On Round-Optimal Zero Knowledge in the Bare Public-Key Model

Alessandra Scafuro and Ivan Visconti

Dipartimento di Informatica, University of Salerno, ITALY
{scafuro,visconti}@dia.unisa.it

Abstract. In this paper we revisit previous work in the BPK model and point out subtle problems concerning security proofs of concurrent and resettable zero knowledge (c$\mathcal{ZK}$ and r$\mathcal{ZK}$, for short). Our analysis shows that the c$\mathcal{ZK}$ and r$\mathcal{ZK}$ simulations proposed for previous (in particular *all* round-optimal) protocols are distinguishable from real executions. Therefore some of the questions about achieving round optimal c$\mathcal{ZK}$ and r$\mathcal{ZK}$ in the BPK model are still open. We then show our main protocol, $\Pi_{c\mathcal{ZK}}$, that is a round-optimal concurrently sound c$\mathcal{ZK}$ argument of knowledge (AoK, for short) for **NP** under standard complexity-theoretic assumptions. Next, using complexity leveraging arguments, we show a protocol $\Pi_{r\mathcal{ZK}}$ that is round-optimal and concurrently sound r$\mathcal{ZK}$ for **NP**. Finally we show that $\Pi_{c\mathcal{ZK}}$ and $\Pi_{r\mathcal{ZK}}$ can be instantiated efficiently through transformations based on number-theoretic assumptions. Indeed, starting from any language admitting a perfect Σ-protocol, they produce concurrently sound protocols $\bar{\Pi}_{c\mathcal{ZK}}$ and $\bar{\Pi}_{r\mathcal{ZK}}$, where $\bar{\Pi}_{c\mathcal{ZK}}$ is a round-optimal c$\mathcal{ZK}$AoK, and $\bar{\Pi}_{r\mathcal{ZK}}$ is a 5-round r$\mathcal{ZK}$ argument. The r$\mathcal{ZK}$ protocols are mainly inherited from the ones of Yung and Zhao [31].

1 Introduction

The notion of concurrent zero knowledge (c$\mathcal{ZK}$, for short) introduced in [11] deals with proofs given in asynchronous networks controlled by the adversary.

In [3] Canetti et al. studied the case of an adversary that can reset the prover, forcing it to re-use the same randomness in different executions. They defined as resettable zero knowledge (r$\mathcal{ZK}$, for short) the security of a proof system against such attacks. Very interestingly, r$\mathcal{ZK}$ is proved to be stronger than c$\mathcal{ZK}$.

Motivated by the need of achieving round-efficient r$\mathcal{ZK}$, in [3] the Bare Public-Key (BPK, for short) model has been introduced, with the goal of relying on a setup assumption that is as close as possible to the standard model. Indeed, round-efficient c$\mathcal{ZK}$ and r$\mathcal{ZK}$ are often easy to achieve in other models (e.g., with trusted parameters) that unfortunately are hard to justify in practice.

The BPK *model.* The sole assumption of the BPK model is that when proofs are played, identities of (polynomially many) verifiers interacting with honest provers are fixed. Identities have to be posted to a public directory before proofs start. This registration phase is non-interactive, does not involve trusted parties

or other assumptions, and can be fully controlled by an adversarial verifier. When proofs starts, it is assumed that honest provers interact with registered verifiers only. The BPK model is very close to the standard model, indeed the proof phase does not have any requirement beyond the availability of the directory to all provers, and for verifiers, of a secret key associated to their identities. Moreover, in both phases the adversary has full control of the communication network, and of corrupted players.

The first constant-round r$\mathcal{ZK}$ argument for **NP** in the BPK model has been given in [3]. Then in [18] it is pointed out the subtle separations among soundness notions in the BPK model. Indeed, in contrast to the standard model, the notions of one-time, sequential and concurrent soundness, are distinct in the BPK model. In [18] it is then proved that the protocol of [3] is actually sequentially sound only. Moreover in [18] it is proven that 4 rounds are necessary for concurrent soundness and finally, they showed a 4-round r$\mathcal{ZK}$ argument with sequential soundness. In light of the impossibility proved by [1] (i.e., there exists no 3 round sequentially sound c$\mathcal{ZK}$ conversation-based argument in the BPK model for non-trivial languages) the above 4-round r$\mathcal{ZK}$ argument is round optimal. Concurrent soundness along with r$\mathcal{ZK}$ was achieved in [7], requiring 4 rounds. Further improvements on the required complexity assumptions have been showed in [31] where a 4-round protocol under generic assumptions and an efficient 5-round protocol under number-theoretic assumptions are shown. All previously discussed results on constant-round r$\mathcal{ZK}$ in the BPK model relied on the assumptions that some cryptographic primitives are secure against sub-exponential time adversaries (i.e., complexity leveraging) and obtained black-box simulation.

The question of achieving a constant-round black-box c$\mathcal{ZK}$ argument of knowledge (AoK, for short) in the BPK model without relying on complexity leveraging has been first addressed in [32] and then in [9]. The protocol of [32] needs 4 rounds and enjoys sequential soundness. The protocol given in [9] needs only 4 rounds and enjoys concurrent soundness. A follow up result of [27] showed an efficient transformation that starting from a language admitting a Σ-protocol produces a c$\mathcal{ZK}$ AoK with concurrent soundness needing only 4 rounds and adding only a constant number of modular exponentiations. A more recent result [6] obtains both round optimality and optimal complexity assumptions (i.e., OWFs) in a concurrently sound c$\mathcal{ZK}$ AoK. More sophisticated notions of arguments of knowledge have been given in [10] and in [29,28]. Indeed these papers focus on concurrent knowledge extraction (under different formulations). All above results achieving c$\mathcal{ZK}$ are based on hardness assumptions with respect to polynomial-time adversaries.

1.1 Our Results and Techniques

In this paper we show subtle problems concerning security proofs of various c$\mathcal{ZK}$ and r$\mathcal{ZK}$ arguments in the BPK model [18,32,7,9,27,31,6,29], including *all* round-optimal constructions published so far.

The source of the problem: parallel execution of different sub-protocols. In order to achieve round efficiency, various known protocols, including all round-optimal protocols, consist in parallel executions of sub-protocols that are useful in different ways in the proofs of soundness and $\mathsf{c}\mathcal{ZK}/\mathsf{r}\mathcal{ZK}$. Roughly speaking, there is always a sub-protocol π_0 where in 3 rounds the verifier is required to use a secret related to its identity. Then there is a 3-round sub-protocol π_1 in which the prover convinces the verifier about the validity of the statement and the simulator can do the same by using knowledge of a secret information obtained by rewinding π_0 (in this session or in other sessions corresponding to the same identity). To obtain a 4-round protocol[1], π_1 starts during the second round of π_0. Such round combination yields the following two cases.

First we consider the case in which the simulator needs the extracted secret in order to play the first message of π_1 so that such a message can appear in the final transcript of the simulation. In this case when the simulator needs to run π_1 for the first time with a given identity, it needs first to obtain some secret information by the verifier in π_0. The use of look-ahead threads (i.e., trying to go ahead with a virtual simulation with the purpose of obtaining the required information needed in the main thread of the simulation) would not help here since only a limited polynomial amount of work can be invested for them, and there is always a non-negligible probability that look-ahead threads fail, while in the main thread the verifier plays the next message. Given the above difficulty, the simulator needs to play a *bad* first round in π_1 so that later, when the needed secret information is obtained, the simulator can play again the second round of the protocol, this time playing a *good* first round in π_1. However, this approach suffers of a problem too. Indeed, stopping the main thread and trying to start and complete a new thread leads to a detectable deviation in the final transcript that the simulator will output. Indeed, the fact that the simulator gives up with a thread each time it is stuck, and then starts a new one, as we shall see later, modifies the distribution of the output of the simulator, since the output will then include with higher probability threads that are "easier" to complete (e.g., where the simulator does not get stuck because new sessions for new identities do not appear). Notice that this issue motivates the simulation strategies adopted in previous work on $\mathsf{c}\mathcal{ZK}$ (e.g., [25,23]) where the main thread corresponds to the construction of the view that will be given in output, while other threads are started with the sole purpose of extracting secrets useful to go ahead in the main thread. Similar issues concerning the use of a main thread during the whole simulation have been recently considered in [21] for analyzing previous work on selective decommitments.

We now consider the second case where the simulator does not need any secret to compute the first round of π_1. We observe that this approach could hurt the proof of concurrent soundness, when the latter is proved by means of witness extraction. Indeed, a malicious concurrent prover can exploit the execution of π_0 in a session j, for completing the execution of π_1 in another concurrent session $j' \neq j$ by playing a man-in-the-middle attack such that, when (in the proof

[1] Similar discussions hold for some 5-round protocols when π_0 requires 4 rounds.

of concurrent soundness) one tries to reach a contradiction by extracting the witness from the proof π_1 given in session j', it instead obtains the secret used to simulate π_0 in session j. Instead, if the secret to be extracted from π_1 is fixed from the very first round of π_1, then one can show that it is either independent from the one used in session j (this happens when the secret is used in π_0 of session j after the first round of π_1 in session j' is played), or is dependent but not affected by the rewind of the extraction of session j' (this happens when the secret is used in π_0 of session j before the first round of π_1 in session j' is played).

The use of the secret in the last round of π_1 only, could instead be helpful in the following three cases: I) when one is interested in $\mathsf{r}\mathcal{ZK}$ since in this case soundness is proved through a reduction based on complexity leveraging; II) when $\mathsf{c}\mathcal{ZK}$ with sequential soundness only is desired; III) when the secret needed by the simulator when running π_1 in a session j' is different from the witness used by the verifier in the execution of π_0 in the other sessions. Indeed, in those cases the above discussion does not necessarily apply, and indeed some proposed round-optimal protocols might be secure (see discussion in Section 2), even though their security proofs seem to ignore at least in part the problems that we are pointing out.

Because of the above case I, we believe that achieving 4-round $\mathsf{c}\mathcal{ZK}$ with concurrent soundness in the BPK model under standard assumptions is definitively harder[2] than obtaining 4-round $\mathsf{r}\mathcal{ZK}$ with concurrent soundness in the BPK model through complexity leveraging. This is the reason why we mainly concentrate on achieving $\Pi_{\mathsf{c}\mathcal{ZK}}$ and this will require a new technique. Instead, to obtain $\Pi_{\mathsf{r}\mathcal{ZK}}$, we will just rely on a previous protocol given in [31] and make some minor variations in order to recycle part of the analysis given for $\Pi_{\mathsf{c}\mathcal{ZK}}$.

We stress that in all previous constructions, one could obtain a different protocol that satisfies the desired soundness and zero-knowledge properties by simply running π_0 and π_1 sequentially. Indeed, in this case the simulator can complete π_0 in the main thread, then can run the extractor in another thread, and finally can continue the main thread running π_1 having the secret information. We also stress that all papers that we revisit in this work, achieved also other results that are not affected by our analysis.

We finally note that we did not investigate other round-efficient results in variations of the BPK model [17,33,8], and other results in the BPK model that do not focus on (almost) optimal round complexity [20,30,4].

New techniques for round-optimal $\mathsf{c}\mathcal{ZK}$ *and* $\mathsf{r}\mathcal{ZK}$ *in the* BPK *model.* In the main contribution of this paper we show a protocol and a security proof that close the gap in between lower and upper bounds for the round complexity of concurrently sound $\mathsf{c}\mathcal{ZK}\mathsf{AoK}$ in the BPK model. The result is achieved by using a new technique where in addition to the secret of the verifier corresponding to her identity, there is a temporary secret per session that enables the simulator to

[2] However, when we will then focus on efficient instantiations, we will obtain a 4-round protocol for $\mathsf{c}\mathcal{ZK}$ while $\mathsf{r}\mathcal{ZK}$ will require 5 rounds.

proceed in two modes. Indeed, knowledge of the permanent secret of the verifier allows the simulator to proceed in straight-line in the main thread in sessions started after the extraction of the permanent secret. We show that temporary secrets allow the simulator to proceed with the main thread even for sessions started before the extraction of such secrets.

We implement this technique by means of trapdoor commitments. The proof of $\mathsf{c}\mathcal{ZK}$ will be tricky since it requires the synergy of the two above simulation modes. Each time an extraction procedure is started, the simulator is straight-line in the new thread, and aborts in case an unknown secret key is needed to proceed. Essentially, we can show that the number of extraction procedures of temporary and permanent secret keys correspond to the number of sessions[3]. The proof of concurrent soundness also requires special attention. Indeed while the interplay of temporary and permanent secrets helps the simulator, it could also be exploited by the malicious prover.

Our specifically designed protocol $\Pi_{\mathsf{c}\mathcal{ZK}}$ is a round-optimal concurrently sound black-box perfect $\mathsf{c}\mathcal{ZK}\mathsf{AoK}$ for **NP**, under standard complexity-theoretic assumptions. Then, we show that by using complexity leveraging (and thus assuming the existence of complexity-theoretic primitives secure against sub-exponential time adversaries) a variation of a previous protocol due to Yung and Zhao [31] produces a protocol $\Pi_{\mathsf{r}\mathcal{ZK}}$ that is black-box $\mathsf{r}\mathcal{ZK}$, round-optimal and concurrently sound for **NP**. The variations with respect to the work of [31] allow us to recycle part of the analysis used for $\Pi_{\mathsf{c}\mathcal{ZK}}$. Indeed, as we show in Section 2.2, although fixable, the security proof provided in [31] relies on a simulator that outputs a transcript that is distinguishable from the real execution.

We then show that $\Pi_{\mathsf{c}\mathcal{ZK}}$ and $\Pi_{\mathsf{r}\mathcal{ZK}}$ admit efficient transformations that starting from any language admitting a perfect Σ-protocol, produce concurrently-sound protocols $\bar{\Pi}_{\mathsf{c}\mathcal{ZK}}$ and $\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$, where $\bar{\Pi}_{\mathsf{c}\mathcal{ZK}}$ is a round-optimal black-box perfect $\mathsf{c}\mathcal{ZK}\mathsf{AoK}$, while $\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$ is a 5-round black-box $\mathsf{r}\mathcal{ZK}$ argument. Both transformations only require a constant number of modular exponentiations, and $\bar{\Pi}_{\mathsf{c}\mathcal{ZK}}$ is secure under standard number-theoretic assumptions, while $\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$ also needs number-theoretic assumptions w.r.t. sub-exponential time adversaries. $\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$ will again correspond to a variation of a protocol presented in [31].

It is plausible that motivated by different purposes one can get more general constructions or constructions with better efficiency, assumptions or security, but this is out of the scope of this work.

Notation and tools. We denote by $n \in \mathbb{N}$ the security parameter and by PPT the property of an algorithm of running in probabilistic polynomial-time. We assume confidence with the concepts of witness indistinguishability (WI) and of proof of knowledge. A Σ-protocol $(\mathtt{pok}_1, \mathtt{pok}_2, \mathtt{pok}_3)$ is a 3-round public-coin WI proof of knowledge enjoying the honest-verifier zero knowledge property (HVZK), that is, there exists a PPT simulator that on input the theorem to be proved and the message $\mathtt{pok}_2$, outputs a transcript that is indistinguishable from the transcript

[3] This contrasts with the main technique used in the past in the BPK model, where the extraction procedure were applied only for the identities registered in the directory.

output by the prover. If the output is perfectly indistinguishable the Σ-protocol is called *perfect*. We call *special* a Σ-protocol in which the prover can compute the message $\mathtt{pok}_1$ without knowing the theorem to be proved. We refer to [5] for details on Σ-protocols and to [16] for details on *special* Σ-protocols.

2 Issues in Security Proofs of Previous Results

We now show issues in the proofs of $\mathsf{c}\mathcal{ZK}$ and $\mathsf{r}\mathcal{ZK}$ of known protocols.

2.1 The Case of Π_{MR} [18]

Description of Π_{MR}. In [18], it is shown a 4-round $\mathsf{r}\mathcal{ZK}$ argument with sequential soundness, Π_{MR}, in the BPK model. The identity of the verifier $\mathcal{V}$ is a public-key $\mathtt{pk}$ for a semantically secure encryption scheme, and the secret key $\mathtt{sk}$ is the corresponding private key. In the 1st round, $\mathcal{V}$ sends an encryption c under $\mathtt{pk}$ of a random string $\sigma_{\mathcal{V}}$. The prover $\mathcal{P}$ sends in the 2nd round a random string $\sigma_{\mathcal{P}}$. In the 3rd round $\mathcal{V}$ sends $\sigma_{\mathcal{V}}$ and the randomness used to compute c. Moreover in these first 3 rounds, $\mathcal{V}$ proves to $\mathcal{P}$ knowledge of $\mathtt{sk}$ using Blum's protocol for Hamiltonicity [2]. In the 4th round $\mathcal{P}$ sends a non-interactive zero knowledge (NIZK, for short) proof [12] on string $\sigma = \sigma_{\mathcal{V}} \oplus \sigma_{\mathcal{P}}$ proving that $x \in L$.

The proof of $\mathsf{r}\mathcal{ZK}$ for Π_{MR}. The simulator S discussed in [18] (see also [24]) for Π_{MR} goes as follows. It runs the extractor associated to the proof of knowledge, therefore obtaining $\mathtt{sk}$. Then, it can run in straight-line since the encryption c of $\sigma_{\mathcal{V}}$ can be decrypted using $\mathtt{sk}$, and thus S can choose $\sigma_{\mathcal{P}}$ so that the resulting σ corresponds to the fake random string generated by the NIZK simulator. Then S can complete the protocol running in the 4th round the NIZK simulator.

The above simulation produces a transcript that is distinguishable from the one generated by an honest prover. Indeed, we can give two interpretations to the above simulation and in both cases there exists a successful adversary.

Case 1. The first interpretation is to assume that the extraction of $\mathtt{sk}$ is performed in a look-ahead thread that is played before the main thread (where the simulator computes the actual messages to be given in output). In this case, notice that the proof of knowledge of $\mathtt{sk}$ could be completed by V^* with some probability p unknown to S (S is black-box, and there can be different adversarial verifiers using different values for p, and some of them can be negligible). Therefore, since the attempt of S to extract $\mathtt{sk}$ can not be (in order to have an expected polynomial time simulation) unlimited in time, S must give up if after some polynomial effort $\mathtt{sk}$ has not been extracted. When such a look-ahead thread is aborted, then S continues the main thread and it can happen with non-negligible probability p (since S stopped after a polynomial number of attempts) that V^* completes the proof of knowledge of $\mathtt{sk}$. Since in this case S has already played the second round σ_p, the outcome σ of the coin flipping does not allow S to complete the protocol; if one gives in output such a failure, then the transcript of the simulation would be easily distinguishable. S will therefore

need to abort this main thread and start a new one, having now `sk` as input. The problem in this case corresponds to Case 2 below.

Case 2. The second interpretation consists in assuming that once the verifier completes the proof of knowledge, then S solves the identity by running the extractor of the proof of knowledge, therefore obtaining the secret in time roughly $\texttt{poly}(n)/p$, where p is the probability that $\mathcal{V}$ completes the proof of knowledge. Once the secret key is obtained, S can rewind the verifier and start the proof phase of the simulation from scratch, without changing the key generation phase. S now using knowledge of the secret key can complete in straight-line all sessions that correspond to that solved identity.

The above approach is often used in literature in the BPK model and consists therefore in dividing the simulation in phases. Each time the current phase is not completed in straight-line, an extraction is performed, one more identity is solved and then a new phase is started. Since the number of identities is polynomial, at some point there will be a phase that can be executed in straight-line by the simulator. We show now that this approach is affected by a subtle problem. Indeed the approach of S in this case follows the standard procedure of [14] for the case of stand-alone zero knowledge. Here however, a concurrent malicious verifier V^* can nest polynomially many other sessions each one corresponding to a different identity, and each one using a different abort probability.

Consider the simple case of V^* that only runs two nested sessions, corresponding to two different identities and such that in each session the 3rd round is played with probability $1/2$, adaptively to the transcript so far (i.e., this can be easily done by assuming that the coins used for such a probability are taken from the output of a PRF on input the transcript so far and a seed hardwired in V^*). The nesting is performed by including the whole execution of the 2nd session in between the 2nd and 3rd round of the first session. The view of V^* in the real game with probability $1/4$ includes the two sessions both aborted.

Instead, the output of S will be computed as follows. First of all, it can happen that the simulation is straight-line when V^* aborts in both sessions, and this event happens with probability $1/4$. Then, it can happen that the second session is aborted (probability $1/2$) and the first one is not aborted (probability $1/2$). In this case S performs the extraction of the secret from the first session, and once this is done, since it can not continue the previous execution (in the previous execution σ has been already computed and does not allow S to finish the protocol), it will have to start a new phase, this time having the secret key of the first identity as input. However notice that in this new phase (that happens with probability $\frac{1}{4}$), it can happen that both executions abort since the messages sent by S are different, and therefore the coins used by V^* to decide whether to abort or not, will be computationally independent. Since when an execution starts the case of getting two aborts happens with probability $1/4$, and since this new phase of S starts with probability $1/4$, we have that this produces in the output of S both sessions aborted with probability $\frac{1}{16} = \frac{1}{4} \cdot \frac{1}{4}$. Therefore we have that with probability at least $\frac{5}{16} = \frac{1}{4} + \frac{1}{16}$ the simulator outputs a transcript where both sessions are aborted. Given that in the real game this probability is

only 1/4, we have that the output of the simulator is trivially distinguishable. For simplicity in the above analysis we have ignored the fact that V^* uses a PRF instead of independent coins.

Given the above explicit attack, one might wonder if the protocol is anyway valid and a different simulator or a different interpretation of S can be used to prove the same theorem. Indeed, the above attack is certainly addressable with a slightly more sophisticated ad-hoc simulator. However other more sophisticated attacks can easily hurt the new simulation strategy, as in a cat and mouse game where given an adversary one can find a valid simulator for it; but given the valid simulator for that adversary one can find another adversary that requires another simulator. It is not clear at all whether one can finally design a simulator that works against any adversary, as required by the definition of black-box zero knowledge.

The above difficulties[4] are not an issue when considering the simulators for concurrent zero knowledge [25,23] that indeed use the following strategy: the simulator starts a main thread that is updated with new messages exchanged with V^*; other threads are started only to allow the main thread to proceed successfully, but no thread ever replaces the main thread. This is a well understood strategy that we will also use in our constructions. It however will require a new technique to design a protocol where new threads can help the execution of the main thread (this is precisely the problem of some of previous constructions). The strategy of [25] is actually based on starting look-ahead threads, and the large round complexity tolerates failures of look-ahead threads.

The same attack can be replicated to all other simulators. We have given details to explain the problem with the simulation of Π_{MR}, and under minor variations, all other results [18,32,7,9,27,31,6,29] suffer of similar problems. We will now focus on protocols that however could have a different simulation.

2.2 Replacing Simulation in Phases by Threads

We now discuss 4 previous protocols that besides the issues in the proposed simulation strategies discussed above, seem (in some cases with some fixes) still to be able to admit a simulation strategy based on maintaining a main thread. We stress that later we will show a new technique based on the use of temporary keys along with permanent keys so that the simulator works in two modes that allow it to stick with the main thread. Our technique was never used in previous papers. Protocols below when using a different simulation strategy (in some cases, our new simulation strategy) can potentially achieve some of the 4 results that we will achieve in the next sections. We did not go through details of the proofs of such (in some cases, fixed) 4 protocols. Summing up, we do not claim their security and here we only explain how such protocols and their (distinguishable) simulations in phases could potentially be adjusted in light of our results and techniques.

[4] We stress that such difficulties disappears if round optimality is not needed.

The Case of Π_Z [32]. A 4-round conversation-based $\mathtt{c}\mathcal{ZK}$ argument enjoying sequential soundness only is shown in [32]. While the security proof still relies on the use of a simulator that works in phases, we notice that a different simulator based on keeping a main thread could be used instead. The reason, is that the secret information is needed by the simulator only in the 3rd round of π_1 (see our discussion in Section 1) and, since the achieved result is only sequentially sound, there is no concurrent attack to soundness to take care of.

The Case of Π_{YZ} [31]. In [31], Yung and Zhao showed protocols Π_{YZ} and $\bar{\Pi}_{YZ}$ that are respectively a 4-round concurrently sound $\mathtt{r}\mathcal{ZK}$ argument in the BPK model under general complexity-theoretic assumptions and an efficient 5-round concurrently sound $\mathtt{r}\mathcal{ZK}$ argument under number theoretic assumptions. Both protocols use complexity leveraging and we will now concentrate on Π_{YZ} since the analysis extends also to $\bar{\Pi}_{YZ}$ with one more round.

Π_{YZ} consists of 3 sub-protocols played in parallel. In the first three rounds the verifier, using a special Σ-protocol $\Sigma^{\mathtt{fls}}$, gives a proof of knowledge of its secret key $\mathtt{sk}$ or of a solution of a puzzle. The puzzle was sent by the prover during the second round, and $\Sigma^{\mathtt{fls}}$ is such that knowledge of the theorem (and therefore of the witness) is not required in the first round. The prover gives a resettable WI proof (i.e., the verifier commits to the challenge in the first round) in rounds 2, 3 and 4 where it proves that $x \in L$ or it knows $\mathtt{sk}$. Since black-box extraction of the witness (necessary for the proof of concurrent soundness) is not allowed in the resetting verifier setting, they enforce the extraction using complexity leveraging as follows. The challenge is committed through a trapdoor commitment scheme with a 2-round decommitment phase, where the trapdoor, that is needed only in the opening, corresponds to the solution of the puzzle sent by the prover. Therefore there exists a sub-exponential time extractor that can find the solution of the puzzle, open the commitment in multiple ways and thus extract the actual witness of the prover. This proof of concurrent soundness falls down when one would like to use standard hardness assumptions only (e.g., to prove $\mathtt{c}\mathcal{ZK}$ under standard assumptions). The technical difficulty of implementing efficiently $\Sigma^{\mathtt{fls}}$ is solved by requiring the prover to send the puzzle in a first round, so that an OR composition of Σ-protocols can be used, therefore obtaining a 5-round protocol $\bar{\Pi}_{YZ}$.

As discussed in [31] (see page 136), the simulator runs in different phases, trying in each phase to complete the simulation, but in case it can not, it obtains a new secret key and starts a new phase, with new randomness. This approach as previously discussed makes the output of the simulator distinguishable when playing with some specific adversarial concurrent verifiers. However, we notice that in this case an alternative simulation strategy could be possible. Indeed, when the simulator starts the main thread and gets stuck, it does not actually need to abort it, but instead can start a new thread just to get the secret information to complete the main thread. The reason why this can be possible here (in contrast to previous protocols), is that the simulator needs the secret of the verifier only when it plays the last message of the protocol, therefore it can always perform the extraction (in a new thread) before being stuck. However, as

discussed in the introduction, playing the extracted secret only in the last round exposes the protocol to concurrent soundness attacks. In the very specific case of $\mathsf{r}\mathcal{ZK}$, since soundness is proved through complexity leveraging, the proof of soundness could go through.

The Case of Π_{YYZ} [29,28]. In the concurrently sound $\mathsf{c}\mathcal{ZK}$ protocol presented in [29,28], the simulator is required to commit in the second round to one of the two secret keys of the verifier. This must be done before the verifier completes its proof of knowledge of one of her secret keys. It is immediate to see that precisely as we discussed above, this requires the simulator to try to complete the simulation using new phases (see page 24 of [28]). Therefore the same attacks showed before can be mounted against this simulator too.

In Section 6.2 of [28] an update of the protocol yielding round optimality is suggested. The update consists in replacing a strong WI proof with a 4-round zero-knowledge AoK due to Feige and Shamir [13] (FSZK, for short) such that this protocol can share the statistical WI proof of knowledge given by the verifier. However, in the same section it is then observed that such update hurts the concurrent soundness of their scheme.

Here we observe that since their first subprotocol is a statistical WI argument of knowledge given by the verifier, it can be instantiated under general complexity-theoretic assumptions only requiring a first round from prover to verifier. Indeed this message is needed to establish the parameters for a statistically hiding commitment scheme to be used in the statistical WI argument. Therefore, the resulting construction can be round optimal only when using number-theoretic assumptions, that can be used to implement the statistical WI proof in 3 rounds [26,5].

Our technique based on temporary keys and simulation in two modes can potentially be applied when using FSZK differently, so that concurrent soundness could be preserved. This could be possible when FSZK is played independently of the public keys of the verifier, therefore including some session keys (which would have a role similar to the temporary keys of our technique). Then our new simulation technique could be used to maintain a main thread working in two modes (in one mode using the extracted permanent keys, in the other mode using the simulator of FSZK that use the extracted session keys).

The Case of Π_D [6]. A 4-round concurrently sound $\mathsf{c}\mathcal{ZK}$ argument Π_D in the BPK model under the existence of one-way functions only is showed in [6]. In the first round, the verifier sends a message m_v. Then in the second round the prover sends a statistically binding commitments of potential signatures of messages (under the public-key of the verifier) and a message m_p. In the third round the verifier sends a signature of $(m_v|m_p)$ (instead of the usual proof of knowledge of a secret). In the last 3 rounds $\mathcal{P}$ proves that $x \in L$ or the commitment sent in the second round corresponds to messages $(m'|m'_0)$ and $(m'|m'_1)$ and their signatures, where $m'_0 \neq m'_1$.

Of course since the concurrent adversarial prover can not rewind the verifier, the above argument is sufficient to prove concurrent soundness. Indeed,

signatures received in concurrent proofs always correspond to messages with a different prefix selected by the verifier. The $\mathsf{c}\mathcal{ZK}$ property of the protocol however is problematic again for the very same reasons discussed above. Indeed, the simulator does not have any signature at all when it plays the second round, and thus later on, in order to be able to complete proofs it will have to start new phases where knowledge of the signatures accumulated during previous executions is sufficient to run in straight-line. Indeed, the simulator presented in [6] rewinds the verifier when it is stuck, and produces a new transcript committing to the extracted signatures. As already explained, this makes distinguishable its output w.r.t. real executions.

We finally argue that the protocol could be adjusted to then admit a simulator that keeps a main thread. In contrast to previously discussed protocols, the main advantage of Π_D is that the verifier uses his secret keys to generate signatures of messages with different formats in different sessions. This makes problematic the attack of concurrent soundness, since the execution of concurrent sessions does not provide useful messages to cheat in a specific session. Therefore one could tweak the protocol so that the simulator needs to use the obtained signatures only at the last round. In this way, the simulator could obtain through rewinds two signatures for messages with the same structure, and could use them in the main thread and in all future sessions that correspond to that verifier.

3 Round-Optimal $\mathsf{c}\mathcal{ZK}$ and $\mathsf{r}\mathcal{ZK}$ in the BPK Model

We show under standard complexity-theoretic assumptions round-optimal concurrently sound $\mathsf{c}\mathcal{ZK}$AoK and a $\mathsf{r}\mathcal{ZK}$ argument with complexity leveraging.

3.1 Concurrent Zero Knowledge in the BPK Model

Overview, techniques and proof intuition. In light of the attacks shown in the previous section, we construct a protocol that allows a simulation strategy in which the transcript generated in main thread is kept unchanged. In the following we describe the protocol incrementally.

The public identity of the verifier $\mathcal{V}$ corresponds to a pair of public keys $\mathtt{pk}_0 = f(\mathtt{sk}_0), \mathtt{pk}_1 = f(\mathtt{sk}_1)$, where f is a one-way function, for which $\mathcal{V}$ knows one of the pre-images $\mathtt{sk}_b$, that we call the secret key. Following the usual paradigm used in the BPK model, we require that $\mathcal{V}$ provides a proof of knowledge, using a Σ-protocol, that we denote by $\Sigma^{\mathtt{pk}_j}$, of the secret key associated to the public identity $\mathtt{pk}_j = (\mathtt{pk}_0, \mathtt{pk}_1)$. In the second round, the prover $\mathcal{P}$ first commits to a bit (representing the selection of one of $\mathcal{V}$'s public keys), then it provides a proof that either $x \in L$ or it knows the secret corresponding to the public key selected in the commitment, using a Σ-protocol as well, which we denote by $\Sigma^{\mathtt{L}_j}$. Requiring that $\mathcal{P}$ selects the key already in the first message allows to use the witness-indistinguishability property of Σ-protocols and the binding property of the commitment scheme to prove the concurrent-soundness property.

A simulator for this protocol would extract the secret (by exploiting the proof of knowledge property of $\Sigma^{\mathtt{pk}_j}$) and would complete the protocol $\Sigma^{\mathtt{L}_j}$ using the extracted secret key as witness. However, this step is done without changing the commitment sent in the first message only if S has committed to the bit corresponding to the extracted secret key. Instead, if this is not the case, S has to rewind the verifier and change the commitment, therefore changing the transcript of the main thread, that is precisely the problem of previous works.

We overcome this problem by tweaking the protocol in two ways. First, we require that upon each new execution, $\mathcal{V}$ freshly generates a pair of public parameters and the respective trapdoors for a two-round trapdoor commitment scheme. Such parameters can be seen as temporary keys. $\mathcal{V}$ then sends the public parameters to $\mathcal{P}$ and proves knowledge of one of the trapdoors running an additional Σ-protocol that we denote by $\Sigma^{\mathtt{trap}}$. This will allow the simulator to extract the trapdoor. Second, we require that $\mathcal{P}$, instead of sending the first message of $\Sigma^{\mathtt{L}_j}$ in clear, it sends a trapdoor commitment of it, using both public parameters received from $\mathcal{V}$, i.e., $\mathcal{P}$ computes two commitments, each one with a distinct parameter, of two shares of the first message. The shares are revealed only in the third round of $\Sigma^{\mathtt{L}_j}$, precisely only after $\mathcal{P}$ has seen the challenge for $\Sigma^{\mathtt{L}_j}$ sent by $\mathcal{V}$. Intuitively, due to the binding of the commitment scheme $\mathcal{P}$ is not able to take advantage of the knowledge of the challenge. Furthermore, since the parameters of the trapdoor commitment are freshly generated by $\mathcal{V}$ upon each execution, due to the witness indistinguishability property of $\Sigma^{\mathtt{trap}}$, $\mathcal{P}$ cannot take advantage of concurrent executions with many verifiers, thus concurrent soundness still holds[5]. Indeed, we are able to prove concurrent soundness by showing a concurrent extractor, that extracts the witness from any accepting transcript obtained by any malicious prover. The guarantee of the witness extraction is necessary for the proof of soundness to go through.

Instead the simulator can use its rewinding capabilities to extract the trapdoor by exploiting the proof of knowledge property $\Sigma^{\mathtt{trap}}$, so that in the main thread it can open the commitments of the first round of $\Sigma^{\mathtt{L}_j}$, according to the challenge received from $\mathcal{V}$. Here, the simulator uses the HVZK property of $\Sigma^{\mathtt{L}_j}$. We stress that the simulator does not change messages previously played in the main thread, i.e., the commitments of the first round of $\Sigma^{\mathtt{L}_j}$, but it cheats only in the third round by equivocating one of the commitments by using the trapdoor extracted in the rewinding thread. Since commitments computed by the prover are perfectly hiding and $\Sigma^{\mathtt{L}_j}$ is a perfect Σ-protocol, the simulation will be perfectly indistinguishable from a real execution.

Note that, in order to prevent the blow-up of the running time, it is crucial that the simulator extracts the trapdoor only for sessions for which it has not extracted the secret key yet. Once a secret corresponding to an identity has been extracted, all sessions played by the malicious verifier with such identity are simulated in straight-line.

[5] If instead parameters of the trapdoor commitments were fixed for all executions, then in the proof of soundness one can not derive a contradiction in case $\mathcal{P}$ equivocates the commitment associated to the same trapdoor used as witness in $\Sigma^{\mathtt{trap}}$.

Formal construction. In the following we provide the formal specification of the $\mathsf{c}\mathcal{ZK}\mathsf{AoK}$ protocol that we denote by $\Pi_{\mathsf{c}\mathcal{ZK}}$.

The public file. Let f be a given one-way function $f : \{0,1\}^{\mathtt{poly}(n)} \to \{0,1\}^*$. The jth identity of the public file F is $\mathtt{pk}_j := (\mathtt{pk}_j^0 = f(\mathtt{sk}_j^0), \mathtt{pk}_j^1 = f(\mathtt{sk}_j^1))$ for some values $\mathtt{sk}_j^0, \mathtt{sk}_j^1 \in \{0,1\}^n$.

Sub-Protocols. Let $\mathtt{TCom} = (\mathtt{TSen}, \mathtt{TRec}, \mathtt{TDec})$ be a two-round perfectly-hiding trapdoor commitment scheme and $\mathtt{PHCom} = (\mathtt{PHSen}, \mathtt{PHRec})$ a two-round perfectly-hiding commitment scheme. For simplicity we assume that the public parameter pk for the scheme $\mathtt{PHCom}$ is included in the identity of $\mathcal{V}$. The parameters' generation procedure is denoted by $pk \leftarrow \mathtt{PHRec}(1^n, r)$ (resp. $(pk, trap) \leftarrow \mathtt{TRec}(1^n, r)$) where r is a random string and n is the security parameter. The commitment procedure $(\mathtt{C}, \mathtt{D}) \leftarrow \mathtt{PHSen}(pk, m)$ (resp. $\mathtt{TSen}$) takes as input the public parameter pk and a message m and outputs the commitment $\mathtt{C}$ and the decommitment $\mathtt{D}$. The verification procedure $\mathtt{PHRec}(pk, \mathtt{C}, \mathtt{D}, m)$ (resp. $\mathtt{TRec}$) outputs 1 if $\mathtt{D}$ is a valid decommitment of $\mathtt{C}$ for the message m, under pk. Finally, $(m', \mathtt{D}) \leftarrow \mathtt{TDec}(trap, \mathtt{C}, m', z)$ is the procedure that allows to open $\mathtt{C}$ as any message using the trapdoor $trap$ and some auxiliary information z inherited from the commitment phase.

Auxiliary languages. We use the following **NP** relations and in turn the respective **NP**-languages $L_{\mathtt{pk}_j}, L_{\mathtt{trap}}, L_{\mathtt{sk}_j}, L_j$:

- $\mathcal{R}_{\mathtt{pk}_j} = \{(\mathtt{pk}_j^0, \mathtt{pk}_j^1), \mathtt{sk})$ s.t. $\mathtt{pk}_j^0 = f(\mathtt{sk})$ OR $\mathtt{pk}_j^1 = f(\mathtt{sk})\}$;
- $\mathcal{R}_{\mathtt{trap}} = \{((k_0, k_1), (t, r))$ s.t. $(k_0, t) \leftarrow \mathtt{TRec}(1^n, r)$ OR $(k_1, t) \leftarrow \mathtt{TRec}(1^n, r)\}$;
- $\mathcal{R}_{\mathtt{sk}_j} = \{((\mathtt{C}, \mathtt{pk}_j^0, \mathtt{pk}_j^1), (d, \mathtt{D}, \mathtt{sk}))$ s.t. $\mathtt{PHRec}(pk, \mathtt{C}, \mathtt{D}, d) = 1 \wedge \mathtt{pk}_d^j = f(\mathtt{sk})\}$;
- $\mathcal{R}_{\mathtt{L}_j} := \mathcal{R}_L \vee \mathcal{R}_{\mathtt{sk}_j} = \{(x,\ \mathtt{C},\ \mathtt{pk}_j^0,\ \mathtt{pk}_j^1),\ (w,\ d,\ \mathtt{D},\ \mathtt{sk}))$ s.t. $(x, w) \in R_L \vee ((\mathtt{C}, \mathtt{pk}_j^0, \mathtt{pk}_j^1), (d, \mathtt{D}, \mathtt{sk})) \in \mathcal{R}_{\mathtt{sk}_j}\}$.

Σ-Protocols. The languages showed above are proved by means of Σ-protocols. We denote by $\Sigma^{\mathtt{pk}_j} = (\mathtt{pok}_1^{\mathtt{pk}_j}, \mathtt{pok}_2^{\mathtt{pk}_j}, \mathtt{pok}_3^{\mathtt{pk}_j})$, $\Sigma^{\mathtt{trap}} = (\mathtt{pok}_1^{\mathtt{trap}}, \mathtt{pok}_2^{\mathtt{trap}}, \mathtt{pok}_3^{\mathtt{trap}})$ the Σ-protocols run by $\mathcal{V}$ with identity $\mathtt{pk}_j = (\mathtt{pk}_j^0, \mathtt{pk}_j^1)$ to prove instances of relations $\mathcal{R}_{\mathtt{pk}_j}$ and $\mathcal{R}_{\mathtt{trap}}$ respectively. We denote by $\Sigma^{\mathtt{L}_j} = (\mathtt{pok}_1^{\mathtt{L}_j}, \mathtt{pok}_2^{\mathtt{L}_j}, \mathtt{pok}_3^{\mathtt{L}_j})$ the perfect Σ-protocol run by $\mathcal{P}$ for instances of $\mathcal{R}_{\mathtt{L}_j}$ when interacting with the verifier with identity $\mathtt{pk}_j$.

The Protocol. The protocol is depicted in Fig. 1. By noticing that Blum's protocol [2] is a perfect HVZK Σ-protocol (when the first message is computed with a perfectly-hiding commitment scheme) for **NP** languages, we conclude that Protocol $\Pi_{\mathsf{c}\mathcal{ZK}}$ is a black-box perfect $\mathsf{c}\mathcal{ZK}\mathsf{AoK}$ for all **NP**.

Theorem 1. *If $\Sigma^{\mathtt{pk}_j}, \Sigma^{\mathtt{trap}}$ are Σ-protocols, $\Sigma^{\mathtt{L_j}}$ is a* perfect *Σ-protocol,* $\mathtt{PHCom}$ *is a two-round perfectly-hiding commitment scheme and* $\mathtt{TCom}$ *is a two-round perfectly-hiding trapdoor commitment scheme then $\Pi_{\mathsf{c}\mathcal{ZK}}$ is a 4-round concurrently sound black-box perfect* $\mathsf{c}\mathcal{ZK}\mathsf{AoK}$ *in the* BPK *model for all* **NP**.

Common input: the public file F, n-bit string $x \in L$ and index j specifying the jth entry of F, i.e. $(\mathtt{pk}_j^0 = f(\mathtt{sk}_j^0), \mathtt{pk}_j^1 = f(\mathtt{sk}_j^1))$. $\mathcal{P}$**'s private input:** a witness w for $x \in L$. $\mathcal{V}$**'s private input:** a randomly chosen secret $\mathtt{sk}_j^b$ between $\mathtt{sk}_j^0$ and $\mathtt{sk}_j^1$.

$\mathcal{V}$**-round-1:**

- $r_0, r_1 \xleftarrow{\$} \{0,1\}^n$, $(k_0, t_0) \leftarrow \mathtt{TRec}(1^n, r_0)$; $(k_1, t_1) \leftarrow \mathtt{TRec}(1^n, r_1)$;
- compute $\mathtt{pok}_1^{\mathtt{pk}_j}$ and $\mathtt{pok}_1^{\mathtt{trap}}$;
- send k_0, k_1, $\mathtt{pok}_1^{\mathtt{pk}_j}$, $\mathtt{pok}_1^{\mathtt{trap}}$ to $\mathcal{P}$.

$\mathcal{P}$**-round-2:**

- $(\mathtt{C}, \mathtt{D}) \leftarrow \mathtt{PHSen}(pk, d)$ for a randomly chosen bit d; compute $\mathtt{pok}_2^{\mathtt{pk}_j}$, $\mathtt{pok}_2^{\mathtt{trap}}$;
- compute $\mathtt{pok}_1^{\mathtt{L}_j}$ and compute shares s_0, s_1 s.t. $s_0 \oplus s_1 = \mathtt{pok}_1^{\mathtt{L}_j}$;
- $(\mathtt{tcom}_0, \mathtt{tdec}_0) \leftarrow \mathtt{TSen}(k_0, s_0)$, $(\mathtt{tcom}_1, \mathtt{tdec}_1) \leftarrow \mathtt{TSen}(k_1, s_1)$;
- send $\mathtt{pok}_2^{\mathtt{pk}_j}$, $\mathtt{pok}_2^{\mathtt{trap}}$, $\mathtt{C}$, $\mathtt{tcom}_0, \mathtt{tcom}_1$ to $\mathcal{V}$.

$\mathcal{V}$**-round-3:**

- compute $\mathtt{pok}_3^{\mathtt{pk}_j}$ using as witness $\mathtt{sk}_j^b$;
- compute $\mathtt{pok}_3^{\mathtt{trap}}$ using as witness t_e, r_e for a randomly selected bit e;
- compute $\mathtt{pok}_2^{\mathtt{L}_j}$;
- send $\mathtt{pok}_3^{\mathtt{pk}_j}, \mathtt{pok}_3^{\mathtt{trap}}, \mathtt{pok}_2^{\mathtt{L}_j}$ to $\mathcal{P}$.

$\mathcal{P}$**-round-4:**

- verify that $(\mathtt{pok}_1^{\mathtt{pk}_j}, \mathtt{pok}_2^{\mathtt{pk}_j}, \mathtt{pok}_3^{\mathtt{pk}_j})$ is an accepting transcript of $\Sigma^{\mathtt{pk}_j}$ for the statement $(\mathtt{pk}_j^0, \mathtt{pk}_j^1) \in L_{\mathtt{pk}_j}$, if not abort;
- verify that $(\mathtt{pok}_1^{\mathtt{trap}}, \mathtt{pok}_2^{\mathtt{trap}}, \mathtt{pok}_3^{\mathtt{trap}})$ is an accepting transcript of $\Sigma^{\mathtt{trap}}$ for the statement $(k_0, k_1) \in L_{\mathtt{trap}}$, if not abort;
- compute $\mathtt{pok}_3^{\mathtt{L}_j}$ using the witness w;
- send $\mathtt{pok}_3^{\mathtt{L}_j}, \mathtt{tdec}_0, \mathtt{tdec}_1, s_0, s_1$ to $\mathcal{V}$.

$\mathcal{V}$**-decision:** if $\mathtt{TRec}(k_0, \mathtt{tcom}_0, \mathtt{tdec}_0, s_0) = 1$ AND $\mathtt{TRec}(k_1, \mathtt{tcom}_1, \mathtt{tdec}_1, s_1) = 1$ then $\mathtt{pok}_1^{\mathtt{L}_j} \leftarrow s_0 \oplus s_1$ and accept iff $(\mathtt{pok}_1^{\mathtt{L}_j}, \mathtt{pok}_2^{\mathtt{L}_j}, \mathtt{pok}_3^{\mathtt{L}_j})$ is an accepting transcript of $\Sigma^{\mathtt{L}_j}$ for the statement $(x, \mathtt{C}, \mathtt{pk}_j^0, \mathtt{pk}_j^1) \in L'_j$; else, abort.

Fig. 1. $\Pi_{\mathsf{c}\mathcal{ZK}}$: 4–round concurrently-sound $\mathsf{c}\mathcal{ZK}\mathsf{AoK}$ in the BPK model for all **NP**.

3.2 Resettable Zero Knowledge in the BPK Model

In this section we discuss the updates to (a simpler version of) $\Pi_{\mathsf{c}\mathcal{ZK}}$ to deal with the resetting power of the adversarial verifier V^*.

To suppress the resetting power of V^* we add the commitment of the challenge $\mathtt{pok}_2^{\mathtt{L}_j}$ in the first round, and we require that the randomness of $\mathcal{P}$ is computed by applying a PRF on the transcript obtained so far. This ensures that on the same prefix of interaction the verifier will always get the same response from the prover. $\mathcal{V}$ will then provide the opening of the commitment in the third

round. Unfortunately, this approach prevents the (black-box) extraction of the witness that we need to prove concurrent soundness.

Thus, to allow extraction we need to resort to complexity leveraging arguments. As mentioned in Section 1.1, the use of such techniques allows one to design round-optimal protocols in which the use of the secret extracted from the malicious verifier can be postponed to the last round, ruling out the issues about the indistinguishability of the transcript pointed out in this work (of course only if the simulation strategy does not work in phases).

Therefore, designing a $\mathsf{r}\mathcal{ZK}$ protocol using complexity leveraging is a much simpler task that does not require the two-mode simulation that we used for $\Pi_{\mathsf{c}\mathcal{ZK}}$. Thus, in $\Pi_{\mathsf{r}\mathcal{ZK}}$ we do not need the use of temporary keys along with protocol $\Sigma^{\mathtt{trap}}$, and the prover sends the first round of $\Sigma^{\mathtt{L}_j}$ in clear (we assume that the witness is used only in the third round of $\Sigma^{\mathtt{L}_j}$), instead of sending a trapdoor commitment of it. Moreover in $\Pi_{\mathsf{r}\mathcal{ZK}}$, we do not need that $\mathcal{P}$ commits to one secret already in the second round, and thus the theorem proved with $\Sigma^{\mathtt{L}_j}$ is that either $\mathcal{P}$ knows the witness for $x \in L$ or it knows one of the secret keys (instead of proving that the opening of the commitment points to one of the secrets keys). Instead we need that $\mathcal{P}$, in the second round, computes and sends a puzzle that is solvable in sub-exponential time. Then, we require that instead of the opening of the commitment, in the third round $\mathcal{V}$ sends only the message $\mathtt{pok}_2^{\mathtt{L}_j}$ and it proves, using again a Σ-protocol that we denote by $\mathtt{FLS}^{\mathtt{com}}$, that either message $\mathtt{pok}_2^{\mathtt{L}_j}$ is the valid opening or it knows the solution of the puzzle. Moreover, while $\Sigma^{\mathtt{trap}}$ disappear, in $\Sigma^{\mathtt{pk}_j}$ the verifier proves knowledge of one of the secret keys or of the solution of the puzzle (this update is necessary for the proof of concurrent soundness giving that the prover does not commit in the second round). Note that to preserve round-optimality, $\mathtt{FLS}^{\mathtt{com}}$ and $\Sigma^{\mathtt{pk}_j}$ must be *special* Σ-protocols [16] since the puzzle, that is part of the theorem, is sent by $\mathcal{P}$ only in the second round.

Obviously any malicious verifier, running in polynomial time is not able to solve the puzzle, and is bound on the challenge committed in the first round (thus the zero-knowledge property is preserved). Instead, the extraction of the witness is possible by running in sub-exponential time and solving the puzzle. When the theorem proved is instead false, the extraction will produce a contradiction, breaking the WI of $\mathtt{FLS}^{\mathtt{com}}$ or $\Sigma^{\mathtt{pk}_j}$, or inverting the one-way function used to produce the public keys. All these primitives are setup with ad-hoc security parameters so that they are secure against adversaries that by exhaustive search can solve the puzzle and check membership of the common instances (i.e., the size of such instances must be known before the experiment starts) of the $\mathsf{r}\mathcal{ZK}$ protocol in the language. The final protocol is a variation of the one proposed in [31].

Theorem 2. *If 2-round perfectly hiding commitments and OWPs secure against sub-exponential time adversaries exist then protocol $\Pi_{\mathsf{r}\mathcal{ZK}}$ is a 4-round $\mathsf{r}\mathcal{ZK}$ argument in the* BPK *model with concurrent soundness for all* **NP**.

4 Efficient Instantiations

Here we show efficient transformations that starting from any language L admitting a perfect Σ-protocol, and adding a constant number of modular exponentiations, produce: 1) a 4-round concurrently sound $\mathsf{c}\mathcal{ZK}\mathsf{AoK}$ in the BPK model $\bar{\Pi}_{\mathsf{c}\mathcal{ZK}}$ based on the Discrete Logarithm (DL) assumption; 2) a 5-round concurrently sound $\mathsf{r}\mathcal{ZK}$ argument in the BPK model $\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$ based on the hardness of the $\mathcal{DDH}$ assumption w.r.t. sub-exponential time adversaries.

Interestingly, both protocols are obtained essentially for free, by properly instantiating the sub-protocols in the constructions of $\Pi_{\mathsf{c}\mathcal{ZK}}$ and $\Pi_{\mathsf{r}\mathcal{ZK}}$. All Σ-protocols used in the following transformations are perfect.

$\bar{\Pi}_{\mathsf{c}\mathcal{ZK}}$. Let (G, p, q, g) such that p, q are primes, $p = 2q+1$ and g is a generator of the only subgroup G of order q of $\mathbb{Z}_p^\star$. The one-way function f used to compute the identities of the public file is instantiated with the DL function. Therefore, $\mathtt{pk}_0 = g^{\mathtt{sk}_0} \pmod p$ and $\mathtt{pk}_1 = g^{\mathtt{sk}_1} \pmod p$, where $\mathtt{sk}_0, \mathtt{sk}_1 \xleftarrow{\$} \mathbb{Z}_q$. An identity also contains a pairs (g, h) of generators of G as parameters for a perfectly-hiding commitment scheme. To prove knowledge of one of the secret keys associated to identity $\mathtt{pk}_j = (\mathtt{pk}_0, \mathtt{pk}_1)$ is sufficient to prove the knowledge of the DL of either $\mathtt{pk}_0$ or $\mathtt{pk}_1$, that can be instantiated with Schnorr's [26] Σ-protocol under OR composition, as discussed in [5]. This is the efficient implementation of $\Sigma^{\mathtt{pk}_j}$.

The trapdoor commitment scheme is instantiated with the scheme proposed by Pedersen [22]. Thus, temporary keys consist of the parameters for Pedersen commitment, i.e., $k_b = (g_b, h_b)$, where $h_b = g_b^{t_b} \pmod p$, and g_b is a generator of G, for $b = 0, 1$ and the corresponding trapdoors are t_0, t_1. We stress that t_0, t_1 are generated on the fly and are not contained in the public file. Thus, $\Sigma^{\mathtt{trap}}$ is instantiated again with Schnorr's protocol under OR composition.

The most interesting part consists in the implementation of protocol $\Sigma^{\mathtt{L}_j}$, more specifically the implementation of the Σ-protocol for the relation $\mathcal{R}_{\mathtt{sk}_j}$. The perfectly-hiding commitment of a bit b (i.e., the commitment $\mathtt{C}$ of Fig. 1) is replaced by the Pedersen commitment of $\mathtt{sk}_b$ computed as $\mathtt{C} = h^r \mathtt{pk}_b$ for some random string r and bit b. Then, to prove that $\mathtt{C}$ corresponds to a commitment of $\mathtt{sk}_0$ or $\mathtt{sk}_1$, $\mathcal{P}$ is required to prove the AND of the following statements: 1) knowledge of the DL of $(\mathtt{C}/\mathtt{pk}_b)$, for some bit b, 2) knowledge of the decommitment of $\mathtt{C}$. Both statements can be proved by efficient Σ-protocols based on DL assumption. Since we have a Σ-protocol for L too, putting everything together, $\Sigma^{L'_j}$ is obtained as the composition of these Σ-protocols by means of AND and OR logic operators. All the above computations require a constant number of modular exponentiations. $\bar{\Pi}_{\mathsf{c}\mathcal{ZK}}$ is secure under the DL assumption.

$\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$. The PRF is implemented by the efficient Naor-Reingold PRF [19] based on $\mathcal{DDH}$ assumption. The commitment $\mathtt{pok}_2^{\mathtt{L}_j}$ is implemented with the El Gamal encryption scheme (based on the $\mathcal{DDH}$ assumption), i.e., the commitment of a string m corresponds to the pair $\mathtt{com} = (u = g_c^r \pmod p, v = h_c^r m \pmod p)$, for a randomly chosen r, where g_c is a generator of G and $h_c = g_c^\beta \pmod p$ for some $\beta \leftarrow \mathbb{Z}_q$. Proving knowledge of the decommitment of $\mathtt{com}$ corresponds

to prove that $(G, g, h, u, v/m)$ is a $\mathcal{DDH}$ tuple. The puzzle can be implemented by using again the DL assumption (obviously the use of complexity leveraging requires to work with groups of appropriate size). Having a Σ-protocol (i.e., Schnorr's protocol) to prove knowledge of a solution for the puzzle, and having a Σ-protocol for $\mathcal{DDH}$ problem [15], $\Sigma^{\mathtt{fls}}$ is implemented as the OR composition of these two Σ-protocols. In $\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$ the protocol $\Sigma^{\mathtt{pk}_j}$ is used for an augmented theorem in which $\mathcal{V}$ proves also knowledge of the solution of the puzzle. Thus in $\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$, the protocol $\Sigma^{\mathtt{pk}_j}$ implemented above is used in OR composition with Schnorr's protocol for DL. However, there is a technicality here. When instantiating $\Sigma^{\mathtt{fls}}$, $\Sigma^{\mathtt{pk}_j}$ with the OR composition protocol as shown in [5], we have that $\mathcal{V}$ needs to know the theorem already when computing the first round. Therefore, as already discussed in Section 2 for the case of $\bar{\Pi}_{YZ}$ the puzzle must be sent in the first round and thus $\bar{\Pi}_{\mathsf{r}\mathcal{ZK}}$ is a 5 round protocol. All the above computations require a constant number of modular exponentiations. The resulting protocol is secure under the $\mathcal{DDH}$ assumption w.r.t. sub-exponential time adversaries. The perfect Σ-protocol has to require the use of the witness in the last round only.

Acknowledgments. We thank the anonymous referees for their comments. Research supported in part by the European Commission through the FP7 programme under contract 216676 ECRYPT II, and done in part at UCLA.

References

1. Alwen, J., Persiano, G., Visconti, I.: Impossibility and Feasibility Results for Zero Knowledge with Public Keys. In: CRYPTO '05. LNCS, vol. 3621, pp. 135–151. Springer (2005)
2. Blum, M.: How to Prove a Theorem So No One Else Can Claim It. In: Proceedings of the International Congress of Mathematicians. pp. 1444–1451 (1986)
3. Canetti, R., Goldreich, O., Goldwasser, S., Micali, S.: Resettable Zero-Knowledge (Extended Abstract). In: STOC '00. pp. 235–244. ACM (2000)
4. Cho, C., Ostrovsky, R., Scafuro, A., Visconti, I.: Simultaneously resettable arguments of knowledge. In: TCC '12. LNCS, Springer (2012)
5. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In: CRYPTO '94. LNCS, vol. 839, pp. 174–187. Springer-Verlag (1994)
6. Di Crescenzo, G.: Minimal Assumptions and Round Complexity for Concurrent Zero-Knowledge in the Bare Public-Key Model. In: COCOON '09. LNCS, vol 5609, pp. 127–137. Springer (2009)
7. Di Crescenzo, G., Persiano, G., Visconti, I.: Constant-Round Resettable Zero Knowledge with Concurrent Soundness in the Bare Public-Key Model. In: CRYPTO '04. LNCS, vol. 3152, pp. 237–253. Springer (2004)
8. Di Crescenzo, G., Persiano, G., Visconti, I.: Improved Setup Assumptions for 3-Round Resettable Zero Knowledge. In: ASIACRYPT '04. LNCS, vol. 3329, pp. 530–544. Springer (2004)
9. Di Crescenzo, G., Visconti, I.: Concurrent Zero Knowledge in the Public-Key Model. In: Proc. of ICALP '05. LNCS, vol. 3580, pp. 22–33. Springer (2005)
10. Di Crescenzo, G., Visconti, I.: On Defining Proofs of Knowledge in the Bare Public Key Model. In: ICTCS '07. pp. 187–198. World Scientific (2007)

11. Dwork, C., Naor, M., Sahai, A.: Concurrent Zero-Knowledge. In: STOC '98. pp. 409–418. ACM (1998)
12. Feige, U., Lapidot, D., Shamir, A.: Multiple Non-Interactive Zero Knowledge Proofs Based on a Single Random String. In: FOCS '90. pp. 308–317. IEEE (1990)
13. Feige, U., Shamir, A.: Zero Knowledge Proofs of Knowledge in Two Rounds. In: CRYPTO' 89. LNCS, vol. 435, pp. 526–544. Springer (1990)
14. Goldreich, O., Kahan, A.: How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. J. Cryptology 9(3), pp. 167–190. (1996)
15. Hazay, C., Lindell, Y.: Efficient Secure Two-Party Protocols Techniques and Constructions. Springer (2010)
16. Lapidot, D., Shamir, A.: Publicly Verifiable Non-Interactive Zero-Knowledge Proofs. In: CRYPTO '90. pp. 353–365. Springer-Verlag (1991)
17. Micali, S., Reyzin, L.: Min-Round Resettable Zero-Knowledge in the Public-Key Model. In: EUROCRYPT '01. LNCS, vol. 2045, pp. 373–393. Springer (2001)
18. Micali, S., Reyzin, L.: Soundness in the public-key model. In: CRYPTO '01. LNCS, vol. 2139, pp. 542–565. Springer (2001)
19. Naor, M., Reingold, O.: Number-Theoretic Constructions of Efficient Pseudo-Random Functions. J. ACM 51(2), pp. 231–262. (2004)
20. Ostrovsky, R., Persiano, G., Visconti, I.: Constant-Round Concurrent Non-Malleable Zero Knowledge in the Bare Public-Key Model. In: Proc. of ICALP (2) '08. LNCS, vol. 5126, pp. 548–559. Springer (2008)
21. Ostrovsky, R., Rao, V., Scafuro, A., Visconti, I.: Revisiting Lower and Upper Bounds for Selective Decommitments. In: Cryptology ePrint Archive, Report 2011/536. (2011)
22. Pedersen, T.P.: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: CRYPTO '91. pp. 129–140. Springer-Verlag (1992)
23. Prabhakaran, M., Rosen, A., Sahai, A.: Concurrent Zero Knowledge with Logarithmic Round-Complexity. In: FOCS '02. pp. 366–375 (2002)
24. Reyzin, L.: Zero-Knowledge with Public Keys, Ph.D. Thesis. MIT (2001)
25. Richardson, R., Kilian, J.: On the Concurrent Composition of Zero-Knowledge Proofs. In: EUROCRYPT '99. LNCS, vol. 1592, pp. 415–431. Springer (1999)
26. Schnorr, C.P.: Efficient Signature Generation for Smart Cards. Journal of Cryptology 4(3), pp. 239–252. (1991)
27. Visconti, I.: Efficient Zero Knowledge on the Internet. In: Proc. of ICALP (2) '06. LNCS, vol. 4052, pp. 816–827. Springer (2006)
28. Yao, A.C.C., Yung, M., Zhao, Y.: Concurrent Knowledge-Extraction in the Public-Key model. ECCC 14(002). (2007)
29. Yao, A.C.C., Yung, M., Zhao, Y.: Concurrent Knowledge Extraction in the Public-Key Model. In: ICALP (1) '10. LNCS, vol. 6198, pp. 702–714. Springer (2010)
30. Yi, D., Feng, D., Goyal, V., Lin, D., Sahai, A., Yung, M.: Resettable Cryptography in Constant Rounds - The Case of Zero Knowledge. In: ASIACRYPT '11. LNCS, vol. 7073, pp. 390–406. Springer (2011)
31. Yung, M., Zhao, Y.: Generic and Practical Resettable Zero-Knowledge in the Bare Public-Key Model. In: EUROCRYPT '07. LNCS, vol. 4514, pp. 129–147. Springer (2007)
32. Zhao, Y.: Concurrent/Resettable Zero-Knowledge with Concurrent Soundness in the Bare Public-Key Model and its Applications. In: Cryptology ePrint Archive, Report 2003/265 (2003)
33. Zhao, Y., Deng, X., Lee, C.H., Zhu, H.: Resettable Zero-Knowledge in the Weak Public-Key Model. In: EUROCRYPT '03. LNCS, vol. 2045, pp. 123–139. Springer (2003)