Online/Offline OR Composition of Sigma Protocols

Online/Offline OR Composition of Sigma Protocols

Michele Ciampi
DIEM
Università di Salerno
ITALY
mciampi@unisa.it

Giuseppe Persiano
DISA-MIS
Università di Salerno
ITALY
giuper@gmail.com

Alessandra Scafuro
Boston University and
Northeastern University
USA
scafuro@bu.edu

Luisa Siniscalchi
DIEM
Università di Salerno
ITALY
lsiniscalchi@unisa.it

Ivan Visconti
DIEM
Università di Salerno
ITALY
visconti@unisa.it

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 [CDS94] 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. [CPS+16a] 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 [LS90].

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 [Mau15, CD98]. 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 [CDS94].

Our new transform relies on the DDH assumption (in contrast to the transforms of [CDS94, CPS+16a] that are unconditional). We also show how to strengthen the transform of [CPS+16a] so that it also achieves adaptive soundness, when the underlying combined protocols belong to the class of protocols described in [Mau15, CD98].

Contents

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 essentially every 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, the existence of very efficient WI PoKs for specific languages such as Discrete Log and DDH has been instrumental for constructing efficient maliciously secure two-party computation (see [HL10] and reference within). Round-efficient protocols [Pas03, KO04] require security notions, both with respect to a malicious prover (soundness) and with respect to a malicious verifier (WI), that hold even in presence of *adaptive-input* selection.

Proofs of partial knowledge. In [CDS94], Cramer et al. showed that one can use a specific type of PoK (called a Σ-protocol) to construct an *efficient* PoK for a compound statement. More precisely, the compound statement consists of n instances, and the goal is to prove knowledge of a witness for at least k of the n instances. As such, these proofs are named "*proofs of partial knowledge*" in [CDS94]. The transform of [CDS94] 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 [DSDCPY94] for perfect ZK.

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) [LS90][1]. Proving compound statements via LaSh 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 [CDS94], 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 improved efficiency of [CDS94] must be paid for by the additional rounds need by the larger protocol that uses the PoK as a building block.

The proof of partial knowledge of [CPS+16a]. A very recent work by Ciampi et al. [CPS+16a] makes a first preliminary step towards closing the gap between [LS90] and [CDS94]. [CPS+16a] proposes a different transform for WI proofs of partial knowledge that gives some adaptivity 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 "$x_0 \in L_0 \vee x_1 \in L_1$" in which x_0 and x_1 are not immediately needed (in contrast to [CDS94]). 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 one of x_0 and x_1. The verifier needs to see x_0 and x_1 only at the end, in order to accept/reject the proof. These PoKs are called delayed input in [CPS+16a] as the need of the input is delayed to the very last round for the honest prover. For

[1] See [OV12] for a detailed description of [LS90].

clarity, we stress that a delayed-input protocol is not necessarily secure against inputs that have been adaptively chosen. Indeed, the technique of [CPS+16a] yields a proof of partial knowledge that is delayed input for one of the two instances, is adaptive-input WI but it is not adaptively secure against a malicious prover. The security achieved 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 [Sch89], for proving knowledge of one discrete log. Schnorr's protocol is also delayed-input property and the prover needs not to know the instance g^x in order to compute the first round. However, when we apply known transforms, the resulting protocol loses the delayed-input property. More specifically, both instances g^{x_0} and g^{x_1} are need by [CDS94], and at least one g^{x_0} by [CPS+16a].

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-protocol enjoys a special soundness property, which means that, given two accepting transcripts[2] for the same statement having the same first round, one can efficiently extract a witness for that statement.

We show that delayed-input Σ-protocols are not necessarily 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 the most commonly used Σ-protocols, such as Schnorr's protocol for discrete logarithm, the protocol for Diffie-Hellman (DH) tuples and the protocol of [MP03] for proving knowledge of committed messages, and to all Σ-protocols in the well known class proposed by Cramer in [CD98] and Maurer in [Mau15].

The loss of soundness with respect to provers that adaptively choose their inputs was already noticed in [BPW12] for non-interactive zero-knowledge arguments obtained from Σ-protocols by means of the Fiat-Shamir transform [FS86]. Indeed there are in the literature some incorrect uses of the Fiat-Shamir transform in which 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 can be addressed by giving also 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 [CPS+16a] that is delayed-input with respect to one instance only. We observe that when [CPS+16a] combines protocols belonging to the class of [CD98, Mau15],

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

it is not secure with respect to a malicious prover that is allowed to adaptively choose his input. Therefore the transform of [CPS+16a] is not adaptive-input sound. We stress however, that in the applications targeted in [CPS+16a] 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, [CPS+16a] 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 adaptivity considered in [CPS+16a] by formally defining adaptive-input special soundness. Then we show a compiler that takes as input any delayed-input Σ-protocol belonging to the class specified in [CD98, Mau15], 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 correctly send 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, [CPS+16a]'s transform preserves the adaptivity of the Σ-protocols that are combined. Namely, when applied to Σ-protocols that are already adaptive-input special sound and WI, [CPS+16a]'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.

1.1.1 Our Technique

We provide a technique for composing a delayed-input Σ-protocol for a relation $\mathcal{R}$ in an delayed-input Σ-protocol for the (k, n)-proof of partial knowledge for relation $(\mathcal{R} \vee \ldots \vee \mathcal{R})$.

For better understanding our technique, it is instructive to see why the previous transformation [CDS94] (resp., [CPS+16a]) 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 [CDS94] 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 [CDS94] 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 [CDS94] 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 [CDS94] 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 [CPS+16a] 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 $\mathsf{binding}$ and the remaining $n-k$ are $\mathsf{equivocal}$, 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 was constructed in [ORS15]). With this gadget in hand, we can construct a delayed-input (k, n)-proof of partial knowledge $\Sigma_{k,n}^{\mathsf{OR}}$ 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 $a_1, \ldots, a_n$ using the (k, n)-equivocal commitment scheme discussed above, where the k binding positions are randomly chosen. Thus, the 1st round of protocol $\Sigma_{k,n}^{\mathsf{OR}}$ 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 $x_1, \ldots, x_n$ 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 $\mathtt{com}_1, \ldots, \mathtt{com}_n$ such that k of them are binding and $n-k$ are equivocal. We will use the language DH of DH tuples and we will implement a (k,n)-equivocal commitment scheme very efficiently under the DDH assumption as follows. In the commitment phase, the sender computes n tuples $T_1 = (g_1, A_1, B_1, X_1), \ldots, T_n = (g_n, A_n, B_n, X_n)$ and proves that k out of n tuples are *not* in DH. We show that this can be done using the classical [CDS94] (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 [DG03, CV05, CV07, HL10] 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 [CDS94] 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 $\Sigma_1, \ldots, \Sigma_n$ 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 $x_1, \ldots, x_n$ 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.

1.2 Comparison with the State of the Art

In Table 1 we compare our results with the relevant related work. We consider [LS90], a 3-round public-coin WIPoK that is *fully adaptive-input* and that works for any NP language. We also consider [CDS94] that proposed efficient 3-round public-coin WI proofs of partial knowledge (though, without supporting any adaptivity). Finally, we consider [CPS+16a] 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.

	Assumption	Adaptive WI	Adaptive PoK	NP Reduction
LaSh90 [LS90]	OWP	k out of n (all adaptive)	k out of n (all adaptive)	Yes
CDS94 [CDS94]	/	/	/	No
CPSSV16 [CPS+16a]	/	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.

The 2nd column refers to the computational assumptions needed by [LS90] (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.

1.3 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 [CDS94] requires to know the instances already to compute the first round. Furthermore the work of [LS90] allows the prover to compute the first round offline but in the online phase the prover must perform an NP reduction.

In Table 2[3] we compare the effort of the prover in the online phase in our work and in [CDS94, LS90]. 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 [LS90] the prover computes an NP reduction (2nd row). For our construction and the one of [CDS94] we count the number of modular[4] exponentiations that are computed in the online phase (3rd and 4th rows).

[3]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 certainly more expensive than their non-adaptive counterparts.

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

Below we briefly describe how we have computed the above costs. In [CDS94] 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 [CDS94] the simulator is executed $2(n - k)$ times and moreover k exponentiations are needed to run the prover of Schnorr's protocol.

	$(1, 2)$ DLogs	(k, n) DLogs
LaSh90 [LS90]	NP-reduction	NP-reduction
CDS94 [CDS94]	3 exps	$2n - k$ exps
CPSSV16 [CPS+16a]	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.

In the (1,2)-proof of partial knowledge of [CPS+16a], the 1st round requires 3 exponentiations. Indeed, in the 1st round of [CPS+16a] the prover runs Schnorr's simulator and computes the 1st round of Schnorr's Σ-protocol. The 3rd round of [CPS+16a] has a different analysis depending on which witness is used. When the prover of [CPS+16a] 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 [CPS+16a] costs two exponentiations. Note that in the execution of the construction of [CPS+16a] 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 [CDS94] 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 [CDS94] since the correctness of the pair of tuples can be self-verified.

2 Preliminaries

We use $\mathbb{N}$ to denote the set of all natural numbers. For a probabilistic algorithm A, $A(x)$ denotes the probability distribution of the output of A when run with x as input. We use $A(x;r)$ instead to denote the output of A when run on input x and coin tosses r. We let PPT stand for probabilistic polynomial time.

A *polynomial-time* relation $\mathcal{R}$ is a relation for which membership of (x,w) to $\mathcal{R}$ can be decided in time polynomial in $|x|$. If $(x,w) \in \mathcal{R}$ then we say that w is a *witness* for *instance* x. A polynomial-time relation $\mathcal{R}$ is naturally associated with the NP language $L_{\mathcal{R}}$ defined as $L_{\mathcal{R}} = \{x | \exists w : (x,w) \in \mathcal{R}\}$. Similarly, an NP language is naturally associated with a polynomial-time relation. Following [GMY06], 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. 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 a proof system for $\mathcal{R}$, the verifier runs the protocol only if the common input x belongs to $\hat{L}_{\mathcal{R}}$ and it immediately rejects 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 and interacting with M_0 running on input x_0 and 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 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 $\mathcal{V}$'s messages consist of his coin tosses.*

Definition 3 ([Dam10])**.** *Let $k : \{0,1\}^* \to [0,1]$ be a function. A complete protocol $(\mathcal{P},\mathcal{V})$ is a* proof of knowledge *for the relation $\mathcal{R}$ with* knowledge error *k if the property of* Knowledge soundness *is satisfied.*

- Knowledge soundness: *there exists a constant $c > 0$ and a probabilistic oracle machine* $\mathsf{Extract}$*, called the* extractor*, such that for every interactive prover $\mathcal{P}^\star$ and every input x, the machine* $\mathsf{Extract}$ *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* $\mathsf{Extract}$ *outputs a string w such that $(x,w) \in \mathcal{R}$ within an expected number of steps bounded by $|x|^c/(\epsilon(x) - k(x))$.*

Σ-protocols. Here we introduce a special 3-round public coin protocol widley used in practice: Σ-protocols. Here the verifier sends one single message, called the *challenge*. There exist efficient Σ-protocols for useful languages, and moreover they are easy to work with as already shown in many transforms [DG03, MP03, Vis06, CDV06, BPSV08, YZ07, OPV10, Lin15, CPSV16].

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 by $\mathcal{P}$ 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 $c \neq c'$.

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

- *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 (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 transcript for x. Moreover, the distribution of the output of S on input (x, c) is perfectly[5] 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 $(x, w) \in \mathcal{R}$.*

SHVZK is a weaker requirement than Zero Knowledge but it nonetheless implies non-trivial security against adversarial verifiers.

Theorem 1 ([CDS94])**.** *Every Σ-protocol is Perfect WI.*

The special soundness property makes the challenge length act as a security parameter for a Σ-protocol, in the sense that a Σ-protocol with sufficiently long challenges is also a proof system. More formally, we have.

Theorem 2. *[CDS94, Dam10] Let Π be a Σ-protocol for relation $\mathcal{R}$ and challenge length l. Running Π k-times in parallel for the same instance x corresponds to running Σ-protocol for $\mathcal{R}$ with challenge length $k \cdot l$.*

Following the Theorem 1 of [Dam10] and its security proof we can claim the following theorem.

Theorem 3. *Let Π be a 3-round public-coin protocol that enjoys the property of completeness and that is special sound for a relation $\mathcal{R}$. If Π has challenge length of dimension λ, then Π is a proof of knowledge with knowledge error $2^{-\lambda}$.*

By the above theorem, we have that every Σ-protocol with a sufficiently long challenge is a proof of knowledge with negligible knowledge error.

In this paper we consider a relaxed notion of special soundness, called *t-special soundness*, by which $t = \texttt{poly}(|x|)$ transcripts (with $t > 2$) allow to extract a witness for x. By using a proof similar to the one of Theorem 3, we can prove that a t-special sound protocol is a proof of knowledge with negligible soundness error when the challenge is sufficiently long.

[5] In this work we stick with the requirement of perfect SHVZK for Σ-protocols. *Computational* special HVZK has also been considered in the literature.

2.1 Adaptive-Input Special Soundness and Proof of Knowledge

In this paper we make use of Σ-protocols that enjoys a stronger definition of completeness: *delayed-input completeness*. In a Σ-protocol that enjoys this property the prover does not need to know both theorem and witness in order to output the first round of the protocol.

Definition 5 (Delayed-Input Σ-protocol [CPS+16a])**.** *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 $\ell = |x|$.*[6]

In this paper we also consider a stronger notion of special-soundness and PoK that could be enjoyed by a Σ-protocol that is delayed-input.

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 differences that Π has to be delayed-input, and 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

[6] 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.

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* $\mathsf{AdvAWI}_{\Pi,\mathcal{A}}(\lambda, \mathsf{aux}) \leq \nu(\lambda)$.

2.3 The DDH assumption

Let $\mathcal{G}$ be a cyclic group, g generator of $\mathcal{G}$ and let A, B and X be elements of $\mathcal{G}$. We say that (g, A, B, X) is a *Diffie-Hellman tuple* (a *DH tuple*, in short) if $A = g^\alpha, B = g^\beta$ for some integers $0 \leq \alpha, \beta \leq |\mathcal{G}| - 1$ and $X = g^{\alpha\beta}$. If this is not the case, the tuple is called *non-DH*. To verify that a tuple is DH, it is sufficient to have the discrete log α of A to the base g and then to check that $X = B^\alpha$. We thus define the polynomial-time relation $\mathsf{DH} = \{((g, A, B, X), \alpha) : A = g^\alpha \text{ and } X = B^\alpha\}$ of the DH tuples.

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. For sake of concreteness, we consider the specific group generator GG 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 $\mathcal{G}$.

Assumption 1 (DDH Assumption)**.** *For every probabilistic polynomial-time algorithm* $\mathcal{A}$ *there exists a negligible function* ν *s.t.*

$$\Big|\text{Prob}\left[\ (\mathcal{G}, q, g) \leftarrow \mathsf{GG}(1^\lambda); \alpha, \beta, \gamma \leftarrow \mathbb{Z}_q : \mathcal{A}((\mathcal{G}, q, g), g^\alpha, g^\beta, g^\gamma) = 1\ \right] -$$
$$\text{Prob}\left[\ (\mathcal{G}, q, g) \leftarrow \mathsf{GG}(1^\lambda); \alpha, \beta, \gamma \leftarrow \mathbb{Z}_q : \mathcal{A}((\mathcal{G}, q, g), g^\alpha, g^\beta, g^{\alpha\beta}) = 1\ \right]\Big| \leq \nu(\lambda).$$

In our construction we will need the concept of a oneNDH tuple; that is, a tuple $T = (g, A, B, X)$ such that $A = g^\alpha$, $B = g^\beta$ and $X = g^{\alpha\beta+1}$. It is easy to see that, under the DDH assumption, randomly selected oneNDH tuples are indistinguishable from randomly selected non-DH tuples. Indeed, let $T = (g, A, B, X)$ be any tuple and consider tuple $T' = (g, A, B, X \cdot g)$. Then we have that if T is a randomly selected DH tuple, then T' is a randomly selected oneNDH tuple; whereas, if T is a randomly selected non-DH tuple then T' is statistically close to a randomly selected non-DH

tuple. Moreover, by transitivity, we have that, under the DDH assumption, randomly selected oneNDH tuples are indistinguishable from randomly selected DH tuples. We can thus state the following lemma.

Lemma 1. *Under the DDH assumption, for every probabilistic polynomial-time algorithm $\mathcal{A}$ there exists a negligible function ν s.t.*

$$\Big|\text{Prob}\Big[\ (\mathcal{G},q,g)\leftarrow \mathsf{GG}(1^\lambda);\alpha,\beta\leftarrow\mathbb{Z}_q:\mathcal{A}((\mathcal{G},q,g),g^\alpha,g^\beta,g^{\alpha\beta+1})=1\ \Big]-$$
$$\text{Prob}\Big[\ (\mathcal{G},q,g)\leftarrow \mathsf{GG}(1^\lambda);\alpha,\beta\leftarrow\mathbb{Z}_q:\mathcal{A}((\mathcal{G},q,g),g^\alpha,g^\beta,g^{\alpha\beta})=1\ \Big]\Big|\leq\nu(\lambda).$$

We next describe a Σ-protocol Π^{1ndh} for proving that at least k out of n given tuples are oneNDH. The Σ-protocol is based on the perfect WI Σ-protocol Π^{ddh} of [CDS94] for proving that at least k of n given tuples are DH. We stress that the Σ-protocol Π^{1ndh} that we construct in this section is also perfect WI.

Formally, we construct Σ-protocol $\Pi_k^{\mathsf{1ndh}}=(\mathcal{P}_k^{\mathsf{1ndh}},\mathcal{V}_k^{\mathsf{1ndh}})$ for the polynomial-time relation

$$\mathsf{1ndh}_k=\Big\{\Big(\big((g_1,A_1,B_1,X_1),\ldots,(g_n,A_n,B_n,X_n)\big),((\alpha_{b_1},b_1)\ldots,(\alpha_{b_k},b_k))\Big):$$
$$1\leq b_1<\cdots<b_k\leq n \text{ and } A_{b_i}=g_{b_i}^{\alpha_{b_i}} \text{ and } X_{b_i}=g_{b_i}\cdot B_{b_i}^{\alpha_{b_i}},\ \text{for } i=1,\ldots,k\Big\}.$$

$\mathcal{P}_k^{\mathsf{1ndh}}$ and $\mathcal{V}_k^{\mathsf{1ndh}}$, on input tuples $(g_1,A_1,B_1,X_1),\ldots,(g_n,A_n,B_n,X_n)$, construct tuples (g_1,A_1,B_1,Y_1), $\ldots,(g_n,A_n,B_n,Y_n)$ by setting $Y_i=X_i/g_i$, for $i=1,\ldots,n$. Then $\mathcal{P}_k^{\mathsf{1ndh}}$ and $\mathcal{V}_k^{\mathsf{1ndh}}$ engage in Σ-protocol Π^{ddh} with the constructed tuples $(g_1,A_1,B_1,Y_1),\ldots,(g_n,A_n,B_n,Y_n)$ as input.

Theorem 4. *Π^{1ndh} is a perfect WI Σ-protocol for the polynomial-time relation* $\mathsf{1ndh}$.

Proof. The perfect WI property follows from the perfect WI of [CDS94]. 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 for it. On the other hand, if fewer than k of the input tuples are oneNDH then the constructed tuples contain fewer than k DH tuples. □

2.4 Instance-Dependent Trapdoor Commitment

We define the notion of a 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* $\mathtt{com}$.

A 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*

$$\mathrm{Prob}\left[\ (\mathtt{com}, \mathtt{dec}) \leftarrow \mathsf{Com}(x, m) : \mathsf{Dec}(x, \mathtt{com}, \mathtt{dec}, m) = 1\ \right] = 1.$$

- **Binding**: *if $x \notin L_{\mathcal{R}}$ then for every commitment* com *there exists at most one message m s.t.* $\mathsf{Dec}(x, \mathtt{com}, \mathtt{dec}, m) = 1$ *for any value* 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*

$$\mathrm{Prob}\left[\ b \leftarrow \{0,1\}; (\mathtt{com}, \mathtt{dec}) \leftarrow \mathsf{Com}(1^\lambda, x, m_b) : b = \mathcal{A}(\mathsf{aux}, x, \mathtt{com}, m_0, m_1)\ \right] \leq \frac{1}{2}.$$

- **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 [Dam10, HL10, DN02]. Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a Σ-protocol for the polynomial-time relation $\mathcal{R}$ with the associated NPlanguage $L_{\mathcal{R}}$ 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 x, and $m \in \{0,1\}^\lambda$, runs $(\mathtt{com}, \mathtt{dec}) \leftarrow S(x, m)$ and output $(\mathtt{com}, \mathtt{dec})$.
- Dec^Π takes as input $x, \mathtt{com}, \mathtt{dec}, m$ and gives its to $\mathcal{V}$. Outputs what $\mathcal{V}$ outputs.
- Fake_1^Π takes as input x, w, and samples a random string ρ and runs $\mathcal{P}$ on input $(1^\lambda, x, w; \rho)$ to get the 1st message $\mathtt{a}$ of Π. Fake_1^Π sets $\mathtt{rand} = \rho$, $\mathtt{com} = \mathtt{a}$ 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 $\mathtt{z}$ of Π. Fake_1^Π sets $\mathtt{dec} = \mathtt{z}$ and outputs dec.

Theorem 5. CS^Π *is an IDTC for the polynomial-time relation $\mathcal{R}$.*

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

For a polynomial-time relation $\mathcal{R}$, we define the *k-threshold* relation $\mathcal{R}_k$ as follows

$$\mathcal{R}_k = \Big\{ \Big((x_1, \ldots, x_n), ((w_1, d_1), \ldots, (w_k, d_k)) \Big) : 1 \leq d_1 < \cdots < d_k \leq n \\ \text{and } (x_{d_i}, w_i) \in \mathcal{R}, \text{ for } i = 1, \ldots, k \Big\}.$$

In this section we show that if $\mathcal{R}$ admits a delayed-input Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$, then, under the DDH assumption, $\mathcal{R}_k$ has a delayed-input WIPoK $\Pi_k = (\mathcal{P}_k, \mathcal{V}_k)$. Protocol Π_k uses Π and Π_k^{1ndh} as sub-protocols.

Let us give the intuition behind our construction. The prover starts by randomly selecting k oneNDH tuples and $n-k$ DH tuples. Each tuple is used to commit to a random and independently selected first message of protocol Π by using an IDTC $\mathsf{CS} = (\mathsf{Com}, \mathsf{Dec}, (\mathsf{Fake}_1, \mathsf{Fake}_2))$ (see. Sec. 2.4) for the polynomial time relation DH (defined in Sec. 2.3). More specifically the algorithm Com is used to computes the commitments w.r.t. the DH tuples, and the algorithm Fake_1 to computes the commitments w.r.t. the oneNDH tuples.

All commitments and tuples are sent to the verifier. Note that Π is a delayed-input Σ-protocol and thus the inputs are not needed to complete this step. Then the verifier's challenge c and n inputs along with the witnesses for k of them are made available to the prover. The prover associates each input for which a witness is available with one of the commitments sent to the verifier in the first round that was computed with respect to a oneNDH tuple. The commitment is opened by using Dec and the third round message is computed by the prover by using the witness. Instead, for an input for which no witness is available, the prover runs the simulator of Π and obtains an accepting transcript (a, c, z). The prover then associates a commitment computed with respect to a DH tuple and computes, by using the algorithm Fake_2, an opening of it as a. The the prover sends the opening and z to the verifier. In sums, for each input, the verifier receives an accepting transcript whose first message is shown to have been committed to in the first round. To ensure that the prover cannot cheat and choose n DH tuples, the prover and the verifier engage in parallel into an execution of Σ-protocol Π_k^{1ndh} described in Section 2.

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

1. Set $(\mathcal{G}, p, g) \leftarrow \mathsf{GG}(1^\lambda)$.
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 Π_k^{1ndh} on input $\mathcal{T} = (T_1, \ldots, T_n)$, witness $((\alpha_{b_1}, b_1), \ldots, (\alpha_{b_k}, b_k))$ and randomness ρ thus obtaining message a^{1ndh}. Send a^{1ndh} to $\mathcal{V}^k$.
5. For $i = 1, \ldots, n$:
   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 $(x_1, \ldots, x_n)$ and witnesses $(w_{d_1}, \ldots, w_{d_k})$ for inputs $x_{d_1}, \ldots, x_{d_k}$ (we denote by $\widetilde{d}_1, \ldots, \widetilde{d}_{n-k}$ the indices of the inputs for which no witness has been provided).
2. Compute the third round of Π^{1ndh} using c as challenge to get z^{1ndh} and send it to $\mathcal{V}^k$.

3. Pick a random permutation σ of $\{1,\ldots,k\}$ to associate each of the k oneNDH tuples $T_{b_1},\ldots,T_{b_k}$ with one of the k inputs $x_{d_1},\ldots,x_{d_k}$ for which a witness is available.
4. For $i=1,\ldots,k$:
   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 $\{1,\ldots,n-k\}$ to associate each of the $n-k$ DH tuples $T_{\widetilde{b}_1},\ldots,T_{\widetilde{b}_{n-k}}$ to one of the k inputs $x_{\widetilde{d}_1},\ldots,x_{\widetilde{d}_{n-k}}$ for which no witness is available.
6. For $i=1,\ldots,n-k$:
   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 $j=1,\ldots,n$: send M_j to $\mathcal{V}^k$.

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

1. $(a^{\mathsf{1ndh}},c,z^{\mathsf{1ndh}})$ is an accepting transcript for $\mathcal{V}^{\mathsf{1ndh}}$ with input T;
2. all t_j's are distinct;
3. for $j=1,\ldots,n$: $\mathtt{dec}_{t_j}$ is a valid decommitment of $\mathtt{com}_{t_j}$ with respect to T_{t_j};
4. for $j=1,\ldots,n$: (a_j,c,z_j) is an accepting transcript for $\mathcal{V}$ with input x_j;

3.1 (Adaptive-Input) Proof of Knowledge

In this section we start by proving that Π_k is a proof of knowledge and then we prove that if Π adaptive-input special sound then Π_k is an adaptive-input proof of knowledge.

Theorem 6. *Protocol Π_k is a proof of knowledge for $\mathcal{R}_k$.*

Proof. The completeness property is straightforward from the completeness of Π_k^{1ndh} and Π and from the correctness and trapdorness properties of the Instance-Dependent Trapdoor Commitment scheme used.

Now we proceed to prove that Π_k is N-special sound with $N=k(n-k+1)+1$. By the arguments of Section 2 about the proof of knowledge property of protocols that enjoy N-special soundness, we can conclude that Π_k is a proof of knowledge. Specifically, we show that there exists an efficient extractor that, for any sequence $(x_1,\ldots,x_n)$ of n inputs and for any set of N 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.

From any two (out of the N) accepting transcripts of Π^k, we can extract a collision for Π_k^{1ndh}. By the special soundness of Π_k^{1ndh}, the collision gives us the witnesses that k of the tuples used in the N transcripts are oneNDH. Without loss of generality, we assume that we obtain witnesses for $T_1,\ldots,T_k$ and we call them the *good* tuples. This implies that commitments $\mathtt{com}_1,\ldots,\mathtt{com}_k$ that appear in the shared first round of the N transcripts are opened to the same strings $a_1,\ldots,a_k$ in all the N transcripts. The k good tuples can be associated in each of the N transcripts to different

inputs; however, if the same good tuple is associated with the same input x_i in two different transcripts then we obtain a collision for Π with input x_i and thus, by the special soundness of Π, it is possible to extract a witness for x_i. We call such inputs *good* as well. Notice that we have N transcripts and k good tuples and thus we have $k \cdot N$ associations of good tuple to inputs.

We next prove that the N transcripts contain collisions of protocol Π with good tuples for at least k inputs and thus at least k witnesses can be extracted. For the sake of contradiction, suppose that there are only $m \leq k-1$ good inputs and let us count the good-tuple-to-input association (which we know to be $k \cdot N$) by counting the contributions of the good inputs and of the non-good inputs, separately. Clearly, each of the m good inputs is associated with at most one good tuple for each of the N transcripts thus contributing $m \cdot N$ associations. Each of the remaining $n-m$ inputs is associated with each of the k good tuples in at most *one* of the N transcripts thus contributing an extra $(n-m) \cdot k$ associations. We have thus counted a total of $m \cdot N + (n-m) \cdot k$ associations. Since the number of associations is an increasing function of m and since $m \leq k-1$, we have

$$\begin{aligned} m \cdot N + (n-m) \cdot k &= m \cdot (N-k) + n \cdot k \\ &\leq (k-1) \cdot (N-k) + n \cdot k \\ &\leq k \cdot N - N + k \cdot (n-k+1) \\ &= k \cdot N - 1 \end{aligned}$$

Contradiction.

□

Theorem 7. *If Π is adaptive-input special sound then Π^k is an adaptive-input proof of knowledge for $\mathcal{R}_k$.*

Proof. The delayed-input property follows from the completeness of protocols Π_k^{1ndh}, the correctness and trapdorness properties of the Instance-Dependent Trapdoor Commitment scheme used, and from the delayed-input completeness of Π.

To complete the proof we proceed similarly to the security proof of Theorem 6 by proving that the protocol Π^k is adaptive-input special sound. More specifically we show that there exists an efficient algorithm that on input 2 accepting transcripts (a, c_1, z_1) (a, c_2, z_2) for Π^k such that

- 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 $(x_1^2, \ldots, x_n^2)$,
- they share the same first round and
- they include different challenges,

outputs, k witnesses for each of the two sequences of theorems (for a total of at most $2 \cdot k$ different witnesses).

The extractor is based on the following observations.

First of all, observe that, by the special soundness of protocol Π_k^{1ndh}, 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. Given that Π^k is adaptive-input special sound, the property of adaptive-input PoK follows immediately from the arguments of Section 2.1. □

3.2 Adaptive-Input Witness Indistinguishability

Here we prove that Π_k is WI with respect to a PPT adversary $\mathcal{A}$ that is allowed to select the instance and the witnesses after receiving the first round. We have the following theorem.

Theorem 8. *Under the DDH assumption, if* Π *is SHVZK for* $\mathcal{R}$ *then* Π_k *is adaptive-input WI for relation* $\mathcal{R}_k$*.*

Proof. For sake of contradiction, let $\mathcal{A}$ be a PPT adversary and aux an auxiliary information for which $\mathsf{AdvAWI}_{\Pi_k,\mathcal{A}}(\lambda, \mathsf{aux})$ is a non-negligible function of λ. We let $X = (x_1, \ldots, x_n)$ denote the instance output by $\mathcal{A}$ at Step 2 of $\mathsf{ExpAWI}_{\Pi_k,\mathcal{A}}$ and we let $W^0 = ((w_1^0, d_1^0), \ldots (w_k^0, d_k^0))$ and $W^1 = ((w_1^1, d_1^1), \ldots (w_k^1, d_k^1))$ denote the witnesses output. We remark that $(x_{d_i^b}, w_i^b) \in \mathcal{R}$ for $i = 1, \ldots, k$ 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 call m the number of *unique* instances (these are instances for which a witness appears in exactly one of W^0 and W^1).

For integer $0 \leq m \leq k$ for which $\mathcal{A}$ has a positive probability of outputting X, W^0, W^1 with m unique instances, we define $\mathsf{AdvAWI}(\lambda, \mathsf{aux}|m)$ to be the advantage of $\mathcal{A}$ conditioned on the event that there are m unique instances. Notice that we dropped the indices $\mathcal{A}$ and Π_k not to overburden the notation. Since $\mathsf{AdvAWI}(\lambda, \mathsf{aux})$ is non-negligible in λ there must exist integer $0 \leq m \leq k$ such that $\mathsf{AdvAWI}(\lambda, \mathsf{aux}|m)$ is defined and non-negligible and the probability of having m unique instances is also non-negligible. Notice that the value of m depends solely on the adversary $\mathcal{A}$ and we consider it fixed throughout the proof. In designing the reductions to the problem of distinguishing DH tuples from oneNDH tuples (which, by Lemma 1, is hard under the DDH assumption), we shall assume that $\mathcal{A}$ would output X, W^0 and W^1 with m unique instances. When this is actually the case, which by our choice of m has a non-negligible probability of occurring, we will have a non-negligible advantage in solving the problem; when instead the number of unique instances is other than m, we will output a random guess. This is sufficient to prove that a non-negligible advantage in distinguishing DH tuples from oneNDH tuples is obtained.

We rename the instances of X output by $\mathcal{A}$, so that W^0 and W^1 can be written as

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

and

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

that is, we sort the instances of X so that the first m instances, $x_1, \ldots, x_m$, have a witness only in W^1; the next m instances, $x_{m+1}, \ldots, x_{2m}$, have a witness only in W^0; and the last $k-m$ instances,

$x_{2m+1}, \ldots, x_{m+k}$, have witnesses in the both. For any two such W^0 and W^1, we define the following intermediate sequences of witnesses $W_1, \ldots, W_k$:

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

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

   Note that W_i contains witnesses for $(x_1, \ldots, x_i, x_{m+1+i}, \ldots, x_{2m})$. 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.

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 start by proving indistinguishability of $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ for $i = 0, \ldots, m-1$. We will then argue about the indistinguishability of $\mathcal{H}_{m+i}$ and $\mathcal{H}_{m+i+1}$ for $i = 0, \ldots, k-m-1$. We assume inductively that in $\mathcal{H}_i$ the probability that there will be m unique instances is non-negligible and prove that this is still the case in $\mathcal{H}_{i+1}$. We remind the reader that, in $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$, the challenger $\mathcal{C}$ uses witnesses for the following k instances:

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

The differences between the two hybrid experiments are relative to instances x_{i+1} and x_{m+i+1} and to the witness used by the prover to complete the proof that at least k tuples are oneNDH. Specifically,

1. the transcript of Π for x_{i+1} is produced by the simulator in $\mathcal{H}_i$ and by the prover in $\mathcal{H}_{i+1}$;
2. the first messages of the transcript of Π for x_{i+1} is committed to with a DH tuple in $\mathcal{H}_i$ and with a oneNDH tuple in $\mathcal{H}_{i+1}$;
3. the transcript of Π for x_{m+i+1} is produced by the prover in $\mathcal{H}_i$ and by the simulator in $\mathcal{H}_{i+1}$;
4. the first messages of the transcript of Π for x_{m+i+1} is committed to with a oneNDH tuple in $\mathcal{H}_i$ and with a NDH tuple in $\mathcal{H}_{i+1}$;
5. protocol Π^{1ndh} is completed by using the witnesses for the oneNDH tuples associated with $x_1, \cdots, x_i, x_{m+i+2}, \cdots, x_{2m}, x_{2m+1}, \cdots, x_{m+k}$. In addition, the witness for the oneNDH tuple associated with x_{m+i+1} is used by $\mathcal{H}_i$ and the witness for the oneNDH tuple associated with x_{i+1} is used by $\mathcal{H}_{i+1}$.
6. in addition, in both hybrid experiments exactly k of the tuples are oneNDH and exactly $n-k$ are DH.

To prove indistinguishability of $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ we consider four intermediate hybrids: $\mathcal{H}_i^1, \ldots, \mathcal{H}_i^4$.

$\boldsymbol{\mathcal{H}_i^1}$ differs from $\mathcal{H}_i$ in the way the transcript of Π instance x_{i+1} is computed. Specifically, instead of using the SHVZK simulator of Π, $\mathcal{H}_i^1$ uses the algorithm of the prover of Π run with w_{i+1} as input. Indistinguishability of $\mathcal{H}_i^1$ and $\mathcal{H}_i$ follows directly from the following two observations. First, observe that x_{i+1} is associated in both $\mathcal{H}_i$ and $\mathcal{H}_i^1$ with a DH tuple and therefore the commitment is perfectly hiding. Moreover, Π's simulator is perfect.

$\boldsymbol{\mathcal{H}_i^2}$ differs from $\mathcal{H}_i^1$ in the way the tuples used to compute the commitments are chosen. Specifically, $\mathcal{H}_i^2$ chooses $k+1$ oneNDH tuples, $n-k-1$ DH tuples (as opposed to k oneNDH tuples and $n-k$ DH tuples). The extra oneNDH tuple is used by $\mathcal{H}_i^2$ to compute the commitment of the first message of Π that will be associated to x_{i+1} in the third round. As a sanity check, notice that in this case the commitment associated with x_{i+1} is binding but this is not a problem as an accepting transcript of Π for x_{i+1} can be computed by using witness w_{i+1}.

We observe that Lemma 1 implies that the probability that, in $\mathcal{H}_i^2$, $\mathcal{A}$ produces two sequences W^0 and W^1 of witnesses with m unique instances is still non-negligible[7].

Indistinguishability of $\mathcal{H}_i^1$ and $\mathcal{H}_i^2$ follows by Lemma 1. Let $T = (g, A, B, X)$ be either a random oneNDH tuple or a random DH tuple and consider the following simulator algorithm S that on input T produces the view of $\mathcal{A}$ in $\mathcal{H}_i^1$ or in $\mathcal{H}_i^2$ depending on whether T is DH or non-DH. The n tuples selected by S consist in k randomly selected oneNDH tuples, $n-k-1$ randomly selected DH tuples and T. Then, S executes the same steps as in $\mathcal{H}_i^1$ and will associate tuple T with x_{i+1}. Notice that S does not need a witness for T being DH to complete the execution as the accepting transcript of Π for x_{i+1} is computed using the witness w_{i+1} and the prover's algorithm and thus there is no need of a trapdoor to change the first message committed in the 1st round.

$\boldsymbol{\mathcal{H}_i^3}$ differs from $\mathcal{H}_i^2$ as protocol Π^{1ndh} is completed by using the witness of the tuple associated with x_{i+1} and not the tuple associated with x_{m+i+1} (along, obviously, the tuples associated with $x_1, \ldots, x_i$ and with $x_{m+i+2}, \ldots, x_{m+k}$). Indistinguishability of $\mathcal{H}_i^3$ and $\mathcal{H}_i^2$ follows from the perfect WI of Π^{1ndh}. Notice also that the only change intervenes after $\mathcal{A}$ has output the instances and the witnesses and therefore the probability of having m unique instances does not change.

$\boldsymbol{\mathcal{H}_i^4}$ differs from $\mathcal{H}_i^3$ in the way the tuples used to compute the commitments are chosen. Specifically, $\mathcal{H}_i^4$ chooses k oneNDH tuples, $n-k$ DH tuples (as opposed to $k+1$ oneNDH tuples and $n-k-1$ DH tuples). The extra DH tuple is used by $\mathcal{H}_i^4$ to compute the commitment of the first message of Π that will be associated to x_{m+i+1} in the third round. Indistinguishability of $\mathcal{H}_i^3$ and $\mathcal{H}_i^4$ follows by the same argument used for the indistinguishability of $\mathcal{H}_i^2$ and $\mathcal{H}_i^1$ and also by the same argument the probability of having m unique instances stays non-negligible.

Finally, we observe that $\mathcal{H}_i^4$ differs from $\mathcal{H}_{i+1}$ in the way the transcript of Π instance x_{m+i+1} is computed. Specifically, instead of using the prover of Π, $\mathcal{H}_i^4$ uses the simulator of Π. Indistinguishability of $\mathcal{H}_i^4$ and $\mathcal{H}_{i+1}$ follows by the same argument used for the indistinguishability of $\mathcal{H}_i$ and $\mathcal{H}_i^1$. Notice also that the only change intervenes after $\mathcal{A}$ has output the instances and the witnesses and therefore the probability of having m unique instances does not change.

We have thus proved that $\mathcal{H}_0$ is indistinguishable from $\mathcal{H}_m$. To complete the proof, we need to prove that $\mathcal{H}_{m+i}$ and $\mathcal{H}_{m+i+1}$ are indistinguishable for $i = 0, \ldots, k-m-1$. 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}: $\mathcal{H}_{m+i}$

[7] Actually, the probability is, up to a negligible additive factor, the same as in $\mathcal{H}_i^1$ which in turn is exactly the same as in $\mathcal{H}_i$

uses the witness from W^0 whereas $\mathcal{H}_{m+i+1}$ uses the witness from W^1. Indistinguishability then follows directly from the Perfect WI of Π.

□

4 Adaptive-Input Special-Soundness of Σ-Protocols

In this section, we first show Σ-protocols that are not secure when the adversarial prover can choose the statement adaptively after seeing the verifier's challenge. We then give an efficient compiler that, on input a Σ-protocol belonging to the general class considered in [Mau15, CD98], outputs a Σ-protocol that is adaptive-input sound.

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.

Issues with soundness. Let us consider the following well-known Σ-protocol Π_{DH} for relation DH (see Section 2.3). On common input $T = (g, A, B, X)$ and honest prover's private input α such that $A = g^\alpha$ and $X = B^\alpha$, the following steps are executed. We denote the size of the group $\mathcal{G}$ by q.

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 next show that the above Σ-protocol does not enjoy special soundness if an adversarial prover is allowed to select X adaptively after seeing the challenge c sent by the honest verifier. Indeed, consider the following two conversations $((a = g^r, x = B^s), c_1, z_1 = r + \alpha \cdot c_1)$ and $((a = g^r, x = B^s), c_2, z_2 = r + \alpha \cdot c_2)$ for tuples (g, A, B, X_1) and (g, A, B, X_2) where $A = g^\alpha$, $X_1 = g^{\gamma_1}$ and $X_2 = g^{\gamma_2}$ for $\gamma_i = \frac{z_i - s}{c_i} = \alpha + \frac{r-s}{c_i}$, 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 succeeds in convincing the verifier even though the statement is false.

A similar argument can be used to prove that the Σ-protocol of [MP03] 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 Schnorr's Σ-protocol [Sch89] 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, and thus and adversarial prover can only violate special soundness; that is, it is not possible to extract a witness from two accepting transcripts with the same first message.

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.

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

Since $c_1 \neq c_2$, the linear system above has q^2 solutions and thus the two transcripts give no information on either of the two witnesses.

4.2 A Compiler for Adaptive-Input Special Soundness

In this section, we construct an adaptive-input special soundness Σ-protocol Π_f^{a} for proving knowledge of the pre-image of a homomorphic function. Our construction is based on the Σ-protocol $\Pi_f = (\mathcal{P}_f, \mathcal{V}_f)$ given in [CD98, Mau15] that generalizes and subsumes several Σ-protocols, including the one by Schnorr [Sch89], Guillou-Quisquater [GQ88] and the Σ-protocol for DH tuples [DH76]. Thus our construction can be seen as a compiler that adds adaptive-input special soundness to the Σ-protocols generalized by the protocol of [CD98, Mau15].

Let us proceed more formally. Let $(\mathcal{G}, \star)$ and $(\mathcal{H}, \otimes)$ be two groups with efficient operations and let $f : \mathcal{G} \to \mathcal{H}$ 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. We next describe Σ-protocol Π_f for relation $\mathcal{R}_f = \{(x, w) : x = f(w)\}$. Here prover $\mathcal{P}_f$ and verifier $\mathcal{V}_f$ receive as input the description of groups $\mathcal{G}$ and $\mathcal{H}$ along with an element $x \in \mathcal{H}$. In addition, $\mathcal{P}_f$ receives w such that $x = f(w)$ as a private input. Let $\mathcal{C}$ be an arbitrary subset of $\mathbb{N}$.

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

Theorem 3 of [Mau15] 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 $u \in \mathcal{G}$ such that the following two conditions are satisfied are known:

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 to give Schnorr's protocol, the Guillou-Quisquater protocol or the protocol for DH tuples. 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 $\Pi_f^{\mathbf{a}}$. We have seen that in the case of the Σ-protocol for DH, if $\mathcal{P}^*$ sends a non-DH tuple in the first round, instead of a DH tuple as prescribed by the protocol, then he succeeds in violating special soundness. We next show that if we force the prover to correctly compute the first message then the protocol is adaptive-input special soundness.

Roughly speaking, we augment Π_f by requiring the prover to also give a proof of knowledge of the randomness used to compute the first round a of Π_f which for Π_f consists in executing a second instance of the same protocol on input the first message a. We observe that the second instance of Π_f is not subject to an adaptive-input attack since the adversarial prover is committed to using the first message a of the first instance of Π_f as input.

More precisely, we consider the 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, in both instances, the verifier of Π_f accepts. For our propose we need to set the challenge space of Π_f^{a} as $\mathcal{C}_f^{\mathrm{a}} = \mathcal{C}_f - \{0\}$, where $\mathcal{C}_f$ is the challenge space of Π_f. The reason why we cannot have $\mathcal{C}_f^{\mathrm{a}} = \mathcal{C}_f$ will be clear in the proof of adaptive-input special soundness. For now we just make the following considerations.

1. If Π_f has challenge space $\mathcal{C}_f$ of size one, then Π_f is already adaptive-input special sound, therefore our compiler just sets $\Pi_f^{\mathrm{a}} = \Pi_f$.

2. If Π_f has challenge space $\mathcal{C}_f$ of size two, and $|\mathcal{C}_f - \{0\}| = 1$, then our compiler outputs $\Pi_f^{\mathrm{a}} = \Pi_f$ setting $\mathcal{C}_f^{\mathrm{a}} = \mathcal{C}_f - \{0\}$. We observe that, by definition, Π_f^{a} enjoys the property of adaptive-input special soundness.

3. In general, if $\mathcal{C}_f$ contains 0, then the output protocol of the compiler has challenge smaller than $\mathcal{C}_f$, and therefore has also worst knowledge error than the starting protocol. This can be easily avoided by using Theorem 2 in order to amplify the challenge space of Π_f before using our compiler. We recall that preserve the knowledge error is important in order to preserve the PoK property of the input protocol[8].

Let us now prove that Π_f^{a} enjoys adaptive-input special soundness (when we are not in the cases 1 and 2 described before). Suppose we have two accepting transcripts of Π_f^{a} that share the first message but have different challenge messages. By special soundness of the second instance of Π_f, we can extract the randomness r used to compute the first message a of the first instance of Π_f from the two sub-transcripts of the second instance of Π_f. We conclude the proof by showing that it is possible to compute a witness for x from one accepting transcript (a, c, z) of Π_f for x if r such that $f(r) = a$ is available. Let y and u be such that $f(u) = x^y$ and $\gcd(y, c) = 1$. These values can be computed since f satisfies the conditions set by Theorem 3 of [Mau15]. More precisely, we observe that the Theorem 3 of [Mau15] works only if $\gcd(y, c) = 1$ and $c \neq 0$, and this is the reason why we require that the challenge space of Π_f^{a} does not contain the challenge 0. Then, by using the extended GCD algorithm, we can compute α and β such that $y \cdot \alpha + c \cdot \beta = 1$. Finally, we set $w = u^{\alpha} \star (r^{-1} \star z)^{\beta}$. Now observe that $f(z) = a \otimes x^c$ and this implies that $f(r^{-1} \star z) = x^c$. Therefore, $f(w) = f(u^{\alpha} \star (r^{-1} \star z)^{\beta}) = f(u)^{\alpha} \otimes f(z \star r^{-1})^{\beta} = x^{y\alpha} \otimes x^{\beta c} = x$ that proves that $f(w) = x$.

We have thus proved the following theorem.

[8] We observe that the Theorem 3 cannot be directly applied to the Σ-protocol described in this section because of the different notation used to describe the challenge space of a Σ-protocol.

Theorem 9. *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 [CPS+16a]'s Transform

Ciampi et al. in [CPS+16a] give a compiler that takes as input two Σ-protocols, Π_0 and Π_1 for languages L_0 and L_1, and constructs a new Σ-protocol Π^{OR} for $L_0 \vee L_1$ in which the instance for language L_1 is needed by the prover only in the 3rd round. The compiler requires Π_1 to be delayed input. In this section we show that if Π_1 enjoys adaptive-input special soundness, then so does Π^{OR}.

5.1 Overview of the Construction of [CPS+16a]

We start by giving a succinct description of the main building block used by [CPS+16a].

t-Instance-Dependent Trapdoor Commitment. In a t-Instance-Dependent Trapdoor Commitment (a t-IDTC) scheme for polynomial-time relation $\mathcal{R}$, one party can commit a message m from a predefined message space M with respect to an instance x. The commitment produced is hiding and binding unless a witness w such that $(x, w) \in \mathcal{R}$; in this case, the commitment can be equivocated.

More precisely, a t-IDTC scheme consists of a triple of PPT algorithms $(\mathsf{TCom}, \mathsf{TDec}, \mathsf{TFake})$ with the following syntax. The *commitment* algorithm TCom takes as input an instance x and a message m and returns a commitment $\mathtt{com}$ of m with respect to x along with a decommitment $\mathtt{dec}$. The decommitment algorithm TDec takes as input $\mathtt{com}$, $\mathtt{dec}$ and m and x and verifies whether $\mathtt{dec}$ is an opening of $\mathtt{com}$ as m with respect to x. TFake is the *equivocation* procedure that, given a witness for an instance x, a commitment $\mathtt{com}$ of message m with respect to x along with the random coin tosses used to produce it and message m' outputs a decommitment $\mathtt{dec}'$ of $\mathtt{com}$ as m'. It is required that a t-IDTC is *correct* (honestly computed commitments are correctly decommitted), *hiding* (honestly computed commitments hide the message), *trapdoor* (a fake commitment is indistinguishable from an honestly computed commitment) and *t-Special Extractable*. 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. A construction of a 2-IDTC scheme that is perfect hiding, perfect trapdoor and 2-Special Extractable from a special type of Σ-protocols is given in [CPS+16a].

5.1.1 The Construction of [CPS+16a]

Let $\mathcal{R}_0$ be a relation admitting a t-IDTC scheme with $t = 2$ and let $\mathcal{R}_1$ be a relation admitting an delayed-input Σ-protocol Π_1; we denote by S_1 the associated simulator for the honest verifier zero knowledge. We next describe the Σ-protocol $\Pi^{\mathsf{OR}} = (\mathcal{P}^{\mathsf{OR}}, \mathcal{V}^{\mathsf{OR}})$ of [CPS+16a] for the OR relation:

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

We assume that x_0 and the length of x_1 in unary are available from the onset of the protocol whereas x_1 and the witness w such that $((x_0, x_1), w) \in \mathcal{R}^{\mathsf{OR}}$ are given before the 3rd round. Obviously, the verifier does not get to learn w.

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 Σ-protocol Π_1 using r_1 as random coin tosses;
   1.2. compute a pair $(\mathtt{com}, \mathtt{dec}_1)$ of commitment and decommitment of a_1 with respect to x_0.
   1.3. send $\mathtt{com}$ 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 Π_1, z_1, using as input (x_1, w, c);
   3.2. send $(\mathtt{dec}_1, a_1, z_1)$ to $\mathcal{V}^{\mathsf{OR}}$;
   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 $\mathtt{com}$ as a_2;
   3.4. send $(\mathtt{dec}_2, a_2, z_2)$ to $\mathcal{V}^{\mathsf{OR}}$.
4. $\mathcal{V}^{\mathsf{OR}}$ receives (a, z) and $\mathtt{dec}$ from $\mathcal{P}^{\mathsf{OR}}$ and accepts if and only if the following conditions are satisfied:
   4.1. (a, c, z) is an accepting conversation for x_1;
   4.2. $\mathtt{dec}$ is a valid decommitment of $\mathtt{com}$ for a message a.

5.2 Adaptive-Input Soundness of Π^{OR}

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

Theorem 10. *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 $(\mathtt{com}, c, (a, z, \mathtt{dec}))$ for input (x_0, x_1) and an accepting transcript $(\mathtt{com}, c', (a', z', \mathtt{dec}'))$ 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 adaptive-input special soundness of Π_1, there exists an efficient extractor 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 for x_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 $\Sigma_1, \ldots, \Sigma_n$ for polynomial-time relations $\mathcal{R}_1, \ldots, \mathcal{R}_n$, we construct an Adaptive-Input Proof of Partial Knowledge $\Gamma_k^{\mathcal{R}_1,\ldots,\mathcal{R}_n} = (\mathcal{P}_{\Gamma_k}^{\mathcal{R}_1,\ldots,\mathcal{R}_n}, \mathcal{V}_{\Gamma_k}^{\mathcal{R}_1,\ldots,\mathcal{R}_n})$ for the k-threshold polynomial-time relation

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

Not to overburden our notation, we will just write $\mathcal{R}_k, \Gamma_k, \mathcal{P}_{\Gamma_k}$, and $\mathcal{V}_{\Gamma_k}$, whenever $\mathcal{R}_1, \ldots, \mathcal{R}_n$ are clear from the context.

Let us start by providing some intuition on Γ_k. For $j = 1, \ldots, n$, the prover of Γ_k computes two first-round messages for Σ_j and commits to each one using a different tuple. Note that Σ_j is delayed-input so this step can be performed without knowing the instances (nor the witnesses). The two tuples, T_j^0 and T_j^1, used to compute the commitments for j are chosen so that they share the first three components and T_j^1 is DH and, consequently, T_j^0 is non-DH. Therefore, the verifier is guaranteed that, for each j, at most one of the commitments of the tuple is DH (and thus at most one commitment can be equivocated). For each j, the tuples T_j^0 and T_j^1 and the relative commitments $\mathtt{com}_j^0$ and $\mathtt{com}_j^1$ are sent in random order to the verifier. The prover then receives the n instances, witnesses for k of them and the verifier's challenge. Then, for each instance x_j for which a witness is available, the prover decommits the commitment computed using T_j^0 (the non-DH tuple) and computes the third round message by running the prover of Σ_j. On the other hand, for each instance x_j for which no witness is provided, the prover runs the simulator of Σ_j and produces an accepting transcript; then the commitment computed using T_j^1 (the DH tuple) is opened as the first message of the simulated transcript, by using equivocation. In both cases, the decommitment and the accepting transcript are sent to the verifier.

The verifier expects to receive an accepting transcript for each instance with the first message coming (through equivocation for $n-k$ of them) from a commitment sent as part of the first round. Moreover, to ensure that the prover has not cheated by equivocating too many commitments, the prover and the verifier engage in parallel in a protocol by which the prover proves that from among the n unopened commitments at least k are associated with DH tuples; this implies, by the way the pairs of tuples have been constructed, that at least k of the opened commitments are associated with non-DH tuples. We denote the resulting protocol by Π_k. Notice that when Π_k starts, the prover does not know which commitment will be opened (since this depends on the witnesses received after the challenge); this can be handled by our protocol that it only needs instances and witnesses before the third round starts.

1st round. $\mathcal{P}_{\Gamma_k} \Rightarrow \mathcal{V}_{\Gamma_k}$:

$\mathcal{P}_{\Gamma_k}$ 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 $(\mathcal{G}, p, g) \leftarrow \mathsf{GG}(1^\lambda)$.

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, by construction, T_j^1 is a DH tuple).

3. Select random string R and use it to compute the first round message a^{Π_k} of Π_k by running prover $\mathcal{P}_k$.
   Send a^{Π_k} to $\mathcal{V}_{\Gamma_k}$.

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-round messages a_j^0 and a_j^1 of Σ_j by running the prover of Σ_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 $\mathcal{V}_{\Gamma_k}$.

2nd round. $\mathcal{V}_{\Gamma_k} \Rightarrow \mathcal{P}_{\Gamma_k}$: $\mathcal{V}_{\Gamma_k}$ randomly selects a challenge c and sends it to $\mathcal{P}_{\Gamma_k}$.

3rd round. $\mathcal{P}_{\Gamma_k} \Rightarrow \mathcal{V}_{\Gamma_k}$:

$\mathcal{P}_{\Gamma_k}$ receives theorems $x_1, \ldots, x_n$ and, for $d_1 < \ldots < d_k$, 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 z_j by running prover of Σ_j on input (x_j, w_l), randomness R_j^0 used to compute the first round a_j^0, and 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. Use j as a shorthand for $\widetilde{d}_l$.

   2.2. Set $U_j = T_j^0$ and $\hat{U}_j = T_j^1$.

   2.3. Run the simulator of Σ_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 $\mathcal{V}_{\Gamma_k}$.

4. Compute the third round z^{Π_k} of Π_k by running prover of Π_k on input tuples $(U_1, \ldots, U_n)$, witnesses $\alpha_{d_1}, \ldots, \alpha_{d_k}$ and randomness R used to compute the first round a^{Π_k}.

$\mathcal{V}_{\Gamma_k}$ accepts if and only if the following conditions are satisfied.

1. For $j = 1, \ldots, n$
   Check that the two tuples sent for j in the first round share the first three components.
   Write M_j as $M_j = (a_j, z_j, \mathtt{dec}_j, \hat{U}_j)$.
   Check that (a_j, c, z_j) is an accepting conversation of Σ_j for instance x_j.
   Check that $\mathtt{dec}_j$ is a decommitment as a_j with respect to tuple $\hat{U}_j$ of one of $\mathtt{com}_j^0$ and $\mathtt{com}_j^1$ and that tuple $\hat{U}_j$ was associated with j in the first round. Denote by U_j the other tuple associated with j.
2. Check that $(a^{\Pi_k}, c, z^{\Pi_k})$ is an accepting conversation of $\mathcal{V}_{\Gamma_k}$ for instances $U_1, \ldots, U_n$.

6.1 (Adaptive-Input) Proof of Knowledge

In this section we prove that Γ_k enjoys the property of PoK. Also, assuming that $\Pi_1, \ldots, \Pi_n$ are adaptive-input special sound, we prove that Γ_k enjoys the property of adaptive-input PoK.

Theorem 11. *Γ_k is a proof of knowledge for $\mathcal{R}_k^{\mathcal{R}_1,\ldots,\mathcal{R}_n}$.*

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.

As for the previous security proofs, we proceed by proving that our protocol is N-special sound. More precisely we prove that Γ_k is $(2n+k)$-special sound assuming that Π_k is adaptive-input special sound.

First of all we observe that combining the arguments given in the security proof of Theorem 7, and the fact that the well known Σ-protocol for DH tuple [DH76] can be converted into one that enjoys adaptive-input special soundness (see. Sec. 4.2), we can easily claim that Π_k is adaptive-input special sound.

Now we are ready to prove that there exists an efficient extractor which, for any sequence $(x_1, \ldots, x_n)$ of n fixed inputs and for any set of $2n + k$ accepting conversations of Γ_k that share the same first message and have different challenges, outputs the witness of w_i s.t. $(x_{d_i}, w_i) \in \mathcal{R}_{d_i}$ for $i = 1, \ldots, k$, with $1 \leq d_1 < \cdots < d_k \leq n$.

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 12. *If Π^i, for $i = 1, \ldots, n$, are adaptive-input special sound then Γ_k is an adaptive-input proof of knowledge for $\mathcal{R}_k^{\mathcal{R}_1,\ldots,\mathcal{R}_n}$.*

Proof. As has been done in the previous security proof, we assume without loss of generality that Π_k is adaptive-input special sound.

The delayed-input completeness of Γ_k follows from the delayed-input completeness of protocols Π_k, the correctness and trapdorness properties of the Instance-Dependent Trapdoor Commitment scheme used, and from the delayed-input completeness of Π^i for $i = 1, \ldots, n$.

In order to prove that Γ_k enjoys the property of adaptive-input PoK we need to show an extractor $\mathsf{AExtract}$ that, given oracle access to $\mathcal{P}^{\star}_{\Gamma_k}$, outputs an instance x with the respective witness w (see. Sec 2.1). $\mathsf{AExtract}$ complete an execution of the protocol Γ_k acting as the verifier $\mathcal{V}_{\Gamma_k}$ does, in order to obtain a theorem x^1 and a transcript (a, c^1, z^1) (that is accepting with non-negligible probability). At this point $\mathsf{AExtract}$, for $j = 2, \ldots, N$, rewinds $\mathcal{P}^{\star}_{\Gamma_k}$ before that the challenge has been sent, picks a fresh challenge c^j (never used before), and send it to Γ_k in order to obtain, with non-negligible probability, another accepting transcript a, c^j, z^j with respect to a theorem $x^j = (x^j_1, \ldots, x^j_n)$.

First of all we observe that from any two (out of the N) accepting transcripts of Π_k, we can extract a collision for Π_k. By the adaptive-input special soundness of Π_k, the collision gives us the witnesses that k of the tuples associated to the commitments opened in the last round of each accepting transcript are non-DH. Without loss of generality, we refer to that tuples as $T^j_1, \ldots, T^j_k$ for the j-th transcript, with $j = 1, \ldots, N$.

In order to extract a witness for $x^1 = (x^1_1, \ldots, x^1_n)$ it is necessary to obtain, for every $t = 1, \ldots k$, at least one transcript (a, c^y, z^y) that has $T^y_t = T^1_t$, with $y \in \{2, \ldots, N\}$. With sufficiently large $N = \mathtt{poly}(\lambda)$, the probability that this condition is satisfied is non-negligible. The proof ends with the observation that, because of the adaptive-input special soundness property of $\Pi^1, \ldots, \Pi^n$, when a (binding) commitment $\mathtt{com}$ computed with respect to the tuple T^1_t (for $t = 1, \ldots, k$) is opened two times, then is it possible to extract the witness for the theorems proved using as a first round the message committed in $\mathtt{com}$.

□

6.2 Adaptive-Input Witness Indistinguishability

Theorem 13. *If relations $\mathcal{R}_1, \ldots, \mathcal{R}_n$ admit a SHVZK Σ-protocol then, under the DDH assumption, Γ_k is adaptive-input WI for $\mathcal{R}^{\mathcal{R}_1, \ldots, \mathcal{R}_n}$.*

Proof. We adopt the same framework of the proof for the case of one relation. Specifically, we have the same definition of hybrid witness sequence W_i and of hybrid experiment $\mathcal{H}_i$, for $i = 0, \ldots, k$. We start by proving indistinguishability of $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ for $i = 0, \ldots, m-1$. The differences between the tho hybrids are relative to inputs x_{i+1} and x_{m+i+1} and the witness used to complete protocol Π_k. We consider the following intermediate hybrids.

$\boldsymbol{\mathcal{H}^1_i}$ differs from $\mathcal{H}_i$ in the way the transcript of Σ_{i+1} for instance x_{i+1} is computed. Specifically, instead of using the SHVZK simulator of Σ_{i+1}, $\mathcal{H}^1_i$ uses the algorithm of the prover of Σ_{i+1} run with w_{i+1} as input. Notice that in both hybrids the tuple $\widehat{U}_{i+1}$ used to commit the first-round message of Σ_{i+1} is a DH tuple whereas U_{i+1} is a non-DH tuple. In $\mathcal{H}^1_i$ however the equivocability of the commitment is not used and the transcript is completed using the witness and the prover's algorithm for Σ_{i+1}. Indistinguishability of $\mathcal{H}^1_i$ and $\mathcal{H}_i$ follows directly from the perfect SHVZK.

$\boldsymbol{\mathcal{H}^2_i}$ differs from $\mathcal{H}^1_i$ in the fact that tuple $\widehat{U}_{i+1}$ is a non-DH tuple and U_{i+1} is a DH tuple. In other words, in $\mathcal{H}^2_i$ the first-round message of the transcript relative to x_{i+1} shown at the third round has been committed to by a non-DH tuple.

Suppose now, for sake of contradiction, that there exists a PPT distinguisher D and a polynomial $\mathtt{poly}(\cdot)$ such that, for sufficiently large security parameters λ,

$$p_1(\lambda) \geq p_2(\lambda) + 1/\mathtt{poly}(\lambda),$$

where $p_1(\lambda)$ and $p_2(\lambda)$ are, respectively, the probabilities that D outputs 1 on input the view of $\mathcal{A}$ in $\mathcal{H}_i^1$ and in $\mathcal{H}_i^2$. We use D and $\mathcal{A}$ to design an algorithm $\mathcal{B}$ that has a non-negligible advantage in the DDH game. Algorithm $\mathcal{B}$ receives as input a challenge tuple $T = (g, A, B, X)$, randomly selects Y and constructs the tuple $\widehat{T} = (g, A, B, Y)$ that shares the first three components with T. Then $\mathcal{B}$ simulates the view of $\mathcal{A}$ in $\mathcal{H}_1^i$ with the following two modifications: T and $\widehat{T}$ are the tuples associated with $i+1$; in the third round, $\mathcal{B}$ selects one of the two at random and the commitment computed at the first round with respect to the selected tuple is opened. $\mathcal{B}$ feeds D with the view generated by interacting with $\mathcal{A}$ and records D's output. Finally, $\mathcal{B}$ outputs 1 if and only if it had selected T and D has output 1 or it had selected $\widehat{T}$ and D has output 0.

Now, let us compute the probability that $\mathcal{B}$ outputs 1 when T is DH tuple. Notice that if T is selected (and this happens with probability 1/2) then $\mathcal{B}$ has produced the view of $\mathcal{H}_i^1$ and thus the probability that D outputs 1 is p_1. On the other hand, if $\widehat{T}$ is selected then $\mathcal{B}$ has produce the view of $\mathcal{H}_i^2$ and thus the probability that D outputs 0 is $1 - p_2$. Therefore, it T is DDH, $\mathcal{B}$ outputs 1 with probability $1/2 \cdot (1 + p_1 - p_2)$.

Consider now the case in which T is non-DH. In this case, both T and $\widehat{T}$ are non-DH and thus the view received by D is independent from which one of the two is selected by $\mathcal{B}$. We thus denote by p the probability that D outputs 1 when the view contains two non-DH tuples. As in the previous case, $\mathcal{B}$ outputs 1 if T is selected and D outputs 1 (this event has probability $1/2p$) and $\widehat{T}$ is selected and D outputs 0 (this event has probability $1/2 \cdot (1-p)$). Therefore $\mathcal{B}$ has probability 1/2 of outputting 1 when it receives a non-DH tuple in output.

We thus conclude that $\mathcal{B}$ breaks the DDH assumption. Contradiction.

$\boldsymbol{\mathcal{H}_i^3}$ differs from $\mathcal{H}_i^2$ in the witness used to compute an accepting transcript for Π_k. More specifically α_{i+1} is used instead of α_{m+i+1} in the tuple that define the witness for Π_k. Observe that this is possible because U_{i+1} is a DH tuple. Suppose now, for sake of contradiction, that there exists a PPT distinguisher D and a polynomial $\mathtt{poly}(\cdot)$ such that, for sufficiently large security parameters λ,

$$p_1(\lambda) \geq p_2(\lambda) + 1/\mathtt{poly}(\lambda),$$

where $p_1(\lambda)$ and $p_2(\lambda)$ are, respectively, the probabilities that D outputs 1 on input the view of $\mathcal{A}$ in $\mathcal{H}_i^2$ and in $\mathcal{H}_i^3$. We use D and $\mathcal{A}$ to design an algorithm $\mathcal{B}$ that has a non-negligible advantage in break the adaptive WI property of Π_k. Algorithm $\mathcal{B}$ receives as input the first round challenge a^{Π_k} and computes all the other informations needed to compute the first round of the protocol of Γ_k. $\mathcal{B}$, upon receiving the challenge c computes the challenge theorem and witnesses as following:

$$X_{\Pi_k} = (U_1, \ldots, U_n);$$

$$W^0_{\Pi_k} = (\alpha_1, \ldots, \alpha_i, \alpha_{m+i+1}, \alpha_{m+i+2}, \ldots, \alpha_{2m}, \alpha_{2m+1}, \ldots \alpha_{m+k});$$

$$W^1_{\Pi_k} = (\alpha_1, \ldots, \alpha_{i+1}, \alpha_{m+i+2}, \ldots, \alpha_{2m}, \alpha_{2m+1}, \ldots \alpha_{m+k}).$$

Then sends them, with the challenge c, to the challenger. $\mathcal{B}$, upon receiving z^{Π_k}, completes the third round protocol using z^{Π_k}, and sends it to $\mathcal{A}$. $\mathcal{B}$ feeds D with the view generated by interacting

with $\mathcal{A}$ and records D's output. Finally, $\mathcal{B}$ outputs 1 if and only if the witness $W^0_{\Pi_k}$ has been used by the challenger and D has output 1 or has been used $W^1_{\Pi_k}$ and D has output 0.

$\boldsymbol{\mathcal{H}^4_i}$ differs from $\mathcal{H}^3_i$ in the fact that tuple $\widehat{U}_{m+i+1}$ is a DH tuple and U_{m+i+1} is a non-DH tuple. In other words, in $\mathcal{H}^4_i$ the first-round message of the transcript relative to x_{m+i+1} shown at the third round has been committed to by a DH tuple. The indistinguishability between $\mathcal{H}^3_i$ and $\mathcal{H}^4_i$ follows the same arguments of the indistinguishability between $\mathcal{H}^2_i$ and $\mathcal{H}^1_i$

Finally, we observe that $\mathcal{H}^4_i$ differs from $\mathcal{H}_{i+1}$ in the way the transcript of Π_{m+i+1} for the instance x_{m+i+1} is computed. Specifically, instead of using the prover of Π, $\mathcal{H}^4_i$ uses the simulator of Π. Indistinguishability of $\mathcal{H}^4_i$ and $\mathcal{H}_{i+1}$ follows by the same argument used for the indistinguishability of $\mathcal{H}_i$ and $\mathcal{H}^1_i$.

We have thus proved that $\mathcal{H}_0$ is indistinguishable from $\mathcal{H}_m$. To complete the proof, we need to prove that $\mathcal{H}_{m+i}$ and $\mathcal{H}_{m+i+1}$ are indistinguishable for $i = 0, \ldots, k-m-1$. 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}: $\mathcal{H}_{m+i}$ uses the witness from W^0 whereas $\mathcal{H}_{m+i+1}$ uses the witness from W^1. Indistinguishability then follows directly from the Perfect WI of Π.

□

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.

A preliminary version of this work will appear in the proceedings of the Eurocrypt 2016 [CPS+16b].

References

[BPSV08] Carlo Blundo, Giuseppe Persiano, Ahmad-Reza Sadeghi, and Ivan Visconti. 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*, pages 364–378, 2008. (Cited on page 11.)

[BPW12] David Bernhard, Olivier Pereira, and Bogdan Warinschi. 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*, pages 626–643, 2012. (Cited on page 4.)

[CD98] Ronald Cramer and Ivan Damgård. Zero-knowledge proofs for finite field arithmetic; or: Can zero-knowledge be for free? In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 424–441. Springer, 1998. (Cited on pages 1, 4, 5, 22, and 23.)

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *Advances in Cryptology — CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer Berlin Heidelberg, 1994. (Cited on pages 1, 3, 4, 5, 6, 7, 8, 9, 11, and 14.)

[CDV06] Dario Catalano, Yevgeniy Dodis, and Ivan Visconti. 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*, pages 120–144, 2006. (Cited on page 11.)

[CPS+16a] Michele Ciampi, Giuseppe Persiano, Alessandra Scafuro, Luisa Siniscalchi, and Ivan Visconti. 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*, pages 112–141, 2016. (Cited on pages 1, 2, 3, 4, 5, 6, 8, 9, 12, and 25.)

[CPS+16b] Michele Ciampi, Giuseppe Persiano, Alessandra Scafuro, Luisa Siniscalchi, and Ivan Visconti. Online/offline or composition of sigma protocols. In *Advances in Cryptology - EUROCRYPT 2016, 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016*, 2016. (Cited on page 32.)

[CPSV16] Michele Ciampi, Giuseppe Persiano, Luisa Siniscalchi, and Ivan Visconti. 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*, pages 83–111, 2016. (Cited on page 11.)

[CV05] Dario Catalano and Ivan Visconti. Hybrid trapdoor commitments and their applications. In *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings*, pages 298–310, 2005. (Cited on page 7.)

[CV07] Dario Catalano and Ivan Visconti. Hybrid commitments and their applications to zero-knowledge proof systems. *Theor. Comput. Sci.*, 374(1-3):229–260, 2007. (Cited on page 7.)

[Dam10] Ivan Damgård. On Σ-protocol. http://www.cs.au.dk/~ivan/Sigma.pdf, 2010. (Cited on pages 10, 11, and 15.)

[DG03] Ivan Damgård and Jens Groth. 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*, pages 426–437, 2003. (Cited on pages 7 and 11.)

[DH76] W Diffie and ME Hellman. New directions in cryptography, ieee transctions in information theory. *Volume IT-22, November*, 1976. (Cited on pages 23 and 29.)

[DN02] Ivan Damgård and Jesper Buus Nielsen. 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*, pages 581–596, 2002. (Cited on page 15.)

[DSDCPY94] Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, and Moti Yung. On monotone formula closure of SZK. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 454–465, 1994. (Cited on page 3.)

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 186–194, 1986. (Cited on page 4.)

[GMY06] Juan A. Garay, Philip MacKenzie, and Ke Yang. Strengthening zero-knowledge protocols using signatures. *Journal of Cryptology*, 19(2):169–209, 2006. (Cited on page 10.)

[GQ88] Louis C. Guillou and Jean-Jacques Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In Christoph G. Günther, editor, *Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*, volume 330 of *Lecture Notes in Computer Science*, pages 123–128. Springer, 1988. (Cited on page 23.)

[HL10] Carmit Hazay and Yehuda Lindell. *Efficient Secure Two-Party Protocols - Techniques and Constructions*. Information Security and Cryptography. Springer, 2010. (Cited on pages 3, 7, and 15.)

[KO04] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, pages 335–354, 2004. (Cited on page 3.)

[Lin15] Yehuda Lindell. 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*, pages 93–109, 2015. (Cited on page 11.)

[LS90] Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. In *Advances in Cryptology - CRYPTO*, 1990. (Cited on pages 1, 3, 8, and 9.)

[Mau15] Ueli Maurer. Zero-knowledge proofs of knowledge for group homomorphisms. *Designs, Codes and Cryptography*, pages 1–14, 2015. (Cited on pages 1, 4, 5, 22, 23, and 24.)

[MP03] Daniele Micciancio and Erez Petrank. 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*, pages 140–159, 2003. (Cited on pages 4, 11, and 22.)

[OPV10] Rafail Ostrovsky, Omkant Pandey, and Ivan Visconti. 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*, pages 535–552, 2010. (Cited on page 11.)

[ORS15] Rafail Ostrovsky, Silas Richelson, and Alessandra Scafuro. 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*, pages 339–358, 2015. (Cited on page 6.)

[OV12] Rafail Ostrovsky and Ivan Visconti. Simultaneous resettability from collision resistance. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:164, 2012. (Cited on page 3.)

[Pas03] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 160–176. Springer, 2003. (Cited on page 3.)

[Sch89] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer New York, 1989. (Cited on pages 4, 22, and 23.)

[Vis06] Ivan Visconti. 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*, pages 22–33, 2006. (Cited on page 11.)

[YZ07] Moti Yung and Yunlei Zhao. 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*, pages 129–147, 2007. (Cited on page 11.)