1$. We remind the reader that, in $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$, the challenger $\mathcal{C}$ uses witnesses for the following k inputs:

$\mathcal{H}_i$	$x_1 \cdots x_i$		x_{m+i+1}	$x_{m+i+2} \cdots x_{2m}$	$x_{2m+1} \cdots x_{m+k}$
$\mathcal{H}_{i+1}$	$x_1 \cdots x_i$	x_{i+1}		$x_{m+i+2} \cdots x_{2m}$	$x_{2m+1} \cdots x_{m+k}$

To prove indistinguishability of $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ we consider six intermediate hybrids: $\mathcal{H}_i^1, \ldots, \mathcal{H}_i^6$.

1. $\boldsymbol{\mathcal{H}_i^1(\lambda, \mathsf{aux})}$ differs from $\mathcal{H}_i(\lambda, \mathsf{aux})$ in the way that the accepting transcript for the theorem x_{i+1} is computed. More precisely in $\mathcal{H}_i(\lambda, \mathsf{aux})$ the SHVZK simulator of Π was used to compute the transcript for x_{i+1} while in $\mathcal{H}_i^1(\lambda, \mathsf{aux})$ the transcript for x_{i+1} is computed using the honest-prover procedure that has also w_{i+1} as input. To prove the indistinguishability of the hybrids we can easily invoke the SHVZK property of Π. We remark that this is possible only because the commitment of the first round of Π with respect to the theorem x_{i+1} is hiding.
2. $\boldsymbol{\mathcal{H}_i^2(\lambda, \mathsf{aux})}$ differs from $\mathcal{H}_i^1(\lambda, \mathsf{aux})$ in the way the tuples used to compute the commitments are chosen. More precisely k tuples oneNDH are chosen, $n-k-1$ tuple DH are chosen and the last tuple is chosen non-DH. The additional non-DH tuple is used to compute the commitment of the first message of Π that will be associated to the theorem x_{i+1} in the third round. Even in this case is possible to compute an accepting transcript for Π^k because $k+1$ witnesses are used instead of k, therefore there is no problem if $k+1$ commitments are binding. The indistinguishability between the two hybrids is ensured by the DDH-assumption.
3. $\boldsymbol{\mathcal{H}_i^3(\lambda, \mathsf{aux})}$ the only difference between this hybrid experiment and $\mathcal{H}_i^2(\lambda, \mathsf{aux})$ is that instead of a non-DH tuple, a oneNDH tuple is chosen. As in the previous hybrid experiment the considered tuple is used to compute the commitment of the first message of Π that will be associated to the theorem x_{i+1} in the third round. The indistinguishability between the two hybrids is ensured by the DDH-assumption.
4. $\boldsymbol{\mathcal{H}_i^4(\lambda, \mathsf{aux})}$. The differences between this hybrid and $\mathcal{H}_i^3(\lambda, \mathsf{aux})$ are that we use k tuples oneNDH $n-k-1$ tuples DH and one tuple non-DH. In this

case the additional non-DH tuple is used to commit the first round of Π that will be use as the first round of the accepting transcript with respect to x_{m+i+1}. By the DDH-assumption and perfect WI property of $\Pi_{n,k}$ we can claim that this hybrid is indistinguishable from the previous one.

5. $\boldsymbol{\mathcal{H}_i^5(\lambda, \mathsf{aux})}$. The differences between this hybrid and $\mathcal{H}_i^4(\lambda, \mathsf{aux})$ are that we again use k oneNDH and $n-k$ DH tuples. In this case the additional DH tuple is used to commit to the first round of Π that will be used as the first round of the accepting transcript with respect to x_{m+i+1}. By the DDH-assumption we can claim that this hybrid is indistinguishable from the previous one.
6. $\boldsymbol{\mathcal{H}_i^6(\lambda, \mathsf{aux})}$ differs from $\mathcal{H}_i^5(\lambda, \mathsf{aux})$ in the way that the accepting transcript for the theorem x_{m+i+1} is computed. More precisely in $\mathcal{H}_i^5(\lambda, \mathsf{aux})$ the honest-prover procedure of Π was used to compute the accepting transcript for x_{m+i+1}. In $\mathcal{H}_i^5(\lambda, \mathsf{aux})$ the transcript for x_{m+i+1} is computed using the SHVZK simulator of Π. To prove the indistinguishability of this hybrid we invoke the SHVZK property of Π. We remark that this is possible only because the commitment of the first round of Π with respect to the theorem x_{m+i+1} is hiding. We observe that this hybrid is equal to $\mathcal{H}_{i+1}(\lambda, \mathsf{aux})$.

Now we are able to complete the first part of the proof observing that [10].

$$\mathcal{H}_i(\lambda, \mathsf{aux}) \approx \mathcal{H}_i^1(\lambda, \mathsf{aux}) \approx \cdots \approx \mathcal{H}_i^6(\lambda, \mathsf{aux}) = \mathcal{H}_{i+1}(\lambda, \mathsf{aux}).$$

We have thus proved that $\mathcal{H}_0$ and $\mathcal{H}_m$ are indistinguishable. To complete the proof, we need to prove that $\mathcal{H}_{m+i}$ and $\mathcal{H}_{m+i+1}$ are indistinguishable for $i = 0, \dots, k-1$. This follows directly from the observation that $\mathcal{H}_{m+i}$ and $\mathcal{H}_{m+i+1}$ only differ in the witness used for x_{2m+i+1} as in $\mathcal{H}_{m+i}$ the witness from W^0 is used by $\mathcal{C}$ whereas in $\mathcal{H}_{m+i+1}$ $\mathcal{C}$ uses the witness from W^1. Indistinguishability follows directly from the Perfect WI of Π.

4 On Adaptive-Input Special-Soundness of Σ-Protocols

In this section we show that Σ-Protocols are not secure when the adversarial prover can choose the statement adaptively, when playing the 3rd round. These issues for the case of the Fiat-Shamir transform were noted in [1].

We then show an efficient compiler that on input a Σ-protocol belonging to the general class considered in [21,9], outputs a Σ-protocol that is secure against adaptively chosen statements.

4.1 Soundness Issues in Delayed-Input Σ-Protocols

We start by showing that the notion of adaptive-input special soundness is non-trivial in the sense that there are Σ-protocols that are not special sound when the statement is chosen adaptively at the 3rd round.

[10] See the full version of this work [7] for a formal description of the hybrid experiments.

Issues with soundness. Let us consider the following well-known Σ-protocol Π_{DH} for relation DH. On common input $T = (g, A, B, X)$ and private input α such that $A = g^{\alpha}$ and $X = B^{\alpha}$ for the prover, the following steps are executed. We denote by q the size of the group $\mathcal{G}$.

1. $\mathcal{P}$ picks $r \in \mathbb{Z}_q$ at random and computes and sends $a = g^r$, $x = B^r$ to $\mathcal{V}$;
2. $\mathcal{V}$ chooses a random challenge $c \in \mathbb{Z}_q$ and sends it to $\mathcal{P}$;
3. $\mathcal{P}$ computes and sends $z = r + c\alpha$ to $\mathcal{V}$;
4. $\mathcal{V}$ accepts if and only if: $g^z = a \cdot A^c$ and $B^z = x \cdot X^c$.

We now show that the above Σ-protocol is not special sound when an adversarial prover selects X adaptively.

Consider the following two conversations $((a = g^r, x = B^s), c_1, z_1 = r + \alpha \cdot c_1)$ and $((a = g^r, x = B^s), c_2, z_2 = r + \alpha \cdot c_2)$ respectively for tuples (g, A, B, X_1) and (g, A, B, X_2) where $A = g^{\alpha}$, $X_1 = g^{\gamma_1}$ and $X_2 = g^{\gamma_2}$ and $\gamma_i = \frac{z_i - s}{c_i} = \alpha + \frac{r-s}{c_i}$, for $i = 1, 2$. It is easy to see that both conversations are accepting (for their respective inputs) and that, if $r \neq s$, neither tuple is a DH tuple and therefore no witness can be extracted. Notice that this is a very strong soundness attack since the adversarial prover can succeed in convincing the verifier even though the statement is false. A similar argument can be used to prove that the Σ-protocol of [22] for relation $\mathsf{Com} = \{((g, h, G, H, m), r) : G = g^r \text{ and } H = h^{r+m}\}$ does not enjoy adaptive-input special soundness.

Issues with special soundness. Let us now consider the case of Schnorr's Σ-protocol [27] for relation $\mathsf{DLog} = \{((\mathcal{G}, g, Y), y) : g^y = Y\}$. Clearly, this is a different case since there is no false theorem to prove, but the attack can only consist in proving a statement violating special soundness (i.e., even though there are two accepting transcripts with the same first message no witness can be extracted).

In Schnorr's protocol, the prover on input $(Y, y) \in \mathsf{DLog}$ starts by sending $a = g^r$, for a randomly chosen $r \in Z_q$. Upon receiving challenge c, $\mathcal{P}$ replies by computing $z = r + yc$. $\mathcal{V}$ accepts (a, c, z) if $g^z = a \cdot Y^c$.

Consider now accepting transcripts (a, c_i, z_i) with respect to inputs Y_i, $i = 1, 2$. In this case, to extract witnesses y_i s.t. $((\mathcal{G}, g, Y_i), y_i) \in \mathsf{DLog}$ one has to solve the following system with unknowns r, y_1, and y_2.

$$\begin{cases} z_1 = r + c_1 \cdot y_1 \\ z_2 = r + c_2 \cdot y_2 \end{cases}$$

Clearly the system above has q solutions and thus it gives no information on any of the two witnesses.

4.2 A Compiler for Adaptive-Input Special Soundness

In this section we show how to upgrade special soundness to adaptive-input special-soundness in all Σ-protocols belonging to the interesting class of Σ-protocols proposed in [9,21].

We show a compiler that obtains a Σ-protocol Π_f^{a} for proving knowledge of the pre-image of a homomorphic function. Our compiler takes as input a Σ-protocol $\Pi_f = (\mathcal{P}_f, \mathcal{V}_f)$ for the same generic relation that includes Schnorr's [27], Guillou-Quisquater [16] and the Σ-protocol for DH tuples as special cases [9,21].

Let $(\mathcal{G}, \star)$ and $(\mathcal{H}, \otimes)$ be two groups with efficient operations and let $f : \mathcal{G} \rightarrow \mathcal{H}$ be a one-way homomorphism from $\mathcal{G}$ to $\mathcal{H}$. That is, for all $x, y \in \mathcal{G}$, we have that $f(x \star y) = f(x) \otimes f(y)$ and it is infeasible to compute w from $f(w)$ for a randomly chosen w. In protocol Π_f for relation $\mathcal{R}_f = \{(x, w) : x = f(w)\}$, prover and verifier receive as input a description of the groups $\mathcal{G}$ and $\mathcal{H}$ and $x \in \mathcal{H}$. The prover receives w such that $x = f(w)$ as a private input. The prover and verifier execute the following steps:

1. $\mathcal{P}_f$ picks $r \leftarrow \mathcal{G}$, sets $a \leftarrow f(r)$ and sends a to $\mathcal{V}_f$;
2. $\mathcal{V}_f$ randomly selects a challenge c and sends it to $\mathcal{P}_f$;
3. $\mathcal{P}_f$ on input r, x, w and c computes $z = r \star w^c$ and sends it to $\mathcal{V}_f$;
4. $\mathcal{V}_f$ accepts if and only if $f(z) = a \otimes x^c$.

It is easy to see that this protocol can be instantiated to give Schnorr's [27] and Guillou-Quisquater [16] Σ-protocols as special cases. Theorem 3 of [21] describes necessary conditions for Π_f to be special sound. Specifically, given a collision (a, c_1, z_1) and (a, c_2, z_2) for common input x, it is possible to extract w such that $x = f(w)$ if integer y and element $u \in \mathcal{G}$ are known and it holds that

1. $\gcd(c_1 - c_2, y) = 1$;
2. $f(u) = x^y$.

It is not difficult to see that this is the case when the protocol is instantiated for all the relations described above. We also observe that, since Schnorr's protocol is a special case of this protocol, protocol Π_f does not enjoy adaptive-input special soundness.

From Π_f to Π_f^a. We next show how to efficiently transform this Σ-protocol into one that enjoys adaptive-input special soundness. The underlying idea is that an adaptive attack against such protocol consists in misbehaving when playing the first round. For instance, in the case of the Σ-protocol for DH, $\mathcal{P}^*$ has to send a non-DH tuple in the 1st round while instead the protocol asks for a DH tuple. We therefore can convert Π_f into Π_f^{a} by asking the prover to also give an auxiliary proof where it proves knowledge of the randomness used to correctly compute the first round of Π_f. Notice that on this auxiliary proof an adaptive-input selection attack can not take place since the adversarial prover is stuck with the content of the 1st round of Π_f that therefore specifies already the statement to prove. We now show that special soundness allows to get the randomness used to compute the first round of Π_f. Then the same argument shown in Theorem 3 of [21] allows to extract the witness from a single transcript.

We now discuss the compiler and why it works more formally.

Let us start with the following observation. Consider an accepting transcript (a, c, z) of Π_f for input x. If r such that $f(r) = a$ is available, then it is possible to compute a witness w for x. Indeed, from Theorem 3 of [21] it follows that we can compute w as $w = u^\alpha \star (z \star r^{-1})^\beta$, where α and β are such that $y \cdot \alpha + c \cdot \beta = 1$[11]. We use an argument already used in Theorem 3 of [21] in order to prove that $f(w) = x$, where $w = u^\alpha \star (z \star r^{-1})^\beta$. First we observe that $f(z) = a \otimes x^c$ and this implies that $f(z \star r^{-1}) = x^c$. Then we observe (like in [21]) that $f(w) = f(u^\alpha \star (z \star r^{-1})^\beta) = f(u)^\alpha \otimes f(z \star r^{-1})^\beta = x^{y\alpha} + x^{\beta c} = x$ that proves that $f(w) = x$.

Consider protocol Π_f^{a} consisting of the parallel execution of two instances of Π_f. For common input x, the first instance of Π_f is executed on common input x, whereas in the second instance the common input is the first message a of the first instance. The verifier of Π_f^{a} sends the same challenge to both instances and accepts if and only if it accepts in both instances. Since in a collision the first message is fixed, both transcripts have the same first message a and therefore we can invoke special soundness to extract r such that $f(r) = a$. Once r is available, we apply the observation above and extract witnesses for x_1 and x_2 (the two inputs of the two 3-round transcripts constituting the collision). We have thus the following theorem.

Theorem 8. *If there exists a Σ-protocol Π_f for $\mathcal{R}_f$, then there exists a Σ-protocol Π_f^a for $\mathcal{R}_f$ that enjoys adaptive-input special soundness.*

5 On the Adaptive-Input Soundness of [6]'s Transform

Ciampi et al. in [6] show a compiler that takes as input two Σ-protocols, Π_0 and Π_1 for languages L_0 and L_1, and outputs a new Σ-protocol Π^{OR} for $L_0 \vee L_1$ in which the instance for the language L_1 is required by the prover only in the 3rd round. The compiler requires that Π_1 be delayed input and they show that the output of the compiler is a Σ-protocol, therefore it enjoys special soundness. In this section we assume that Π_1 is adaptive-input special sound, and we will show that Π^{OR} enjoys also adaptive-input special soundness.

5.1 Overview of the Construction of [6]

We start with a succinct description of the main building block used in of [6].

t-instance-dependent trapdoor commitment. Ciampi et al. in [6] define the notion of a t-Instance-Dependent Trapdoor Commitment (t-IDTC) scheme. Such a scheme works with respect to an polynomial-time relation $\mathcal{R}$. More formally, given the pair x, w s.t. $(x, w) \in \mathcal{R}$, it is possible to compute a commitment (with respect to some massage space M) using only the instance x, and the message. After that it is possible to open the commitment if one knows the randomness used in the commitment phase, or it is possible to equivocate the commitment using the witness w.

[11] By the first condition of Theorem 3 of [21] we have that $\gcd(c, y) = 1$ and thus α and β can be computed by using the extended Euclidean gcd algorithm.

The t-IDTC scheme is defined by a triple of PPT algorithm (TCom, TDec, TFake) where TCom, TDec are the honest commitment and decommitment procedures and TFake is the equivocation procedure that, given a witness for an instance x, equivocates any commitment computed using x as input of TCom. The properties of a t-IDTC scheme are: correctness, hiding, trapdoor and t-Special Extractability. The property of t-Special Extractability informally says that if the sender opens the same commitment in t different ways, then it is possible to efficiently extract the witness w. For more details see [6].

The authors of [6] show how to construct a 2-IDTC schemes are perfect hiding, perfect trapdoor and 2-Special Extractable from Σ-protocols.

In the rest of this section when a player runs the algorithm TCom on input x, m, obtains the pair ($\mathtt{com}$, $\mathtt{dec}$) where $\mathtt{com}$ is the commitment of the message m, and $\mathtt{dec}$ is the decommitment value. To check if $\mathtt{dec}$ is a valid decommitment of $\mathtt{com}$ with respect to the message m, we use the algorithm TDec. To compute a fake opening of the commitment $\mathtt{com}$ with respect to a message $m' \neq m$ a player can use the algorithm TFake using as input ($\mathtt{com}$, $\mathtt{dec}$).

The Construction of [6] Let $\mathcal{R}_0$ be a relation admitting a t-IDTC scheme, with $t = 2$ or $t = 3$. Let $\mathcal{R}_1$ be a relation admitting an delayed-input Σ-protocol Π_1 with associated simulator S^1.

We show a Σ-protocol $\Pi^{\mathsf{OR}} = (\mathcal{P}^{\mathsf{OR}}, \mathcal{V}^{\mathsf{OR}})$ for the OR relation:

$$\mathcal{R}^{\mathsf{OR}} = \Big\{((x_0, x_1), w) : ((x_0, w) \in \mathcal{R}_0 \wedge x_1 \in \hat{L}_{\mathcal{R}_1}) \ \mathsf{OR}\ ((x_1, w) \in \mathcal{R}_1 \wedge x_0 \in \hat{L}_{\mathcal{R}_0})\Big\}.$$

The initial common input is x_0 and the other input x_1 and the witness w for (x_0, x_1) are available to the prover only at the 3rd round. We let $b \in \{0, 1\}$ be such that $(x_b, w) \in \mathcal{R}_b$. The construction of [6] is described below.

Common input: $(x_0, 1^\lambda)$, where λ is the length of the instance of $\hat{L}_{\mathcal{R}_1}$.

```
1. $\mathcal{P}^{\mathsf{OR}}$ executes the following steps:
  1.1. pick random $r_1$ and compute the 1st round $a_1$ of the delayed-input $\Sigma$-
       protocol $\Pi_1$;
  1.2. compute a pair $(\mathtt{com}, \mathtt{dec}_1)$ of commitment and decommitment of $a_1$;
  1.3. send com to $\mathcal{V}^{\mathsf{OR}}$.
2. $\mathcal{V}^{\mathsf{OR}}$ sends a random challenge $c$.
3. $\mathcal{P}^{\mathsf{OR}}$ on input $((x_0, x_1), c, (w, b))$ s.t. $(x_b, w) \in \mathcal{R}_b$ executes the following
   steps:
  3.1. If $b = 1$,
           compute the 3rd round of $\Pi_1$, $z_1$, using as input $(x_1, w, c)$;
  3.2. send $(\mathtt{dec}_1, a_1, z_1)$ to $\mathcal{V}^{\mathsf{OR}}$r;
  3.3. If $b = 0$,
           run simulator $S^1$ on input $x_1$ and $c$ obtaining $(a_2, z_2)$;
           use trapdoor to compute decommitment $\mathtt{dec}_2$ of com as $a_2$;
  3.4. send $(\mathtt{dec}_2, a_2, z_2)$ to $\mathcal{V}^{\mathsf{OR}}$.
4. $\mathsf{V}^{\mathsf{OR}}$ accepts if and only if the following conditions are satisfied:
  4.1. $(a, c, z)$ is an accepting conversation for $x_1$;
  4.2. dec is a valid decommitment of com for a message $a$.
```

5.2 Adaptive-Input Security of Π^{OR}

We now show that Π^{OR} preserves the adaptive-input special soundness of the underlying Σ-protocol.

Theorem 9. *If $\mathcal{R}_0$ admits a 2-IDTC and $\mathcal{R}_1$ admits a delayed-input adaptive-input special-sound Σ-protocol, then Π^{OR} is an adaptive-input special-sound Σ-protocol.*

Proof. The claim follows from the adaptive-input special soundness of the underlying Σ-protocol Π_1 and from the 2-Special Extractability property of the 2-IDTC scheme. More formally, consider an accepting transcript $(\mathtt{com}, c, (z, a, \mathtt{dec}))$ for input (x_0, x_1) and an accepting transcript $(\mathtt{com}, c', (z', a', \mathtt{dec}'))$ for input (x_0, x_1'), where $c' \neq c$ and x_1 is potentially different from x_1'. We observe that:

- if $a = a'$ then by the property of adaptive-input special soundness of Π_1 there exists an efficient extractor $\mathsf{AExtract}$ that, given as input $((a, c, z), x_1)$ and $((a', c', z'), x_1')$, outputs w_1 and w_1' s.t. $(x_1, w_1) \in \mathcal{R}_1$ and $(x_1', w_1') \in \mathcal{R}_1$;
- if $a \neq a'$, then $\mathtt{dec}$ and $\mathtt{dec}'$ are two openings of $\mathtt{com}$ with respect to x_0 for messages $a \neq a'$; then we can obtain a witness w_0 by the 2-Special Extractability of the 2-IDTC scheme.

A similar arguments can be used to show that if $\mathcal{R}_0$ admits a 3-IDTC and $\mathcal{R}_1$ admits a delayed-input Σ-protocol with adaptive-input special soundness, then Π^{OR} enjoys the adaptive-input proof of knowledge property.

6 Extension to Multiple Relations

In this section, we generalize the result of Section 3 to the case of different relations. More specifically, given delayed-input Σ-protocols $\Pi_1, \ldots, \Pi_n$ for polynomial-time relations $\mathcal{R}_1, \ldots, \mathcal{R}_n$, we construct, for some positive constant k, Adaptive-Input Proof of Partial Knowledge Γ for the *threshold* polynomial-time relation

$$\mathcal{R}^{\mathsf{thres}} = \Big\{ \Big((x_1, \ldots, x_n, k), \big((w_1, d_1) \ldots, (w_k, d_k)\big) \Big) : 1 \leq d_1 < \cdots < d_k \leq n$$
$$\text{and } (x_{d_i}, w_i) \in \mathcal{R}_i \text{ for } i = 1, \ldots, k \text{ and } x_1 \in \hat{L}_1, \ldots, x_n \in \hat{L}_n \Big\}.$$

We remind the reader that $\hat{L}_1, \ldots, \hat{L}_n$ are the input languages associated with the polynomial-time relations $\mathcal{R}_1, \ldots, \mathcal{R}_n$.

Protocol Γ uses delayed-input protocol Π^k, in the adaptive-input special-soundness version, presented in Section 3 for relation $\mathsf{NDH}_{k,n}$. We remark that protocol $\Pi^{\mathrm{ddh}}_{k,n}$ of Section 2.3 would not work here since the prover of Γ learns the actual statement to be proved just before the third round.

1st round. $\Gamma.\mathsf{Prover} \Rightarrow \Gamma.\mathsf{Verifier}$:
$\Gamma.\mathsf{Prover}$ receives as unary inputs the security parameter λ, the number n of theorems that will be given as input at the beginning of the third round, and the number k of witnesses that will be provided.

1. Set $(\mathcal{G}, p, g) \leftarrow \mathsf{IG}(1^\lambda)$.
2. For $j = 1, \ldots, n$
   2.1. Randomly sample a non-DH tuple $T_j^0 = (g_j, A_j, B_j, X_j)$ over $\mathcal{G}$, along with α_j such that $A_j = g_j^{\alpha_j}$.
   2.2. Set $Y_j = B_j^{\alpha_j}$ and $T_j^1 = (g_j, A_j, B_j, Y_j)$ (note that the quadruple T_j^1 is by construction a DH tuple).
3. Select a random string $R_{k,n}$ and use it to compute the first round message $a_{k,n}$ of Π^k by running prover $\mathcal{P}^k$.
   Send $a_{k,n}$ to $\Gamma.\mathsf{Verifier}$.
4. For $j = 1, \ldots, n$
   4.1. Select random strings R_j^0 and R_j^1 and use them to compute the first rounds a_j^0 and a_j^1 of Π_j by running prover $\mathcal{P}_j$.
   4.2. Compute the pair $(\mathtt{com}_j^0, \mathtt{dec}_j^0)$ of commitment and decommitment of the message a_j^0 using non-DH tuple T_j^0.
   4.3. Compute the commitment $\mathtt{com}_j^1$ (of the message a_j^1) using the DH tuple T_j^1.
   4.4. Send pairs $(T_j^0, \mathtt{com}_j^0)$ and $(T_j^1, \mathtt{com}_j^1)$ in random order to $\Gamma.\mathsf{Verifier}$.

2nd round. $\Gamma.\mathsf{Verifier} \Rightarrow \Gamma.\mathsf{Prover}$: $\Gamma.\mathsf{Verifier}$ randomly selects a challenge c and sends it to $\Gamma.\mathsf{Prover}$.

3rd round. $\Gamma.\mathsf{Prover} \Rightarrow \Gamma.\mathsf{Verifier}$:
$\Gamma.\mathsf{Prover}$ receives theorems $x_1, \ldots, x_n$ and, for $d_1 < \ldots \leq d_k$, witnesses $w_1, \ldots, w_k$ for theorems $x_{d_1}, \ldots, x_{d_k}$, respectively. We let $\widetilde{d}_1 < \ldots < \widetilde{d}_{n-k}$ denote the indices of the theorems for which no witness has been provided.

1. For $l = 1, \ldots, k$
   1.1. Use j as a shorthand for d_l.
   1.2. Set $U_j = T_j^1$ and $\hat{U}_j = T_j^0$.
   1.3. Compute third round z_j of Π_j by running prover $\mathcal{P}_j$ on input (x_j, w_i), randomness R_j^0 used to compute the first round a_j^0, and a challenge c.
   1.4. Set $M_j = (a_j^0, z_j, \mathtt{dec}_j^0, \hat{U}_j)$.
2. For $l = 1, \ldots, n-k$
   2.1. Set $j = \widetilde{d}_l$.
   2.2. Set $U_j = T_j^0$ and $\hat{U}_j = T_j^1$.
   2.3. Run the simulator S_j of Π_j on input x_j and c therefore obtaining $(\widetilde{a}_j^1, z_j)$.
   2.4. Use the trapdoor α_j to compute the decommitment $\mathtt{dec}_j^1$ of $\mathtt{com}_j^1$ as $\widetilde{a}_j^1$.
   2.5. Set $M_j = (\widetilde{a}_j^1, z_j, \mathtt{dec}_j^1, \hat{U}_j)$.
3. For $l = 1, \ldots, n$ send M_l to $\Gamma.\mathsf{Verifier}$.
4. Compute the third round $z_{k,n}$ of Π^k by running prover $\mathcal{P}^k$ of Π^k on input tuples $(U_1, \ldots, U_n)$, witnesses $\alpha_{d_1}, \ldots, \alpha_{d_k}$ and randomness $R_{k,n}$ used to compute the first round $a_{k,n}$.

$\Gamma.\mathsf{Verifier}$ accepts if and only if the following conditions are satisfied.

1. Check that $(a_{n,k}, c, z_{n,k})$ is an accepting conversation for $\mathcal{V}^k$ for input $U_1, \ldots, U_n$.

2. For $i = 1 \ldots n$
   Check that tuples T_i^0 and T_i^1 differ only in the last component.
   Check that $\{U_i, \hat{U}_i\} = \{T_i^0, T_i^1\}$.
   Write M_i as $M_i = (a_i, z_i, \mathtt{dec}_i, \hat{U}_i)$.
   Check that $\mathtt{dec}_i$ is a decommitment of one of $\mathtt{com}_i^0$ and $\mathtt{com}_i^1$ as a_i with respect to tuple $\hat{U}_i$.
   Check that (a_i, c, z_i) is an accepting conversation for Π_i on input x_i.

Theorem 10. *Γ is a proof of knowledge.*

Proof. The completeness property follows from the completeness of protocols Π^k and Π_i, for $i \in \{1, \ldots, n\}$, and from the correctness and trapdoorness property of the Instance-Dependent Trapdoor Commitment scheme used.

Now we proceed by proving that our protocol is $(2n + k)$-special sound and then, using the arguments of Section 2 about the proof of knowledge property of protocols that enjoy t-special soundness, we can conclude the proof claiming that Γ is a proof of knowledge. In more details, we prove that there exists an efficient extractor which, for any sequence $(x_1, \ldots, x_n)$ of n inputs and for any set of $2n+k$ accepting conversations of Γ that share the same first message and have different challenges, outputs the witness of w_i s.t. $(x_i, w_i) \in \mathcal{R}_i$ for some $i \in \{1, \ldots, n\}$. The extractor considers a set of $2n + k$ accepting conversations a, c^j, z^j (with $j = 1, \ldots, 2n+k$) such that they share the same first message and have different challenges. For each a, c^j, z^j (with $j = 1, \ldots, 2n+k$) processed by the extractor one of the following two cases is possible.

1. There are two conversations of Σ-protocol Π^i for theorem x_i that share the same first message a_i and have two different challenges. Then by the special soundness property of Π^i one can efficiently get a witness w_i for theorem x_i.
2. If the new accepting transcript a, c^j, z^j does not allow the extractor to obtain the witness then a new non-DH tuple is used for the first time in the accepting conversation a, c^j, z^j.

The proof ends with the observation that the algorithm stops after k times that the first case occurs, while the second case occurs at most $2n$ times.

Theorem 11. *Under the DDH assumption, if Π_i is SHVZK for $\mathcal{R}_i$, for $i \in \{1, \ldots, n\}$, then Γ is adaptive-input WI for $\mathcal{R}^{\mathsf{thres}}$, for a constant k.*

Proof sketch. The definition of adaptive-input WI gives to the adversary $\mathcal{A}$ the power to choose both theorems and witnesses upon receiving the first message from the challenger. This implies that in Γ the first round should be computed without knowing which witnesses will be chosen by $\mathcal{A}$, and without knowing for what instances the witnesses will be available in the third round. It is easy to see that the first round of Γ is independent from the $\mathcal{A}$ could have. Unfortunately if we follow the same proof of Theorem 7, considering a similar sequence of hybrids experiments, we will have to define hybrid experiments in which the first round depends on which witnesses will be received from $\mathcal{A}$ at the second round. This implies that the only way for the challenger to complete these hybrid experiments consists in guessing the instances that correspond to the witnesses that will be received. This explains why k is a constant.

We now explain with more details the differences between the security proof of Theorem 7 and the one needed for protocol Γ. The security proof of Theorem 7 works for every k because the n instances $x_1, \ldots, x_n$ that will be sent by $\mathcal{A}$ in the protocol Π^k belong to the same NP-language L. Furthermore the Σ-protocol Π used in Π^k is delayed input. Hence for a first round a_i of Π it is possible to create an accepting transcript (a_i, c, z) for a theorem x_j, for $i, j \in \{1, \ldots, n\}$ (if one has the witness w_j clearly). Therefore the assignment of the values $a_1, \ldots, a_n$ committed in the first round with the theorems $x_1, \ldots, x_n$ is made only at the third round. This property holds in all hybrid experiments. Now we consider the protocol Γ. The arguments described above are clearly not applicable to prove that Γ is adaptive-input WI.

More in details, during the first round of Γ, for each language L_i we compute the first message of protocol Π_i and commit to it twice using the instance-dependent trapdoor commitment associated to a DH tuple and to a non-DH tuple, for $i \in \{1, \ldots, n\}$. Hence for each a_i we compute an equivocal commitment and a binding commitment. First note that these two commitments are linked to a fixed language L_i (in contrast to the first round of Π^k). When in the security proof we need to consider the hybrid experiment in which $n+1$ non-DH tuples (one non-DH tuple per pair except one pair where both tuples are non-DH) are used (as in the proof of Theorem 7), we have to commit the first round of a_i, for some $i \in \{1, \ldots, n\}$, using two commitments that are perfectly binding. Therefore the only way that we have to compute an accepting transcript with respect to the language L_i consists in using the witness for the instance x_i that will be sent by $\mathcal{A}$. Unfortunately we have no guarantee that $\mathcal{A}$ will send w_i, and thus the experiment will have to try again. For lack of space, further details can be found in the full version of this work.

7 Acknowledgments

We thank the anonymous reviewers of Eurocrypt 2016 for many insightful comments and suggestions. This work has been supported in part by "GNCS - INdAM" and in part by the EU COST Action IC1306.

References

1. Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In: Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings. pp. 626–643 (2012)
2. Blundo, C., Persiano, G., Sadeghi, A., Visconti, I.: Improved security notions and protocols for non-transferable identification. In: Computer Security - ESORICS

2008, 13th European Symposium on Research in Computer Security, Málaga, Spain, October 6-8, 2008. Proceedings. pp. 364–378 (2008)

3. Catalano, D., Dodis, Y., Visconti, I.: Mercurial commitments: Minimal assumptions and efficient constructions. In: Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings. pp. 120–144 (2006)
4. Catalano, D., Visconti, I.: Hybrid trapdoor commitments and their applications. In: Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings. pp. 298–310 (2005)
5. Catalano, D., Visconti, I.: Hybrid commitments and their applications to zero-knowledge proof systems. Theor. Comput. Sci. 374(1-3), 229–260 (2007)
6. Ciampi, M., Persiano, G., Scafuro, A., Siniscalchi, L., Visconti, I.: Improved or-composition of sigma-protocols. In: Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II. pp. 112–141 (2016)
7. Ciampi, M., Persiano, G., Scafuro, A., Siniscalchi, L., Visconti, I.: Online/offline or composition of sigma protocols. Cryptology ePrint Archive, Report 2016/175 (2016), http://eprint.iacr.org/
8. Ciampi, M., Persiano, G., Siniscalchi, L., Visconti, I.: A transform for NIZK almost as efficient and general as the fiat-shamir transform without programmable random oracles. In: Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II. pp. 83–111 (2016)
9. Cramer, R., Damgård, I.: Zero-knowledge proofs for finite field arithmetic; or: Can zero-knowledge be for free? In: Krawczyk, H. (ed.) Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings. Lecture Notes in Computer Science, vol. 1462, pp. 424–441. Springer (1998)
10. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y. (ed.) Advances in Cryptology — CRYPTO '94, Lecture Notes in Computer Science, vol. 839, pp. 174–187. Springer Berlin Heidelberg (1994)
11. Damgård, I.: On Σ-protocol. http://www.cs.au.dk/~ivan/Sigma.pdf (2010)
12. Damgård, I., Groth, J.: Non-interactive and reusable non-malleable commitment schemes. In: Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA. pp. 426–437 (2003)
13. Damgård, I., Nielsen, J.B.: Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In: Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings. pp. 581–596 (2002)
14. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings. pp. 186–194 (1986)
15. Garay, J.A., MacKenzie, P., Yang, K.: Strengthening zero-knowledge protocols using signatures. Journal of Cryptology 19(2), 169–209 (2006)
16. Guillou, L.C., Quisquater, J.: A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In: Günther, C.G. (ed.) Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings. Lecture Notes in Computer Science, vol. 330, pp. 123–128. Springer (1988)

17. Hazay, C., Lindell, Y.: Efficient Secure Two-Party Protocols - Techniques and Constructions. Information Security and Cryptography, Springer (2010)
18. Katz, J., Ostrovsky, R.: Round-optimal secure two-party computation. In: Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology-Conference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings. pp. 335–354 (2004)
19. Lapidot, D., Shamir, A.: Publicly verifiable non-interactive zero-knowledge proofs. In: Advances in Cryptology - CRYPTO (1990)
20. Lindell, Y.: An efficient transform from sigma protocols to NIZK with a CRS and non-programmable random oracle. In: Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I. pp. 93–109 (2015)
21. Maurer, U.: Zero-knowledge proofs of knowledge for group homomorphisms. Designs, Codes and Cryptography pp. 1–14 (2015)
22. Micciancio, D., Petrank, E.: Simulatable commitments and efficient concurrent zero-knowledge. In: Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings. pp. 140–159 (2003)
23. Ostrovsky, R., Pandey, O., Visconti, I.: Efficiency preserving transformations for concurrent non-malleable zero knowledge. In: Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings. pp. 535–552 (2010)
24. Ostrovsky, R., Richelson, S., Scafuro, A.: Round-optimal black-box two-party computation. In: Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II. pp. 339–358 (2015)
25. Pass, R.: Simulation in quasi-polynomial time, and its application to protocol composition. In: Biham, E. (ed.) Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings. Lecture Notes in Computer Science, vol. 2656, pp. 160–176. Springer (2003)
26. Santis, A.D., Crescenzo, G.D., Persiano, G., Yung, M.: On monotone formula closure of SZK. In: 35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994. pp. 454–465 (1994)
27. Schnorr, C.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) Advances in Cryptology — CRYPTO' 89 Proceedings, Lecture Notes in Computer Science, vol. 435, pp. 239–252. Springer New York (1989)
28. Visconti, I.: Efficient zero knowledge on the internet. In: Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II. pp. 22–33 (2006)
29. Yung, M., Zhao, Y.: Generic and practical resettable zero-knowledge in the bare public-key model. In: Advances in Cryptology - EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20-24, 2007, Proceedings. pp. 129–147 (2007)