Characterizing Deterministic-Prover Zero Knowledge

Nir Bitansky* Arka Rai Choudhuri †

Abstract

Randomness is typically thought to be essential for zero knowledge protocols. Following this intuition, Goldreich and Oren (Journal of Cryptology 94) proved that auxiliary-input zero knowledge cannot be achieved with a deterministic prover. On the other hand, positive results are only known in the honest-verifier setting, or when the prover is given at least a restricted source of entropy.

We prove that removing (or just bounding) the verifier's auxiliary input, deterministic-prover zero knowledge becomes feasible:

- Assuming non-interactive witness-indistinguishable proofs and subexponential indistinguishability obfuscation and one-way functions, we construct deterministic-prover zero-knowledge arguments for $\mathsf{NP} \cap \mathsf{coNP}$ against verifiers with bounded non-uniform auxiliary input.
- Assuming also keyless hash functions that are collision-resistant against bounded-auxiliary-input quasipolynomial-time attackers, we construct similar arguments for all of NP.

Together with the result of Goldreich and Oren, this characterizes when deterministic-prover zero knowledge is feasible. We also demonstrate the necessity of strong assumptions, by showing that deterministic prover zero knowledge arguments for a given language imply witness encryption for that language. We further prove that such arguments can always be collapsed to two messages and be made laconic. These implications rely on a more general connection with the notion of predictable arguments by Faonio, Nielsen, and Venturi (PKC 17).

*Tel Aviv University. Email: nirbitan@tau.ac.il. Member of the Check Point Institute of Information Security. Supported by the Alon Young Faculty Fellowship, by Len Blavatnik and the Blavatnik Family foundation, and an ISF grant 18/484.

†Johns Hopkins University. Email: achoud@cs.jhu.edu. This work was done in part when the author was visiting Tel Aviv University and supported by the Check Point Institute of Information Security. The author is also supported in part by DARPA/ARL Safeware Grant W911NF-15-C-0213, NSF Grants CNS-1908181, CNS-1414023, CNS-1814919, NSF CAREER 1942789, Samsung Global Research Outreach award, Johns Hopkins University Catalyst award and the Office of Naval Research Grant N00014-19-1-2294.

1 Introduction

Goldwasser, Micali, and Rackoff [GMR89] founded the concept of *zero-knowledge proofs* on two main elements: *interaction and randomness*. While both interaction and verifier randomness are known to be essential for zero knowledge, the answer as to whether the prover must also be randomized is not as definite. Goldreich and Oren [GO94] showed that prover randomness is essential in order to achieve *auxiliary-input zero-knowledge* for non-trivial languages. According to this notion, motivated by composition [GK96], anything that a verifier can learn from the proof, on top of the auxiliary information z it already possesses, can be efficiently simulated given the same auxiliary information z.

So when is deterministic-prover zero knowledge possible? So far, deterministic prover zero knowledge have only been shown to exist in the *honest-verifier setting*. Here Faonio, Nielsen, and Venturi [FNV17] proved that any NP language $\mathcal{L}$ that has a *witness encryption scheme* [GGSW13], also has a deterministic-prover honest-verifier (perfect) zero-knowledge argument, or proof, if the language $\mathcal{L}$ has a *hash proof system* [CS02]. A similar result was recently shown by Dahari and Lindell [DL20]. In the same work, Dahari and Lindell also show a statistically sound honest-verifier zero knowledge protocol with an unbounded honest prover for all of NP assuming doubly-enhanced injective one-way functions. In the malicious verifier setting, they give a protocol satisfying a non-standard distributional notion of zero knowledge. In their definition, the prover has access to a pair of witnesses sampled from a distribution, which satisfy a certain entropy guarantee.

Whether zero knowledge with a truly deterministic prover is possible considering any meaningful form of malicious verifiers remains unknown.

1.1 This Work

We prove that deterministic-prover zero knowledge for non-trivial languages is feasible for the class of malicious verifiers with bounded auxiliary input.

Theorem 1 (Informal)**.** *Assuming non-interactive witness-indistinguishable proofs and subexponentially-secure indistinguishability obfuscation and one-way functions, there exist two-message deterministic-prover arguments for* $\mathsf{NP} \cap \mathsf{coNP}$ *that are zero-knowledge against bounded-auxiliary-input verifiers.*[1]

Theorem 2 (Informal)**.** *Assuming also keyless hash functions that are collision-resistant against bounded-auxiliary-input quasipolynomial-time attackers, there exist similar arguments for all of* NP*.*

By zero knowledge against bounded-auxiliary-input verifiers we formally mean that for any polynomial bound b, there exists a corresponding deterministic-prover argument that is zero knowledge against (malicious) verifiers with non-uniform auxiliary input of size at most b. This, in particular, includes the class of *uniform verifiers*, considered in the original zero-knowledge definition of [GMR89]. We stress that the running time of the verifier may be an arbitrary polynomial, potentially larger than b. Also, indistinguishability of simulated and real proofs holds against non-uniform distinguishers of arbitrary polynomial size. Same goes for soundness, which holds against non-uniform provers of arbitrary polynomial size.

Together with the impossibility result of Goldreich and Oren for unbounded auxiliary input, the above results give a complete picture of when exactly deterministic-prover zero knowledge is feasible. We note

[1] Indistinguishability obfuscation implies non-interactive witness indistinguishable proofs, but with a randomized verifier [BP15], which is insufficient for our purpose. The verifier can be derandomized under a worst-case Nisan-Wigderson [NW94] type derandomization assumption [BV17]. Non-interactive witness indistinguishable proofs with a deterministic verifier are also known from standard assumptions on bilinear maps [GOS06].

that two-message zero knowledge against unbounded auxiliary input is by itself known to be impossible. Our result indeed circumvents this impossibility (for bounded auxiliary input), but this was already known (with a randomized prover) [BCPR14].

On the Necessity of Strong Assumptions and Predictable Arguments. To demonstrate the feasibility of deterministic-prover zero knowledge, we rely on hardness assumptions that are arguably strong. We show that this is inherent. Specifically, we show that deterministic prover zero-knowledge arguments for NP imply witness encryption for NP, which at this point is only known based on strong assumptions, such as indistinguishability obfuscation.

The implication to witness encryption, in fact, follows from a more general implication to *predictable arguments*. Predictable arguments, introduced by Faonio, Nielsen, and Venturi [FNV17], are arguments where the honest verifier's (private) random coins efficiently determine a unique accepting transcript — in order to convince the verifier, the prover must be consistent with this transcript throughout the entire protocol. We prove that any deterministic-prover zero-knowledge argument against bounded-auxiliary-input verifiers can be turned into a predictable argument. The transformation, in fact, preserves the honest prover algorithm, and in particular also zero knowledge.

Theorem 3 (Informal)**.** *Any deterministic-prover zero-knowledge argument against bounded-auxiliary-input verifiers can be made predictable.*

We also give a transformation that only requires honest-verifier zero knowledge and works provided that the argument is expressive enough (e.g., for all NP or even just NP∩coNP). The fact that deterministic-prover zero knowledge arguments imply witness encryption, then follows from [FNV17] where predictable arguments are shown to imply witness encryption.

Corollary 1 (of Predictability)**.** *Any deterministic-prover zero-knowledge argument against bounded-auxiliary-input verifiers for a language $\mathcal{L}$ implies a witness encryption scheme for $\mathcal{L}$.*

We use additional known results regarding predictable arguments [FNV17] to deduce similar results for deterministic-prover zero knowledge:

Corollary 2 (of Predictability)**.** *Any deterministic-prover zero-knowledge argument against bounded-auxiliary-input verifiers can be reduced to two messages and made laconic.*

Here by *laconic* [GVW01, FNV17] we mean that the prover sends a *single* bit and the soundness error is negligibly close to $1/2$; or more generally, the prover sends ℓ bit in order to obtain a soundness error negligibly close to $2^{-\ell}$.

Non-Black-Box Zero-Knowledge Simulation. The zero-knowledge simulator in our constructed arguments makes non-black-box use of the verifier's code. This is known to be inherent — black-box simulation is impossible in the setting of two (or even three) message zero knowledge against bounded-auxiliary-input verifiers [GK96, BCPR14].

1.2 Technical Overview

We now give an overview of the main ideas and techniques behind our results.

The Deterministic-Prover Zero-Knowledge Protocol. Our starting point is the protocol against honest verifiers based on witness encryption [FNV17]. In their protocol, the verifier simply sends a witness encryption of a random message u with respect to the statement $x \in \mathcal{L}$ to be proven, and expects to get u back from the prover. Witness encryption guarantees that a prover that has a corresponding witness w, can obtain u and convince the verifier. However, if the statement is false, namely $x \notin \mathcal{L}$, u is hidden, and soundness is guaranteed.

While honest verifiers are easy to simulate in this scheme, it is not clear how to simulate malicious verifiers. For this purpose, we aim to add to the protocol *a trapdoor way of obtaining* u. A simulator that has the code of the verifier should be able to extract the message u. In contrast, a malicious prover who doesn't have the code (specifically, the verifier's randomness) should still fail to find u when $x \notin \mathcal{L}$.

Explainable Verifiers. To explain the idea behind the protocol in its simplest form, let us start by assuming that the first message v sent by verifier to the prover is always *explainable* [BKP19]. That is, there exist honest verifier coins r that explain this message as an honest verifier message $\mathsf{v} = \mathsf{V}(x; r)$. The difference between this setting and the honest verifier setting is that the explaining coins r may be distributed arbitrarily and also computationally hard to find.

Our basic idea is for the verifier to send the prover yet another witness encryption of u where *the witness is basically the malicious verifier code* V^*. Our realization of this idea is inspired by Barak's uniform simulation technique [Bar01]. Let b be the given bound on the description size of the verifier including its (bounded) auxiliary input hardwired. Then, the honest verifier samples a long random string $R \leftarrow \{0,1\}^{b+2\lambda}$. Then in addition to the witness encryption of u under the statement $x \in \mathcal{L}$, it sends a witness encryption of u under the statement:

"There exists a program Π *of size* $b + \lambda$ *(namely short) that outputs* R*."*

To argue that the protocol remains sound, we note that except with negligible probability $2^{-\lambda}$ over the choice of r, such a short program does not exist. In this case, witness encryption will guarantee that u remains hidden and soundness is preserved. Furthermore, a simulator in possession of the b-size code V^* of the malicious verifier can now use it to simulate. Specifically, let ℓ be the amount of coins r^* used by V^*, then the simulator will sample r^* using a pseudorandom generator that stretches a seed s^* of length $\approx \lambda$ to a pseudorandom r^* of length ℓ. Looking at the string R that $\mathsf{V}^*(x; r^*)$ outputs, the simulator now possesses a size-$(b + \lambda)$ program Π that computes R — the code of V^* with the seed s^* hardwired. This in turn leads to valid simulation.

Witness Encryption for Unbounded NP Relations and IO. One thing to notice about the latter protocol is that in fact the existence of program Π that outputs R is not an NP statement, unless we restrict the running time of Π to some specific polynomial. However, while the non-uniform description size (equivalently, auxiliary input size) of the malicious verifier V^* is a-priori bounded, its running time is not bounded by any specific polynomial.

Accordingly, we need a strong notion of witness encryption for unbounded non-deterministic relations. Specifically, encryption under a statement x should take time polynomial in $|x|$ (and the security parameter), and not depend on the time required to verify a witness for x. In contrast, decrypting with a witness w should take time proportional to the time required to verify w. Such witness encryption schemes directly follow from known indistinguishability obfuscation (IO) schemes for Turing Machines, which are in turn constructed from subexponentially-secure IO for circuits [KLW15, BCG+18, GS18].

Malicious Verifiers. Having constructed a protocol against explainable verifiers, we use compilers from the literature to turn it into a protocol against arbitrary verifiers. These compilers use non-interactive witness-indistinguishable proofs (NIWIs) in order to enforce explainable behavior on the verifier's side. Being non-interactive verifying, these proofs require no randomness from the honest zero-knowledge prover.

The first such compiler [BKP19] works for $\mathsf{NP} \cap \mathsf{coNP}$ and requires no additional hardness assumptions. The second compiler is taken from [BP04] (where it was used in a different context) and relies in addition on keyless hash functions that are collision resistant against attackers with bounded auxiliary input and quasipolynomial running time, as well as subexponentially secure commitments (which in turn follow from subexponentially secure IO and one-way functions). In the body, we reanalyze these compilers to show that they can be used to enforce *robust explainability*, which roughly means that the verifier's messages are almost always explainable on any efficiently samplable distribution on its coins, a property required for our simulation strategy. See more details in Section 3.

From Deterministic-Prover Zero Knowledge to Predictable Arguments. We now explain how deterministic-prover zero knowledge implies predictable arguments, which in turn imply witness encryption (as well as the additional properties stated in Corollary 2). We start with an oversimplified transformation that captures the main idea, but does not fully work, and then explain how to augment it. This oversimplified transformation, in fact, starts from deterministic-prover *honest-verifier* zero knowledge.

Let (P, V) be our argument, and let Sim be the honest-verifier simulator. We consider a new verifier V' that works as follows. It applies the simulator $\mathsf{Sim}(x)$ to obtain simulated randomness $\tilde{r}$ for the honest verifier along with simulated prover messages $\tilde{\mathsf{p}}_1, \ldots, \tilde{\mathsf{p}}_k$. The verifier V' then certifies that the prover messages lead to an accepting transcript with respect to the verifier coins r. If they do not lead to an accepting transcript, V' automatically rejects; otherwise, it interacts with the prover, and rejects the moment it receives a message $\mathsf{p}_i \neq \tilde{\mathsf{p}}_i$. The described protocol is predictable by construction. Also, since we do not change the honest prover, it is zero knowledge against the same class of verifiers as the original protocol. We now turn to argue that the protocol is complete and sound.

To see that the protocol has almost perfect completeness, consider a distinguisher that has the witness w hardwired. Given a transcript $\mathsf{p}_1, \ldots, \mathsf{p}_k$ and verifier coins r, it can perfectly emulate a conversation between the deterministic prover $\mathsf{P}(x, w)$ and honest verifier $\mathsf{V}(x; r)$ and check whether the produced prover messages are consistent with the input transcript $\mathsf{p}_1, \ldots, \mathsf{p}_k$, and that the transcript is accepting. We deduce that with overwhelming probability the simulator produces simulated messages $\tilde{\mathsf{p}}_1, \ldots, \tilde{\mathsf{p}}_k$, and randomness r, such that the honest prover would produce the same messages, and the transcript will be accepting. To see soundness, notice that if the simulated coins r are pseudorandom and the simulated prover messages $\tilde{\mathsf{p}}_1, \ldots, \tilde{\mathsf{p}}_k$ are accepting, then by the soundness of the original protocol (P, V), it should be hard for an efficient prover to produce messages consistent with $\tilde{\mathsf{p}}_1, \ldots, \tilde{\mathsf{p}}_k$ (or with any accepting transcript).

Above, when proving soundness we actually made the implicit assumption that the honest verifier simulator $\mathsf{Sim}(x)$ produces pseudorandom verifier coins, even when given a no instance $x \notin \mathcal{L}$. Indeed, with respect to random, or pseudorandom, coins, we can argue that it is hard to find accepting transcripts. While this is a natural property, it does not follow directly from honest verifier zero knowledge. To circumvent this difficulty, we slightly augment the above transformation, while relying on zero-knowledge against (not necessarily honest) bounded-auxiliary-input verifiers.

Specifically, the verifier V' uses a pseudorandom generator to sample coins r for the honest verifier V, using a short seed s. It then applies the same procedure as above, except that it runs the simulator

$\mathsf{Sim}(\mathsf{V}_s, x)$ for the deterministic verifier V_s that first derives the coins r from the seed s, and then applies V. By choosing an appropriate pseudorandom generator, we can guarantee that the non-uniform description of V_s is short enough. This transformation guarantees that the simulated coins are pseudorandom, even for a no instance, and allows the above proof to go through. The necessity of zero-knowledge to hold even for verifiers that are not necessarily honest comes from the fact that our description of V_s deviates from the honest verifier strategy. We give another construction of predictable arguments from deterministic-prover arguments that are only honest-verifier zero knowledge, provided that the arguments supports expressive enough languages. See Section A for details.

A Word on Two-Message Laconic Arguments. As stated in Corollary 2, we use the implication to predictable arguments to also derive that any deterministic-prover zero knowledge argument for bounded-auxiliary-input verifiers can be made two message and laconic. This corollary is obtained by applying as is general transformations on predictable arguments [FNV17]. The only thing we need to prove is that these transformations preserve zero knowledge. The only hurdle here is that the mentioned transformations involve parallel repetition for the sake of soundness amplification. We observe that (unlike many-round zero knowledge) two-message zero knowledge against bounded-auxiliary-input verifiers is closed under parallel repetition.

On Deterministic Prover Zero-Knowledge *Proofs*. While our results (in conjunction with prior works) provide a complete picture of deterministic zero-knowledge arguments, our results do not have any bearing on deterministic zero-knowledge *proofs*, where soundness is required to hold against unbounded provers. Completing the picture for *proofs* remains an interesting open problem.

2 Definitions

In this work, we will consider PPT machines with both, bounded and unbounded non-uniform auxiliary input. For simplicity of notation, rather than considering explicit auxiliary input in our definitions, we consider two basic notions of non-uniformity. The corresponding zero knowledge definition will in particular capture the auxiliary input setting. See Remark 1.

1. *non-uniform* PPT: this is the standard notion of non-uniform PPT machines. Formally, a non-uniform PPT $\mathsf{M} = \{\mathsf{M}_\lambda\}_\lambda$ is a family of probabilistic Turing machines (one for each λ), where there exists a polynomial poly, such that the description size $|\mathsf{M}_\lambda|$ and the running time of M_λ are bounded by $\mathsf{poly}(\lambda)$.

2. *b-non-uniform* PPT: These are PPT machines with non-uniform description of size $b(\lambda)$ and arbitrary polynomial running time (possibly larger than $b(\lambda)$). Formally, a b-non-uniform PPT $\mathsf{M} = \{\mathsf{M}_\lambda\}_\lambda$ is a family of probabilistic Turing machines (one for each λ), where $|\mathsf{M}_\lambda| \leq b(\lambda)$ and there exists a polynomial poly, such that the running time of M_λ is bounded by $\mathsf{poly}(\lambda)$.

In both of the above, we often omit from M_λ the subscript λ when it is clear from the context. If we simply say a PPT machine, we mean a uniform one.

Throughout this work, we will talk about computational indistinguishability with respect to non-uniform distinguishers.

Definition 1 (Computational Indistinguishability). *Two ensembles $X = \{X_\alpha\}_{\alpha\in S}$ and $Y = \{Y_\alpha\}_{\alpha\in S}$ are said to be computationally indistinguishable, denoted by $X \approx_c Y$, if for every non-uniform* PPT *distinguisher $\mathcal{D}$, every polynomial* p*, all sufficiently large λ and every $\alpha \in \{0,1\}^{\mathsf{poly}(\lambda)} \cap S$*

$$\left| \Pr\left[\mathcal{D}(1^\lambda, X_\alpha) = 1\right] - \Pr\left[\mathcal{D}(1^\lambda, Y_\alpha) = 1\right] \right| < \frac{1}{\mathsf{p}(\lambda)} \ ,$$

where the probability are taken over the samples of X_α, Y_α and coin tosses of $\mathcal{D}$.

We shall sometimes find it convenient to talk about the stronger notion of statistical indistinguishability, defined below.

Definition 2 (Statistical Indistinguishability). *Two ensembles $X = \{X_\alpha\}_{\alpha\in S}$ and $Y = \{Y_\alpha\}_{\alpha\in S}$ are said to be statistically indistinguishable, denoted by $X \approx_s Y$, if for every polynomial* p*, all sufficiently large λ and every $\alpha \in \{0,1\}^{\mathsf{poly}(\lambda)} \cap S$*

$$\Delta(X_\alpha, Y_\alpha) < \frac{1}{\mathsf{p}(\lambda)} \ ,$$

where $\Delta(X_\alpha, Y_\alpha)$ corresponds to the statistical distance between X_α and Y_α.

2.1 Deterministic-Prover Zero Knowledge Against Bounded-Auxiliary-Input Verifiers

We define the notion of deterministic-prover zero-knowledge arguments against verifiers with bounded auxiliary-input (DPZK). We shall denote by $\mathsf{Out}_A\langle A(a), B(b)\rangle$ the output of party A on execution of the protocol between A with input a, and B with input b. By $\mathsf{View}_A\langle A(a), B(b)\rangle$, we denote the view of party A consisting of the protocol transcript along with its random tape.

Definition 3. *An interactive protocol (P, V) between a* ***deterministic*** *polynomial time prover* P *and* PPT *verifier* V*, for a language $\mathcal{L}$ is a deterministic prover b-bounded-auxiliary-input zero knowledge argument if the following holds.*

Completeness: *For every $x \in \mathcal{L}$,*

$$\Pr[\mathsf{Out}_\mathsf{V}\langle \mathsf{P}(x,w), \mathsf{V}(x)\rangle = 1] = 1 \ .$$

Soundness: *For any non-uniform* PPT P^**, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$ and $x \in \{0,1\}^\lambda \setminus \mathcal{L}$,*

$$\Pr[\mathsf{Out}_\mathsf{V}\langle \mathsf{P}^*, \mathsf{V}(x)\rangle = 1] \leq \mathsf{negl}(\lambda) \ .$$

Zero Knowledge: *There exists a* PPT *simulator* Sim*, such that for every b-non-uniform* PPT *verifier* V^* *of running time at most $t(\lambda)$,*

$$\left\{\mathsf{View}_{\mathsf{V}^*}\langle \mathsf{P}(x,w), \mathsf{V}^*\rangle\right\}_{\substack{\lambda\in\mathbb{N},\\ x\in\mathcal{L}\cap\{0,1\}^\lambda,\\ w\in R_\mathcal{L}(x)}} \approx_c \left\{\mathsf{Sim}(\mathsf{V}^*, 1^t, x)\right\}_{\substack{\lambda\in\mathbb{N},\\ x\in\mathcal{L}\cap\{0,1\}^\lambda,\\ w\in R_\mathcal{L}(x)}} \ .$$

Remark 1 (Universal Simulation). *In the above definition, there exists one universal simulator* Sim *that gets the code of the verifier as input. We note that this definition is known [GO94] to imply the alternative definition of (bounded) auxiliary-input zero knowledge that requires that any for any t-time* V^* *there is a* PPT *simulator* $\mathsf{Sim}_{\mathsf{V}^*}$ *such that given (bounded) auxiliary input z,* $\mathsf{Sim}_{\mathsf{V}^*}(x, z, 1^t)$ *simulates* $\mathsf{V}^*(z)$*.*

2.2 Indistinguishability Obfuscation (IO)

We now give a definition of indistinguishability obfuscator for Turing Machines, which can be constructed from indistinguishability obfuscators for circuits [KLW15, BCG+18, GS18].

Definition 4 (Indistinguishability Obfuscator for Turing Machines)**.** *A succinct indistinguishability obfuscator for Turing machines consists of a PPT machine* iOM *that works as follows:*

- iOM *takes as input the security parameter* 1^λ*, the Turing machine* M *to obfuscate, an input length* n*, and time bound* t*.*
- iOM *outputs a Turing machine* $\widetilde{\mathsf{M}}$ *which is an obfuscation of* M *corresponding to input length* n *and time bound* t*.* $\widetilde{\mathsf{M}}$ *takes as input* $x \in \{0,1\}^n$*.*

The scheme should satisfy the following requirements:

Correctness *For all* $\lambda \in \mathbb{N}$*, for all* $\mathsf{M} \in \mathcal{M}_\lambda$*, for all inputs* $x \in \{0,1\}^n$*, time bounds* t' *such that* $t' \leq t$*, let* y *be the output of* $\mathsf{M}(x)$ *after at most* t *steps, then*

$$\Pr\left[\widetilde{\mathsf{M}} \leftarrow \mathsf{iOM}(1^\lambda, 1^n, 1^{\log t}, \mathsf{M}) \ : \ \widetilde{\mathsf{M}}(x) = y\right] = 1 \ .$$

Security *It holds that*

$$\left\{\mathsf{iOM}(1^\lambda, 1^n, 1^{\log t}, \mathsf{M}_0)\right\}_{\substack{\lambda,t,n,\\ \mathsf{M}_0,\mathsf{M}_1}} \approx_c \left\{\mathsf{iOM}(1^\lambda, 1^n, 1^{\log t}, \mathsf{M}_1)\right\}_{\substack{\lambda,t,n,\\ \mathsf{M}_0,\mathsf{M}_1}} \ ,$$

where $\lambda \in \mathbb{N}$*,* $n \leq t \leq 2^\lambda$*, and* $\mathsf{M}_0, \mathsf{M}_1$ *are any pair of machines of the same size such that for any input* $x \in \{0,1\}^n$ *both halt after the same number of steps with the same output.*

Efficiency and Succinctness *We require that the running time of* iOM *and the length of its output, namely the obfuscated machine* $\widetilde{\mathsf{M}}$*, is* $\mathsf{poly}(|\mathsf{M}|, \log t, n, \lambda)$*. We also require that the running time* $\tilde{t}_x$ *of* $\widetilde{\mathsf{M}}(x)$ *is* $\mathsf{poly}(t_x, |\mathsf{M}|, n, \lambda)$*, where* t_x *is the running time of* $\mathsf{M}(x)$*.*

2.3 Witness Encryption

The following definition of witness encryption is taken from [GGSW13].

Definition 5. *A witness encryption scheme for an* NP *language* $\mathcal{L}$*, with corresponding witness relation* $R_\mathcal{L}$*, consists of the following two polynomial-time algorithms:*

Encryption. *The probabilistic algorithm* $\mathsf{WE.Enc}(1^\lambda, x, m)$ *takes as input a security parameter* 1^λ*, a string* $x \in \{0,1\}^*$*, and a message* $m \in \{0,1\}$*. It outputs a ciphertext* ct*.*

Decryption. *The algorithm* $\mathsf{WE.Dec}(\mathsf{ct}, w)$ *takes as input a ciphertext* ct*, a string* $w \in \{0,1\}^*$*. It outputs either a message* $m \in \{0,1\}$*.*

The above algorithms satisfy the following conditions:

- ***Correctness.*** *For any security parameter* λ*, for any* $m \in \{0,1\}$*, and for any* $(x, w) \in R_\mathcal{L}$*, we have that*

$$\Pr\left[\mathsf{ct} \leftarrow \mathsf{WE.Enc}(1^\lambda, x, m) \ : \ \mathsf{WE.Dec}(\mathsf{ct}, w) = m\right] = 1 \ .$$

– ***Security.*** *For any non-uniform* PPT *adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for any* $\lambda \in \mathbb{N}$*, and any* $x \notin \mathcal{L}$*, we have that*

$$\left\{\mathsf{WE.Enc}(1^\lambda, x, 0)\right\}_{\lambda\in\mathbb{N}, x\notin\mathcal{L}} \approx_c \left\{\mathsf{WE.Enc}(1^\lambda, x, 1)\right\}_{\lambda\in\mathbb{N}, x\notin\mathcal{L}} \quad .$$

We note that the above scheme can be extended to encrypt strings, rather than just bits, by encrypting each bit independently. Witness encryption for all of NP can be constructed from IO for circuits [GGSW13].

2.4 Non-interactive Witness Indistinguishability (NIWI)

Definition 6 ([BOV03])**.** *A non-interactive witness-indistinguishable proof system* $\mathsf{NIWI} = (\mathsf{NIWI.Prov}, \mathsf{NIWI.Ver})$ *for an* NP *relation* $R_\mathcal{L}$ *consists of two polynomial-time algorithms:*

– *a probabilistic prover* $\mathsf{NIWI.Prov}(x, w, 1^\lambda)$ *that given an instance* x*, witness* w*, and security parameter* 1^λ*, produces a proof* π*.*

– *a deterministic verifier* $\mathsf{NIWI.Ver}(x, \pi)$ *that verifies the proof.*

We make the following requirements:

Completeness *for every* $\lambda \in \mathbb{N}, (x, w) \in R_\mathcal{L}$*,*

$$\Pr\left[\pi \leftarrow \mathsf{NIWI.Prov}(x, w, 1^\lambda) \ : \ \mathsf{NIWI.Ver}(x, \pi) = 1\right] = 1$$

Soundness *for every* $x \notin \mathcal{L}$ *and* $\pi \in \{0,1\}^*$*,*

$$\mathsf{NIWI.Ver}(x, \pi) = 0 \quad .$$

Witness Indistinguishability *It holds that*

$$\left\{\mathsf{NIWI.Prov}(x, w_0, 1^\lambda)\right\}_{\substack{\lambda, x,\\ w_0, w_1}} \approx_c \left\{\mathsf{NIWI.Prov}(x, w_1, 1^\lambda)\right\}_{\substack{\lambda, x,\\ w_0, w_1}} \quad ,$$

where $\lambda \in \mathbb{N}, x \in \{0,1\}^\lambda, w_0, w_1 \in R_\mathcal{L}(x)$*.*

2.5 Collision Resistance against Bounded Non-uniform Adversaries

We describe here the notion of keyless collision resistance against quasi-polynomial b-non-uniform adversaries, extending the definition in [BP04].

Syntax. A keyless collision resistance hash function is associated with an input function $\ell(\lambda) > \lambda$ and a polynomial time algorithm H such that $\mathsf{H}(1^\lambda, X)$ is a deterministic algorithm that takes as input an $X \in \{0,1\}^{\ell(\lambda)}$ and outputs a hash $Y \in \{0,1\}^\lambda$.

Definition 7. *We say that* H *is collision-resistant against quasi-polynomial adversaries if for any* b*-non-uniform probabilistic* $2^{\mathsf{poly}(\log\lambda)}$*-time* $\mathcal{A}$*, there exists a negligible function* negl*, such that for any* $\lambda \in \mathbb{N}$*,*

$$\Pr\left[(x_1, x_2) \leftarrow \mathcal{A}(1^\lambda) \ : \ x_1 \neq x_2, \mathsf{H}(1^\lambda, x_1) = \mathsf{H}(1^\lambda, x_2)\right] \leq \mathsf{negl}(\lambda) \quad .$$

2.6 Non-interactive Commitment Schemes

We define below bit commitment schemes

Definition 8 (Non-interactive Bit Commitment Schemes)**.** *A polynomial time computable function:* $\mathsf{Com} : \{0,1\} \times \{0,1\}^\lambda \mapsto \{0,1\}^{\ell(\lambda)}$ *is a bit commitment if it satisfies the properties below:*

Binding: *For any* $r, r' \in \{0,1\}^\lambda, b, b' \in \{0,1\}$*, if* $\mathsf{Com}(b;r) = \mathsf{Com}(b';r')$ *then* $b = b'$*.*

Computational Hiding: *The following holds:*

$$\left\{\mathsf{Com}(0) : r \leftarrow\!\!\$\, \{0,1\}^\lambda\right\} \approx_c \left\{\mathsf{Com}(1;r) : r \leftarrow\!\!\$\, \{0,1\}^\lambda\right\} \ .$$

where computational indistinguishability is with respect to arbitrary non-uniform PPT *distinguisher.*

We note that the above scheme can be extended to commit to strings, rather than just bits, by committing to each bit independently. Looking ahead, we require that the underlying string that is committed can be extracted in quasi-polynomial time. Such commitments can be constructed from subexponentiall-secure injective one-way functions (which in turn can be constructed from subexponential IO and one-way functions).

2.7 Explainable Verifiers

We define here the a variant of the notion of explainable verifiers [BKP19] called robustly-explainable verifiers. Roughly speaking, explainable verifiers are ones whose messages almost always lie in the support of the honest verifier messages (or are abort). Robustly-explainable verifiers are such where this occurs when they use random coins sampled from an arbitrary efficient sampler (and not necessarily the uniform distribution).

Definition 9 (Explainable Message)**.** *Let* $\langle \mathsf{P}, \mathsf{V} \rangle$ *be a two-message protocol. We say that a given message* m *is explainable with respect to* x*, if there exist honest verifier coins* r *such that* $m \in \{V(x;r), \perp\}$*.*

Definition 10 (Robustly-Explainable Verifier)**.** *Let* $\langle \mathsf{P}, \mathsf{V} \rangle$ *be a protocol. A* b*-non-uniform* PPT *verifier* V^* *using* $\ell(\lambda)$ *random coins is robustly-explainable if for any* PPT *sampler* R *on* $\ell(\lambda)$ *bits, there exists a negligible* $\mathsf{negl}(\lambda)$ *such that for any* $\lambda \in \mathbb{N}$ *and* $x \in \lambda$*,*

$$\Pr\left[r \leftarrow R(1^\lambda), m = \mathsf{V}^*(x;r) : m \text{ is explainable}\right] \geq 1 - \mathsf{negl}(\lambda) \ .$$

2.8 Pseudorandom Generators

Definition 11 (Psedudorandom Generators)**.** *A deterministic function* $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^{p(\lambda)}$ *is called a pseudorandom generator (PRG) if:*

1. *(efficiency):* PRG *can be computed in polynomial time,*
2. *(expansion):* $p(\lambda) > \lambda$*,*
3. $\left\{x \leftarrow \{0,1\}^\lambda : \mathsf{PRG}(x)\right\} \approx_c \left\{U_{p(\lambda)}\right\}$*, where* $U_{p(\lambda)}$ *is the uniform distribution over* $p(\lambda)$ *bits.*

3 A Deterministic-Prover Zero-Knowledge Protocol

In this section we present our deterministic prover zero knowledge (DPZK) protocol. As explained in the introduction, we start by describing the protocol for robustly-explainable verifiers. We then show how to compile this protocol to one that is secure against malicious verifiers.

3.1 DPZK for Robustly-Explainable Verifiers

We use the following components for the deterministic prover zero knowledge (DPZK) protocol for an NP language $\mathcal{L}$ against b-non-uniform explainable verifiers.

- A witness encryption scheme $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ for language $\mathcal{L}$.
- An indistinguishability obfuscation (IO) scheme iOM for Turing Machines (TM).

Additionally, we will use the machine described below that outputs the hardcoded secret u given as input the description of a "short" Turing machine that outputs a hardcoded public value R.

Machine: Prog

Hardcoded: R, u
Input: $\mathsf{M} \in \{0,1\}^{\rho(\lambda)}$

if M outputs R
    output u
else
    output $\perp$

In what follows, let $\rho(\lambda) = b(\lambda) + \lambda + \omega(1)$, $\ell(\lambda) = \rho(\lambda) + \lambda$. The protocol is described in Figure 1. We prove the properties of the protocol below.

Completeness. Completeness follows from the correctness of witness encryption.

Soundness. We now prove that the above protocol is sound against computationally bounded provers.

Proposition 1. *Assuming security of the indistinguishability obfuscation scheme and the witness encryption scheme, the protocol is sound.*

Proof. We consider a sequence of hybrids transitioning from the real protocol to an ideal protocol where the probability that the prover convinces the verifier of accepting is clearly negligible.

Hyb_0**:** This is the real protocol.

Hyb_1**:** In this hybrid, we modify the program Prog to Prog' that always output $\perp$.

By our choice of parameters and a union bound, the probability that there exists a machine $\mathsf{M} \in \{0,1\}^{\rho}$ that outputs R is at most $2^{\rho-\ell} = 2^{-\lambda}$. Therefore, except with negligible probability Prog and Prog' are functionally equivalent. The indistinguishability of Hyb_1 and Hyb_0 then follows from the indistinguishability of the IO scheme.

Protocol: DPZK for robustly-explainable verifiers

Common input: Input $x \in \mathcal{L}$, security parameter 1^λ

P**'s auxiliary input:** witness w such that $(x, w) \in R_{\mathcal{L}}$

1. Verifier V computes the first message as
   (a) $R \leftarrow_\$ \{0,1\}^{\ell(\lambda)}$
   (b) $t := \lambda^{\log \lambda}$
   (c) $u \leftarrow_\$ \{0,1\}^\lambda$
   (d) $\widetilde{\mathsf{Prog}} \leftarrow \mathsf{iOM}\left(1^\lambda, 1^\rho, 1^{\log t}, \mathsf{Prog}\left[R, u\right]\right)$
   (e) $\mathsf{ct} \leftarrow \mathsf{WE.Enc}(u, x)$

   send $(R, \mathsf{ct}, \widetilde{\mathsf{Prog}})$ to the prover P.
2. Prover P computes the second message as
   (a) $\widetilde{u} := \mathsf{WE.Dec}(\mathsf{ct}, x, w)$

   send $\widetilde{u}$ to the verifier V.
3. Verifier V performs the check
   (a) if $\widetilde{u} = u$, accept. Else, reject.

Figure 1: Deterministic prover zero-knowledge for robustly-explainable verifiers.

Hyb_2**:** In this hybrid, we additionally change the ciphertext ct of the witness encryption scheme to be the encryption of 0.

Since $x \notin \mathcal{L}$, the indistinguishability between Hyb_2 and Hyb_1 follows from the security of the witness encryption scheme.

It is left to observe that in Hyb_2 the prover obtains no information about u, and thus convinces the verifier with probability at most $2^{-\lambda}$. □

Zero Knowledge. We prove

Proposition 2. *Assuming the existence of pseudorandom generators, the protocol is zero knowledge against b-non-uniform verifiers.*

Proof. We describe the simulation strategy below. In what follows V^* is a b-non-uniform malicious verifier of polynomial running time at most $t(\lambda)$. Additionally, let k be the amount of random coins r^* used by V^*. The simulator Sim will use a PRG $\mathsf{PRG} : \{0,1\}^\lambda \mapsto \{0,1\}^k$.

$\underline{\mathsf{Sim}(\mathsf{V}^*, 1^t, x)}$:

1. Construct verifier V^*_s that has the seed s hardwired. V^*_s computes $\mathsf{PRG}(s)$ and uses it as random coins for V^*. Additionally, V^*_s truncates V^*'s output to R.
2. Initialize V^* with random coins $\mathsf{PRG}(s)$.
3. Given $\widetilde{\mathsf{Prog}}$ from V^*, use the description of V^*_s as input to $\widetilde{\mathsf{Prog}}$ and obtain u.
4. u is then used as the simulated prover message, along with verifier randomness $\mathsf{PRG}(s)$.

First, consider an execution between the prover and augmented verifier $\langle \mathsf{P}(x,w), \mathsf{V}^*_s \rangle$, and let v and p denote the verifier and prover messages in such an execution. Then by pseudorandomness of PRG,

$$\mathsf{View}_{\mathsf{V}^*} \langle \mathsf{P}(x,w), \mathsf{V}^* \rangle \approx_c \mathsf{p}, \mathsf{PRG}(s) \ .$$

Next, by the fact that V^* is robustly explainable, we know that except with negligible probability, $\mathsf{v} = (R, \mathsf{ct}, \widetilde{\mathsf{Prog}})$ is explainable; namely, has the structure prescribed by the honest verifier algorithm. Noting that V^*_s is a program of length $b + \lambda + O(1) < \rho(\lambda)$ and running time at most $t(\lambda)$ that outputs R. By the fact that v is explainable, $\widetilde{\mathsf{Prog}}(\mathsf{V}^*_s) = \mathsf{WE.Dec}(\mathsf{ct}, x, w)$. It follows that

$$\mathsf{p}, \mathsf{PRG}(s) \approx_s \mathsf{Sim}(\mathsf{V}^*, 1^t, x) \ ,$$

and overall

$$\mathsf{View}_{\mathsf{V}^*} \langle \mathsf{P}(x,w), \mathsf{V}^* \rangle \approx_c \mathsf{Sim}(\mathsf{V}^*, 1^t, x) \ ,$$

as required.

□

3.2 From Explainable to Malicious Verifiers

In this section we give generic compilers going from robust-explainable to malicious verifiers. These compilers were constructed in [BKP19] where they were used to enforce explainability and in [BP04] where they were used in a different context. We prove that these compilers, in fact, enforce robust explainability. The statements, and correspondingly the underlying assumptions, change based on whether we want a DPZK for $\mathsf{NP} \cap \mathsf{coNP}$, or for all of NP. We discuss the two cases separately.

3.2.1 DPZK for NP ∩ coNP

We consider languages $\mathcal{L} \in \mathsf{NP} \cap \mathsf{coNP}$, which in turn means that in addition to relation $R_{\mathcal{L}}$, there is also a NP-relation $R_{\overline{\mathcal{L}}}$ to certify that a statement $x \notin \mathcal{L}$.
We use the following primitives in our construction:

- A two-message deterministic-prover zero-knowledge (DPZK) protocol $(e\mathsf{P}, e\mathsf{V})$ secure against robustly-explainable verifiers. Let the verifier and prover messages be denoted by v and p, respectively.

– A non-interactive witness indistinguishable proof (NIWI) $(\mathsf{NIWI.Prov}, \mathsf{NIWI.Ver})$ for the language

$$\mathcal{L}_{\mathsf{NIWI}} = \left\{ (\mathsf{v}, x) \;\middle|\; \exists (r, \bar{w}) \text{ s.t. } \mathsf{v} = e\mathsf{V}(x; r) \text{ OR } R_{\overline{\mathcal{L}}}(x, \bar{w}) = 1 \right\} \ ,$$

namely, either the verifier's message is explainable, or the statement is not in the language. Henceforth, we shall refer to the second half of the 'OR' statement, that the statement is not in the language, to be the *trapdoor statement*.

The protocol is presented in Figure 2.

Protocol: (P,V) for $\mathcal{L} \in \mathsf{NP} \cap \mathsf{coNP}$

Common input: Input $x \in \mathcal{L}$, security parameter 1^λ

P**'s auxiliary input:** witness w such that $(x, w) \in R_{\mathcal{L}}$

1. Verifier V computes the first message as
   (a) $r \leftarrow_\$ \{0,1\}^{p(n)}$
   (b) $\mathsf{v} := e\mathsf{V}(x; r)$
   (c) $x_{\mathsf{NIWI}} := (\mathsf{v}, x)$
   (d) $w_{\mathsf{NIWI}} := (r, \perp)$
   (e) $\mathsf{wi} \leftarrow \mathsf{NIWI.Prov}(x_{\mathsf{NIWI}}, w_{\mathsf{NIWI}})$

   send $(\mathsf{v}, \mathsf{wi})$ to the prover P.
2. Prover P computes the second message as
   (a) $\widetilde{x}_{\mathsf{NIWI}} := (\mathsf{v}, x)$
   (b) if $\mathsf{NIWI.Ver}(\widetilde{x}_{\mathsf{NIWI}}, \mathsf{wi}) \neq 1$, output $\perp$.
   (c) $\mathsf{p} := e\mathsf{P}(x, w, \mathsf{v})$.

   send p to the verifier V.
3. Verifier V performs the check
   (a) if $e\mathsf{V}(x, \mathsf{p}; r) = 1$, accept. Else, reject.

Figure 2: Deterministic-prover zero knowledge for $\mathcal{L} \in \mathsf{NP} \cap \mathsf{coNP}$.

Completeness. Completeness follows directly from the completeness of the underlying protocol and the NIWI proof.

Zero Knowledge. We show how any b-non-uniform malicious verifier V^* for the above protocol can be converted to a robustly-explainable $b + O(1)$-non-uniform verifier against the original protocol.

Claim 1. *There exist an efficient simulator* S *and a verifier* $e\mathsf{V}^*$ *such that*

1. $e\mathsf{V}^*$ *is a robustly explainable verifier against* $\langle e\mathsf{P}, e\mathsf{V}\rangle$.
2. $e\mathsf{V}^*$ *is* $(b + O(1))$*-non-uniform and efficiently constructable from* $e\mathsf{V}^*$.
3. *For every* $x \in \mathcal{L}$,
$$\mathsf{View}_{\mathsf{V}^*}\langle \mathsf{P}(x, w), \mathsf{V}^*\rangle \equiv \mathsf{S}(\mathsf{View}_{e\mathsf{V}^*}\langle e\mathsf{P}(x, w), e\mathsf{V}^*\rangle) \ .$$

Proof. We construct $\mathsf{S}, e\mathsf{V}^*$.

$\underline{e\mathsf{V}^*:}$

1. Emulates V^* and obtains $(\mathsf{v}, \mathsf{wi})$.
2. If wi is not a valid proof for the statement (v, x), send $e\mathsf{P}$ the message $\perp$.
3. Else, send $e\mathsf{P}$ v, and get p.
4. Complete emulation of V^* with message p.

$\underline{\mathsf{S}:}$

1. Outputs the randomness of the emulated V^* (can be derived from the randomness of $e\mathsf{V}^*$,
2. as well as the received prover message p (possibly $\perp$).

The third property asserted in the claim follows by construction of $\mathsf{S}, e\mathsf{V}^*$ and the fact that the prover P checks on its own whether the verifier's proof is accepting. It is left to see that $e\mathsf{V}^*$ is robustly explainable, $(b + O(1))$-non-uniform, and efficiently constructable from V^*. Robust explainability follows directly by the (unconditional) soundness of the NIWI — $e\mathsf{V}^*$ either outputs an explainable message or $\perp$. $(b+O(1))$-non-uniformity and efficient construction follow from the fact that V^* is b-non-uniform and $e\mathsf{V}^*$ uses it as a black box and described by the four code lines above.

□

Claim 1 directly gives rise to a zero knowledge Sim for the protocol (P, V). In what follows, let $e\mathsf{Sim}$ be the simulator of the underlying DPZK protocol against robustly-explainable verifiers.

$\underline{\mathsf{Sim}(\mathsf{V}^*, 1^t, x):}$

1. Construct the explainable verifier $e\mathsf{V}^*$.
2. Output $\mathsf{S}(e\mathsf{Sim}(e\mathsf{V}^*, 1^t, x)$.

The validity of the simulator Sim follows directly from that of $e\mathsf{Sim}$ and Claim 1.

Soundness. For soundness, we show that any cheating prover P^* breaking the soundness of the above protocol, can be converted into a prover $e\mathsf{P}^*$ that breaks the soundness of the underlying protocol. $e\mathsf{P}^*$ will have the witness $\bar{w}$ for $x \notin \mathcal{L}$ hardwired.

$e\mathsf{P}^*$:

1. Obtain message v from $e\mathsf{V}$.
2. Use $\bar{w}$ as the witness to compute the NIWI proof wi.
3. Emulate P^* with $(\mathsf{v}, \mathsf{wi})$ and obtain p.
4. Send p to the verifier $e\mathsf{V}$.

First note that since $\mathcal{L} \in \mathsf{NP} \cap \mathsf{coNP}$, the statement $x \notin \mathcal{L}$ has a witness $\bar{w}$ as required. The only difference in the views of P^* and its emulated version in $e\mathsf{P}^*$ is in the NIWI proof. From the witness indistinguishability of the NIWI, P^*'s success probability does not change by more than a negligible amount.

3.2.2 DPZK for all of NP

As mentioned to in the introduction, for the case of NP, we require stronger primitives. Specifically, we use the following primitives for our construction:

- A two round deterministic prover zero knowledge (DPZK) protocol $(e\mathsf{P}, e\mathsf{V})$ secure against robustly-explainable verifiers. Let the verifier and prover messages be denoted by v and p, respectively.
- A non-interactive commitment scheme Com with perfect binding and computational hiding. Additionally, as mentioned earlier, we require that the plaintext underlying a commitment can be extracted in quasi-polynomial time. Such commitments can be constructed from subexponentiall-secure injective one-way functions (which in turn can be constructed from subexponential IO and one-way functions).
- A keyless collision-resistant hash function H secure against $(b+O(1))$-non-uniform quasi-polynomial time adversaries.
- A non-interactive witness-indistinguishable proof (NIWI) $(\mathsf{NIWI.Prov}, \mathsf{NIWI.Ver})$ for the language

$$\mathcal{L}_{\mathsf{NIWI}} = \Big\{ (\mathsf{v}, x, c) \;\Big|\; \exists (r, r_{\mathsf{Com}}, x_1, x_2) \text{ s.t. } \mathsf{v} = e\mathsf{V}(x; r) \text{ OR } \big(c = \mathsf{Com}((x_1, x_2); r_{\mathsf{Com}}) \wedge x_1 \neq x_2 \wedge \mathsf{H}(1^\lambda, x_1) = \mathsf{H}(1^\lambda, x_2)\big) \Big\} \ ,$$

  namely, either the verifier's message is explainable, or the commitment sent by the verifier contains a collision in H. As before, we shall refer to the second half of the 'OR' statement as the *trapdoor statement*.

The protocol is presented in Figure 3.

Completeness. Follows directly from the completeness of the underlying protocol and the NIWI.

Protocol: (P,V) for $\mathcal{L} \in \mathsf{NP}$

Common input: Input $x \in \mathcal{L}$, security parameter 1^λ

P's auxiliary input: witness w such that $(x, w) \in R_{\mathcal{L}}$

1. Verifier V compute the first message as

   (a) $r \leftarrow_\$ \{0,1\}^{p(n)}$
   (b) $c \coloneqq \mathsf{Com}(0; r_{\mathsf{Com}})$
   (c) $\mathsf{v} \coloneqq e\mathsf{V}(x; r)$
   (d) $x_{\mathsf{NIWI}} \coloneqq (x, \mathsf{v}, c, \mathsf{H})$
   (e) $w_{\mathsf{NIWI}} \coloneqq (r, \bot, \bot)$
   (f) $\mathsf{wi} \leftarrow \mathsf{NIWI.Prov}(x_{\mathsf{NIWI}}, w_{\mathsf{NIWI}})$.

   send $(\mathsf{v}, \mathsf{wi}, c)$ to the prover P.

2. Prover P computes the second message as

   (a) $\widetilde{x}_{\mathsf{NIWI}} \coloneqq (x, \mathsf{v}, c, \mathsf{H})$
   (b) if $\mathsf{NIWI.Ver}(\widetilde{x}_{\mathsf{NIWI}}, \mathsf{wi}) \neq 1$, output $\bot$.
   (c) $\mathsf{p} \coloneqq e\mathsf{P}(x, w, \mathsf{v})$.

   send p to the verifier V.

3. Verifier V performs the check

   (a) if $e\mathsf{V}(x, \mathsf{p}; r) = 1$, accept. Else, reject.

Figure 3: Deterministic prover zero-knowledge for $\mathcal{L} \in \mathsf{NP}$.

Zero Knowledge. For zero knowledge, we follow the same strategy as in the previous subsection and show how any b-non-uniform verifier V^* for the above protocol can be converted into a robustly-explainable $(b + O(1))$-non-uniform verifier against the original protocol.

We argue that Claim 1 also holds for this protocol with the exact same S and $e\mathsf{V}^*$. The only difference is in the proof of robust explainability of the verifier $e\mathsf{V}^*$, which is based on complexity leveraging.

Robust Explainability of $e\mathsf{V}^*$. Fix some PPT sampler R for coins for $e\mathsf{V}^*$ and assume toward contradiction that with noticeable probability it outputs a message v that is not explainable when initialized with random

coins sampled using R. We show that there exists a $(b+O(1))$-non-uniform quasi-polynomial time attacker that finds a collision in H. Recall the $e\mathsf{V}^*$ only outputs a non-$\perp$ message provided that the emulated V^* produces a valid NIWI. By the unconditional soundness of the NIWI, it follows that whenever $e\mathsf{V}^*$ outputs a non-explainable message, it must be that c is a valid commitment to a collision in H. This collision is then be extracted from the commitment in quasi-polynomial time. Note that the corresponding collision finder can be described by $e\mathsf{V}^*$ and R, which have non-uniform description of size $b + O(1)$. □

Zero knowledge of (P, V) now follows from that of $(e\mathsf{P}, e\mathsf{V})$ and the existence of S and $e\mathsf{V}^*$, exactly as in the previous subsection.

Soundness. We show that any cheating prover P^* breaking the soundness of the above protocol, can be converted into a prover $e\mathsf{P}^*$ that breaks the soundness of the underlying robustly-explainable protocol. The reduction is similar to that in the previous subsection with some required changed. $e\mathsf{P}^*$ will have a collision (x_1, x_2) as (part of the) witness for the *trapdoor statement* hardwired in its code.

$\underline{e\mathsf{P}^*:}$

1. Obtain message v from $e\mathsf{V}$.
2. Compute $c = \mathsf{Com}(x_1, x_2; r_{\mathsf{Com}})$.
3. Use $(x_1, x_2, r_{\mathsf{Com}})$ as the witness to compute the NIWI proof wi.
4. Emulate P^* with $(\mathsf{v}, \mathsf{wi})$ and obtain p.
5. Send p to the verifier $e\mathsf{V}$.

The difference in the views of P^* and its emulated version in $e\mathsf{P}^*$ is the commitment to (x_1, x_2) rather than zero, and in the witness used for the NIWI proof. Using the hiding of the commitment (against non-uniform PPT attackers) and the witness indistinguishability of the NIWI, P^*'s success probability does not change by more than a negligible amount.

Remark 2. *We emphasize that for soundness, we require that all the underlying primitives to are secure against non-uniform adversaries since our soundness reduction is non-uniform.*

4 Predictable Arguments and DPZK

In this section, we show that any deterministic-prover zero-knowledge (DPZK) argument against bounded-non-uniform verifier can be made *predictable*. The notion of predictable arguments was introduced in [FNV17], where it is in particular shown to imply witness encryption. In the next section, we address additional properties of DPZK that follow from this connection.

We start by recalling the definition of predictable arguments (PA) [FNV17]. While they also address predictable argument of knowledge, we restrict attention to predictable arguments that are only sound.

Definition 12 (Predictable Argument)**.** *A ρ-round predictable argument is an argument specified by a tuple of algorithms $(\mathsf{Chal}, \mathsf{Resp})$ as described below:*

1. *The verifier $\mathsf{PA.V}$ samples $\left(\vec{c}, \vec{b}\right) \leftarrow \mathsf{Chal}(1^\lambda, x)$, where $\vec{c} := (c_1, \cdots, c_\rho)$ and $\vec{b} := (b_1, \cdots, b_\rho)$.*

2. *For all $i \in [\rho]$ in increasing sequence:*

   (a) $\mathsf{PA.V}$ *sends* c_i *to the* $\mathsf{PA.P}$*;*

   (b) *The prover* $\mathsf{PA.P}$ *computes* $a_i := \mathsf{Resp}(1^\lambda, x, w, c_1, \cdots, c_i)$ *and sends* a_i *to* $\mathsf{PA.V}$*.*

   (c) $\mathsf{PA.V}$ *checks if* $a_i = b_i$*, and returns 0 otherwise.*

3. *If all challenges are answered correctly,* $\mathsf{PA.V}$ *returns 1.*

The protocol is required to satisfy:

Correctness. *There exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $x \in \mathcal{L}$ *such that* $R_L(x, w) = 1$*, we have*

$$\Pr[\mathsf{Out}_{\mathsf{PA.V}}\langle \mathsf{PA.P}(x, w), \mathsf{PA.V}(x)\rangle = 1] \geq 1 - \mathsf{negl}(\lambda) \quad .$$

Soundness. *For any non-uniform* PPT *prover* P^**, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $x \notin \mathcal{L}$*,*

$$\Pr[\langle \mathsf{PA.P}^*, \mathsf{PA.V}(x)\rangle = 1] \leq \mathsf{negl}(\lambda) \quad .$$

A **deterministic-prover zero-knowledge predictable argument (PA-DPZK)** is a deterministic-prover zero-knowledge argument that is also a predictable argument.

We prove the following:

Theorem 4. *Let* (P, V) *be a deterministic-prover zero-knowledge argument for* $\mathcal{L}$ *against bounded-non-uniform verifiers. There exists a verifier* V' *such that* $(\mathsf{P}, \mathsf{V}')$ *is a predictable argument.*

Note that since we do not change the honest prover P it follows that $(\mathsf{P}, \mathsf{V}')$ is also deterministic-prover zero knowledge against the same class of verifiers.

Relying on the following result by Faonio, Nielsen, and Venturi,

Theorem 5 ([FNV17])**.** *If there exists a Predictable Argument (PA) for a language* $\mathcal{L}$*, then there exists a witness encryption scheme for* $\mathcal{L}$*.*

our theorem holds for all $\lambda^{\Omega(1)}$-non-uniform verifiers, and we deduce

Corollary 3. *If there exists a deterministic-prover zero-knowledge argument for* $\mathcal{L}$ *against* $\lambda^{\Omega(1)}$*-non-uniform verifiers, then there exists a witness encryption scheme for* $\mathcal{L}$*.*

We now proceed with the proof.

Proof of Theorem 4. Let (P, V) be a ρ-round DPZK argument for $\mathcal{L}$ against b-non-uniform verifiers, for $b(\lambda) \geq 2\lambda + \omega(1)$. Let $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^\ell$ be a pseudorandom generator, where $\ell(\lambda)$ is the amount of coins used by V. For a given seed $s \in \{0,1\}^\lambda$, we define the deterministic verifier $\mathsf{V}_s(x)$ that derives coins $r = \mathsf{PRG}(s)$ for V then emulates $\mathsf{V}(x; r)$.

The transformed verifier V' is presented in Figure 4.

First, note that the protocol satisfies the structural requirement of a predictable argument. We now move to prove completeness and soundness with respect to the new verifier V'.

The New Verifier V'

Input: x, security parameter 1^λ

1. Sample $s \leftarrow\!\!\$\ \{0,1\}^\lambda$ and construct V_s.
2. Sample $\{\widetilde{\mathsf{p}}_i\}_{i=1}^{\rho} \leftarrow \mathsf{Sim}(\mathsf{V}_s, 1^t, x)$, where t is the running time of V_s.
3. Emulate an execution of $\mathsf{V}_s(x)$ with prover messages $\{\widetilde{\mathsf{p}}_i\}_{i=1}^{\rho}$; let $\{\widetilde{\mathsf{v}}_i\}_{i=1}^{\rho}$ be the resulting verifier messages.
4. If the verifier V_s rejects in the above execution, reject.
5. Proceed interacting with the prover P: at each round $i \in [\rho]$:
   - send $\mathsf{v}_i (= \widetilde{\mathsf{v}}_i)$ to P,
   - if the prover answers with $\mathsf{p}_i = \widetilde{\mathsf{p}}_i$, proceed to the next round,
   - else, reject.
6. Accept.

Figure 4: The Verifier in the Predictable Protocol

Completeness. We show that $(\mathsf{P}, \mathsf{V}')$ is complete based on (a) the completeness of $(\mathsf{P}, \mathsf{V}')$; (b) zero knowledge of $(\mathsf{P}, \mathsf{V}')$; and (c) pseudorandomness of PRG.

Fix any statement $x \in \mathcal{L}$ and corresponding prover witness w. We need to show that in an interaction $\langle \mathsf{P}(x,w), \mathsf{V}'(x) \rangle$, V' rejects with negligible probability. First, by the completeness of (P, V) and the pseudorandomness of PRG, an interaction $\langle \mathsf{P}(x,w), \mathsf{V}_s(x) \rangle\rangle$ is accepting except with negligible probability over the choice of s. Noting that $\mathsf{V}_s(x)$ is b-non-uniform, we can invoke zero knowledge, to deduce that the simulated prover messages $\{\widetilde{\mathsf{p}}_i\}_{i=1}^{\rho}$ make V_s accept with overwhelming probability over the choice of s.

We next argue that the deterministic prover $\mathsf{P}(x,w)$ produces messages $\{\mathsf{p}_i = \widetilde{\mathsf{p}}_i\}_{i=1}^{\rho}$ with overwhelming probability (over the coins of Sim that sampled them). This again follows from zero knowledge. Indeed, we can consider a zero-knowledge distinguisher that has (x, w, s) hardwired, and given messages p_i emulates a conversation of the deterministic $\mathsf{P}(x,w)$ with $\mathsf{V}_s(x)$, and outputs "real" if the corresponding prover messages coincide with p_i, or "simulated" otherwise. If the simulated messages $\widetilde{\mathsf{p}}_i$ are inconsistent with the real prover messages p_i, the distinguisher will tell them apart.

Soundness. We show that $(\mathsf{P}, \mathsf{V}')$ is sound based on (a) the pseudorandomness of PRG; and (b) the soundness of (P, V).

First, note that by pseudorandomness the protocol $(\mathsf{P}, \mathsf{V}_s)$ where s is chosen at random is also sound, since otherwise a cheating prover can be directly used to distinguish real verifier coins form pseudorandom

ones. Next, note that any cheating prover against V' directly implies a cheating prover against V_s (for a random s) by construction. Indeed, V' emulates V_s and accepts only when the prover is consistent with a simulated strategy $\widetilde{\mathsf{p}}_i$ that convinces V_s.[2] Soundness follows.

□

5 Round Reduction and Laconicity

Faonio, Nielsen, and Venturi [FNV17] proved that the round complexity of any predictable argument can be collapsed to one (two messages overall) and that any predictable argument can be made laconic — namely, the prover message is a single bit (or more generally ℓ bits to achieve soundness $\approx 2^{-\ell}$). In this section, we review their transformations and show that they preserve zero knowledge against bounded-non-uniform verifiers. As a corollary of this and the previous section, we deduce that any deterministic-prover zero knowledge argument against bounded-non-uniform verifiers can be collapsed to one round and made laconic.

5.1 Round Reduction

We start by recalling the round-collapsing transformation from [FNV17]. In what follows, let $(\mathsf{P}', \mathsf{V}')$ be a ρ-round predictable argument, the following transformation provides a one round predictable argument (P, V) with a large soundness error (to be dealt with later on). Roughly, the verifier randomly chooses a "cut-off" point i^* for the underlying protocol, and sends all the verifier messages up to, and including, the i^*-th round verifier message to the prover. Being a predictable argument, the verifier is able to do so without requiring the corresponding intermediate prover messages. The prover then iteratively computes the response for each round of the underlying protocol and send over all the prover messages with the verifier accepting if and only if each prover messages corresponds to the predicted prover message.

In [FNV17], it is proven that this protocol has soundness error at most $1 - \rho^{-1} + \mathsf{negl}(\lambda)$. The protocol is then repeated $\omega(\rho \log \lambda)$ times to achieve negligible soundness, using a parallel repetition theorem for one round arguments [BIN97].

Proposition 3. *The round collapsing transformation preserves zero knowledge against b-non-uniform verifiers.*

Proof. We prove the proposition in two steps. First, we show that the transformation in Figure 5 preserves zero-knowledge. Then we show that two-message zero-knowledge against bounded-non-uniform adversaries is closed under parallel repetition.

To prove the first part, let V^* be a b-non-uniform verifier. We show the following claim.

Claim 2. *There exist an efficient simulator S and a verifier V'^* against $\langle \mathsf{P}', \mathsf{V}' \rangle$ such that*

1. *V'^* is $(b + O(1))$-non-uniform and efficiently constructable from V^*.*
2. *For every $x \in \mathcal{L}$,*

$$\mathsf{View}_{\mathsf{V}^*}\langle \mathsf{P}(x, w), \mathsf{V}^* \rangle \equiv \mathsf{S}(\mathsf{View}_{\mathsf{V}'^*}\langle \mathsf{P}'(x, w), \mathsf{V}'^* \rangle) \ .$$

[2] Here we implicitly rely on the fact that the simulator produces an accepting transcript for the deterministic verifier V_s. The deterministic nature of the verifier ensures that the simulator cannot manipulate the verifier's randomness and therefore must produce an accepting transcript is consistent with $\mathsf{V}(\cdot; \mathsf{PRG}(s))$.

Protocol: One Round (P, V)

Common input: Input $x \in \mathcal{L}$, security parameter 1^λ

P**'s auxiliary input:** witness w such that $(x, w) \in R_{\mathcal{L}}$

1. Verifier V
   (a) Samples $i^* \leftarrow\!\!\$\ [\rho]$,
   (b) Samples $(\mathsf{v}_i, \mathsf{b}_i)_{i\in[\rho]} \leftarrow\!\!\$\ \mathsf{V}(x)$.
   (c) Sends $\mathsf{v}_1, \cdots, \mathsf{v}_{i^*}$ to the prover P.
2. Prover P
   (a) For each $i \in [i^*]$, compute $\mathsf{p}_i := \mathsf{P}(x, w, \{\mathsf{v}_j\}_{j\in[i]})$.
   (b) Send $\mathsf{p}_1, \cdots, \mathsf{p}_{i^*}$ to the verifier V.
3. Verifier V accepts if and only if for all $j \in [i^*]$, $\mathsf{p}_j = \mathsf{b}_j$.

Figure 5: Round collapsing transformation.

This claim gives rise to a simulator Sim for (P, V), which simply invokes Sim' of (P, V) on V'^* and then invokes S.

Proof of Claim. We construct $\mathsf{S}, \mathsf{V}'^*$.

V'^*:

1. Emulates V^* and obtains $(\mathsf{v}_1, \ldots, \mathsf{v}_{i^*})$.
2. At each round $i \in [i^*]$, forward v_i to P'.
3. Abort after round i^*.

S:

1. Outputs the randomness of the emulated V^* (can be derived from the randomness of V'^*),
2. as well as the received prover messages $\mathsf{p}_1, \ldots, \mathsf{p}_{i^*}$.

The second property asserted in the claim follows by construction of $\mathsf{S}, \mathsf{V}'^*$ and the construction of P from P' in Figure 5. It is left to see that V'^* is $(b + O(1))$-non-uniform and efficiently constructable from

V^*. $(b + O(1))$-non-uniformity and efficient construction follow from the fact that V^* is b-non-uniform and V'^* uses it as a black box and described by the three code lines above. □

We now prove that closure under parallel repetition.

Claim 3. *For any two-message zero knowledge system (P, V) against b-non-uniform verifiers and a any polynomial ℓ, the ℓ-fold parallel repetition $(\mathsf{P}_{\otimes\ell}, \mathsf{V}_{\otimes\ell})$ is zero knowledge against $(b - O(\log \lambda))$-non-uniform verifiers.*

Proof. In what follows, let Sim be the simulator for the original argument (P, V), and let $\mathsf{V}^*_{\otimes\ell}$ be any $(b - \lambda - O(\log \lambda))$-non-uniform verifier of polynomial running time $t(\lambda)$. We now describe the simulator $\mathsf{Sim}_{\otimes\ell}$ for $(\mathsf{P}_{\otimes\ell}, \mathsf{V}_{\otimes\ell})$. The simulator will use a pseudorandom generator $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^k$, where k is the amount of coins used by $\mathsf{V}^*_{\otimes\ell}$.

$\underline{\mathsf{Sim}_{\otimes\ell}(\mathsf{V}^*_{\otimes\ell}, 1^t, x)}$:

1. Sample a $s \leftarrow\!\!\$\, \{0,1\}^\lambda$.
2. For each $i \in [\ell]$:
   (a) Construct the deterministic verifier $\mathsf{V}^*_{s,i}$ that first derives coins $\mathsf{PRG}(s)$, uses them to emulate $\mathsf{V}^*_{\otimes\ell}$, obtains $\mathsf{v}_1, \ldots, \mathsf{v}_\ell$, and outputs v_i. Let $t' = t + \mathsf{poly}(\lambda)$ be a bound on its running time.
   (b) Sample $\widetilde{\mathsf{p}}_i \leftarrow\!\!\$\, \mathsf{Sim}(\mathsf{V}^*_{s,i}, 1^{t'}, x)$.
3. Output $\widetilde{\mathsf{p}}_1, \ldots, \widetilde{\mathsf{p}}_\ell, \mathsf{PRG}(s)$.

We now prove the validity of $\mathsf{Sim}_{\otimes\ell}$. First, consider an execution between the prover $\mathsf{P}(x, w)$ and verifier $\mathsf{V}^*_s = (\mathsf{V}^*_{s,1}, \ldots, \mathsf{V}^*_{s,\ell})$, and let $\mathsf{p}_1, \ldots, \mathsf{p}_\ell$ denote the prover messages in such an execution. Then by pseudorandomness of PRG,

$$\mathsf{View}_{\mathsf{V}^*_{\otimes\ell}}\langle \mathsf{P}(x,w), \mathsf{V}^*_{\otimes\ell}\rangle \approx_c \mathsf{p}_1, \ldots, \mathsf{p}_\ell, \mathsf{PRG}(s) \ .$$

Noting that $\mathsf{V}^*_{s,i}$ is a program of length at most b and running time at most $t'(\lambda)$, we can invoke the simulation guarantee (P, V). Specifically, we can deduce that

$$\mathsf{p}_1, \ldots, \mathsf{p}_\ell, \mathsf{PRG}(s) \approx_c \widetilde{\mathsf{p}}_1, \ldots, \widetilde{\mathsf{p}}_\ell, \mathsf{PRG}(s) \ .$$

This can be shown by a standard hybrid argument and follows from the fact that $\mathsf{p}_i \approx_c \widetilde{\mathsf{p}}_i = \mathsf{Sim}(\mathsf{V}^*_{s,i}, 1^{t'}, x)$ and that the distinguisher can have (x, w, s) hardwired in order to simulate any other p_j or $\widetilde{\mathsf{p}}_i$. Overall

$$\mathsf{View}_{\mathsf{V}^*_{\otimes\ell}}\langle \mathsf{P}(x,w), \mathsf{V}^*_{\otimes\ell}\rangle \approx_c \mathsf{Sim}_{\otimes\ell}(\mathsf{V}^*_{\otimes\ell}, 1^t, x) \ .$$

□

This complete the proof of Proposition 3. □

5.2 Laconic Prover Messages

As in the previous section, we start by recalling the laconic prover transformation from [FNV17]. In what follows, let $(\mathsf{P}', \mathsf{V}')$ be a one round predictable argument, the following transformation provides a laconic prover predictable argument (P, V) with a soundness error negligibly close to $1/2$, where the prover sends only a single bit. Roughly, the verifier samples a sufficiently large random string γ and sends it to the prover along with the verifier message. The prover responds with a single bit corresponding to the inner product of γ and its own response to the verifier message, with the verifier accepting if only if the bit matches its own computed inner product of γ with the predicted prover message.

Protocol: Laconic Prover (P, V)

Common input: Input $x \in \mathcal{L}$, security parameter 1^λ

P**'s auxiliary input:** witness w such that $(x, w) \in R_{\mathcal{L}}$

1. Verifier V
   (a) Sample $(\mathsf{v}, \mathsf{b}) \leftarrow_\$ \mathsf{V}'(x)$.
   (b) Sample $\gamma \leftarrow_\$ \{0,1\}^{|\mathsf{b}|}$.
   (c) Send v, γ to the prover P.
2. Prover P
   (a) Compute $\mathsf{p} := \mathsf{P}'(x, w, \mathsf{v})$.
   (b) Send $\mathsf{q} := \langle \mathsf{p}, \gamma \rangle$ to the verifier V.
3. Verifier V accepts if and only if $\mathsf{q} = \langle \mathsf{b}, \gamma \rangle$.

Figure 6: Laconic prover transformation.

In [FNV17], it is proven that this protocol has soundness error at most $\frac{1}{2} + \mathsf{negl}(\lambda)$. As we have seen in the previous subsection (Claim 3), the soundness can be amplified in a manner that preserves zero knowledge. Specifically, ℓ repetitions yields a protocol with soundness error at most $2^{-\ell} + \mathsf{negl}(\lambda)$. Therefore, we focus on proving that a single instance of the above transformation preserves zero knowledge.

Proposition 4. *The round collapsing transformation preserves zero knowledge against b-non-uniform verifiers.*

Proof. Let V^* be a b-non-uniform verifier. We show the following claim.

Claim 4. *There exist an efficient simulator S and a verifier V'^* against $\langle \mathsf{P}', \mathsf{V}' \rangle$ such that*

1. V'^* *is* $(b + O(1))$*-non-uniform and efficiently constructable from* V^*.

2. *For every* $x \in \mathcal{L}$*,*
$$\mathsf{View}_{\mathsf{V}^*}\langle \mathsf{P}(x,w), \mathsf{V}^*\rangle \equiv \mathsf{S}(\mathsf{View}_{\mathsf{V}'^*}\langle \mathsf{P}'(x,w), \mathsf{V}'^*\rangle) \ .$$

This claim gives rise to a simulator Sim for (P, V), which simply invokes Sim' of $(\mathsf{P}', \mathsf{V}')$ on V'^* and then invokes S.

Proof of Claim. We construct $\mathsf{S}, \mathsf{V}'^*$.

V'^*:

1. Emulate V^* and obtains (v, γ).
2. Forward v to P'.

S:

1. Outputs the randomness of the emulated V^* (can be derived from the randomness of V'^*),
2. as well as $\langle \mathsf{p}, \gamma\rangle$, where p is the received prover message and γ is derived from the randomness of V^*.

The proof is similar to that of Claim 2 in the previous subsection. The second property asserted in the claim follows by construction of $\mathsf{S}, \mathsf{V}'^*$ and the construction of P from P'. It is left to see that V'^* is $(b + O(1))$-non-uniform and efficiently constructable from V^*. $(b + O(1))$-non-uniformity and efficient construction follow from the fact that V^* is b-non-uniform and V'^* uses it as a black box and described by the two code lines above. □

This completes the proof of Proposition 4. □

References

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *42nd FOCS*, pages 106–115. IEEE Computer Society Press, October 2001.

[BCG+18] Nir Bitansky, Ran Canetti, Sanjam Garg, Justin Holmgren, Abhishek Jain, Huijia Lin, Rafael Pass, Sidharth Telang, and Vinod Vaikuntanathan. Indistinguishability obfuscation for RAM programs and succinct randomized encodings. *SIAM J. Comput.*, 47(3):1123–1210, 2018.

[BCPR14] Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. On the existence of extractable one-way functions. In David B. Shmoys, editor, *46th ACM STOC*, pages 505–514. ACM Press, May / June 2014.

[BIN97] Mihir Bellare, Russell Impagliazzo, and Moni Naor. Does parallel repetition lower the error in computationally sound protocols? In *38th FOCS*, pages 374–383. IEEE Computer Society Press, October 1997.

[BKP19] Nir Bitansky, Dakshita Khurana, and Omer Paneth. Weak zero-knowledge beyond the black-box barrier. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1091–1102. ACM Press, June 2019.

[BOV03] Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 299–315. Springer, Heidelberg, August 2003.

[BP04] Boaz Barak and Rafael Pass. On the possibility of one-message weak zero-knowledge. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 121–132. Springer, Heidelberg, February 2004.

[BP15] Nir Bitansky and Omer Paneth. ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 401–427. Springer, Heidelberg, March 2015.

[BV17] Nir Bitansky and Vinod Vaikuntanathan. A note on perfect correctness by derandomization. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 592–606. Springer, Heidelberg, April / May 2017.

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, Heidelberg, April / May 2002.

[DL20] Hila Dahari and Yehuda Lindell. Deterministic-prover zero-knowledge proofs. Cryptology ePrint Archive, Report 2020/141, 2020.

[FNV17] Antonio Faonio, Jesper Buus Nielsen, and Daniele Venturi. Predictable arguments of knowledge. In Serge Fehr, editor, *PKC 2017, Part I*, volume 10174 of *LNCS*, pages 121–150. Springer, Heidelberg, March 2017.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476. ACM Press, June 2013.

[GK96] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM J. Comput.*, 25(1):169–192, 1996.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, December 1994.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for NIZK. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 97–111. Springer, Heidelberg, August 2006.

[GS18] Sanjam Garg and Akshayaram Srinivasan. A simple construction of iO for turing machines. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 425–454. Springer, Heidelberg, November 2018.

[GVW01] Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *ICALP 2001*, volume 2076 of *LNCS*, pages 334–345. Springer, Heidelberg, July 2001.

[KLW15] Venkata Koppula, Allison Bishop Lewko, and Brent Waters. Indistinguishability obfuscation for turing machines with unbounded memory. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 419–428. ACM Press, June 2015.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

A Predictable Arguments from Honest-Verifier ZK

In Section 4, we showed how to transform any deterministic-prover zero-knowledge (DPZK) protocol into one that is also a predictable argument (PA). In this section, we show that if we start with a weaker notion of deterministic-prover honest verifier zero-knowledge (DP-HVZK) [3] and the existence of an appropriate hard language, we can transform the DP-HVZK protocol into a predictable argument. One caveat of this transformation is that the languages of the DP-HVZK and PA in our transformation will be related, but not identical. As long as the DP-HVZK we start from is for an expressive enough class of languages (e.g. for $\mathsf{NP} \cap \mathsf{coNP}$), we will get a PA for the same class.

Definition 13 (Hard-on-Average Language)**.** *A language $\mathcal{L}$ is hard-on-average if there exist two* PPT *samplers $Y_{\mathcal{L}}, N_{\mathcal{L}}$ where the support of the first is $\mathcal{L}$ and of the second is $\{0,1\}^* \setminus \mathcal{L}$ such that*

$$\left\{x : x \leftarrow Y_{\mathcal{L}}(1^\lambda)\right\}_{\lambda\in\mathbb{N}} \approx_c \left\{x : x \leftarrow N_{\mathcal{L}}(1^\lambda)\right\}_{\lambda\in\mathbb{N}} .$$

We establish the following theorem.

Theorem 6. *If there exists a deterministic-prover honest-verifier zero-knowledge argument (DP-HVZK) for $\mathcal{L} \vee \mathcal{L}_{\mathsf{hard}}$, where $\mathcal{L}_{\mathsf{hard}}$ is a hard-on-average language, then there exists a predictable argument (PA) for $\mathcal{L}$.*

By the fact that both NP and $\mathsf{NP} \cap \mathsf{coNP}$ are closed under OR, we deduce the following corollaries.

Corollary 4. *Assuming DP-HVZK for all of* NP *and hard-on-average languages in* NP*, there is a witness encryption scheme for all of* NP*.*

Corollary 5. *Assuming DP-HVZK for all of* $\mathsf{NP} \cap \mathsf{coNP}$ *and hard-on-average languages in* $\mathsf{NP} \cap \mathsf{coNP}$*, there is a witness encryption scheme for all of* $\mathsf{NP} \cap \mathsf{coNP}$*.*

We note that hard-on-average languages in NP are known to follow from one-way functions, and hard-on-average languages in $\mathsf{NP} \cap \mathsf{coNP}$ are known to follow from one-way permutations.

We now proceed with the proof.

Proof of Theorem 6. To build a predictable argument for $\mathcal{L}$, we use the following primitives:

- A hard language $\mathcal{L}_{\mathsf{hard}}$ given by samplers $(Y_{\mathcal{L}_{\mathsf{hard}}}, N_{\mathcal{L}_{\mathsf{hard}}})$.
- A ρ-round DP-HVZK protocol $\langle \mathsf{P}', \mathsf{V}' \rangle$ for the language $\mathcal{L}_{OR}$ defined below, where the verifier V' sends messages v_i in round i, and the prover P' sends message p_i in round i. We denote by Sim' the corresponding honest-verifier simulator. The language $\mathcal{L}_{OR}$ is defined below,

$$\mathcal{L}_{OR} = \left\{ (x, \widetilde{x}) \;\middle|\; \exists (w, \widetilde{w}) \text{ s.t. } R_{\mathcal{L}}(x, w) = 1 \text{ OR } R_{\mathcal{L}_{\mathsf{hard}}}(\widetilde{x}, \widetilde{w}) = 1 \right\} ,$$

  namely, either the statement x is in $\mathcal{L}$, or $\widetilde{x}$ is in $\mathcal{L}_{\mathsf{hard}}$.

Protocol: PA (P, V)

Common input: Input $x \in \mathcal{L}$, security parameter 1^λ

P**'s auxiliary input:** witness w such that $(x, w) \in R_{\mathcal{L}}$

Verifier V computes

1. $\widetilde{x} \leftarrow Y_{\mathcal{L}_{\mathsf{hard}}}(1^\lambda)$
2. $x' := (x, \widetilde{x})$.
3. $\left(\{(\mathsf{v}_i, \widetilde{\mathsf{p}}_i)\}_{i=1}^{\rho}, \widetilde{r} \right) \leftarrow \mathsf{Sim}'(x')$.

sends $\widetilde{x}$ to the prover P in the first message.

In each round $i \in [\rho]$,

1. Verifier V sends $\mathsf{v_i}$ to the prover P.
2. Prover P computes

   (a) $x' := (x, \widetilde{x})$
   (b) $w' := (w, \perp)$
   (c) $\mathsf{p}_i := \mathsf{P}'(x', w', \{\mathsf{v}_j\}_{j=1}^{i})$

   sends p_i to the verifier V.
3. If for any $i \in [\rho]$, $\mathsf{p}_i \neq \widetilde{\mathsf{p}}_i$, V rejects.

If verifier V has not rejected in all rounds, accept.

Figure 7: Transforming DP-HVZK to PA

The transformation is presented in Figure 7.
Before we proceed with the completeness and soundness, we note that the protocol structure follows that of a predictable argument.

Completeness. We show that (P, V) is complete based on the honest verifier zero-knowledge property of $(\mathsf{P}', \mathsf{V}')$.

Fix any $x \in \mathcal{L}$ and the corresponding witness w, a yes-instance $\widetilde{x} \in \mathcal{L}_{\mathsf{hard}}$, and let $x' = (x, \widetilde{x})$. Let $\widetilde{\mathsf{p}}_1, \ldots, \widetilde{\mathsf{p}}_\rho$ denote the messages and $\widetilde{r}$ denote the verifier randomness simulated by $\mathsf{Sim}'(x')$. We argue that the deterministic prover $\mathsf{P}(x, w)$ produces messages $\{\mathsf{p}_i = \widetilde{\mathsf{p}}_i\}_{i=1}^{\rho}$ with overwhelming probability (over the coins of Sim'). This follows from zero knowledge. Consider a distinguisher that has (x, w) hardwired,

[3] Only zero-knowledge against honestly behaving verifiers.

and given messages p_i and verifier randomness $\widetilde{r}$ emulates a conversation of the deterministic $\mathsf{P}'(x, w)$ with $\mathsf{V}'(x; \widetilde{r})$, and outputs "real" if the corresponding prover messages coincide with p_i, or "simulated" otherwise. If the simulated messages $\widetilde{\mathsf{p}}_i$ are inconsistent with the real prover messages p_i, the distinguisher will tell them apart.

Soundness. We show that (P, V) is sound based on the completeness, soundness and zero knowledge of $(\mathsf{P}', \mathsf{V}')$, as well as the hardness of $\mathcal{L}_{\mathsf{hard}}$.

Fix any $x \notin \mathcal{L}$ and cheating prover P^*. We prove that P^* fails to convince $\mathsf{V}(x)$ of accepting, except with negligible probability. We consider several hybrid experiments transitioning from a real interaction to an ideal interaction. We will show that when moving from one hybrid to the next the prover's chance of convincing the verifier does not decrease by more than a negligible amount. Then we will show that the chance that $\mathsf{V}(x)$ is convinced the final (ideal interaction) hybrid is negligible.

Hyb_0**:** This is a real interaction between P^* and $\mathsf{V}(x)$.

Hyb_1**:** In this hybrid, once V samples a simulated transcript $\widetilde{\mathsf{p}}_1, \ldots, \widetilde{\mathsf{p}}_\rho, \widetilde{r} \leftarrow\!\!\$\, \mathsf{Sim}(x')$, it emulates an execution of $\mathsf{V}'(x'; \widetilde{r})$ with the simulated prover messages and checks whether it is accepting. If it is not, V rejects immediately.

We argue that the probability that P^* convinces $\mathsf{V}(x)$ to accept in this hybrid is negligibly close to that in Hyb_0. For this purpose, we argue that with overwhelming probability $\mathsf{Sim}(x')$ samples an accepting transcript. This is shown based on completeness and zero knowledge of $(\mathsf{P}', \mathsf{V}')$. Specifically, recall that $\mathsf{V}(x)$ samples $\widetilde{x} \in \mathcal{L}_{\mathsf{hard}}$ and thus $x' = (x, \widetilde{x}) \in \mathcal{L}_{OR}$. By the completeness of $(\mathsf{P}', \mathsf{V}')$, in an interaction between $\mathsf{V}'(x')$ and $\mathsf{P}'(x', w')$ where $w' = (\bot, \widetilde{w})$ and $\widetilde{w}$ is a witness for $\widetilde{x}$, the prover convince V' with overwhelming probability. It then follows from zero knowledge of $(\mathsf{P}', \mathsf{V}')$ that $\mathsf{Sim}(x')$ also generates an accepting transcript with overwhelming probability; otherwise, we can non-uniformly fix $\widetilde{x}, \widetilde{w}$ and construct a distinguisher that violates zero knowledge.

Hyb_2**:** In this hybrid, the verifier V does not insist that the prover P^* is consistent with the simulated messages $\widetilde{\mathsf{p}}_1, \ldots, \widetilde{\mathsf{p}}_\rho$. Instead, it emulates $\mathsf{V}'(x'; \widetilde{r})$, and accepts if the messages sent by P^* convince V'.

The probability that V accepts in this hybrid is at least as large as the probability it accepts in Hyb_1. Indeed, any execution that would have been accepted in the previous hybrid Hyb_1 is in particular an execution in which $\mathsf{V}'(x'; \widetilde{r})$ is convinced and thus is also accepted in the current Hyb_2.

Hyb_3**:** In this hybrid, the verifier V does not check that the simulated $\widetilde{\mathsf{p}}_1, \ldots, \widetilde{\mathsf{p}}_\rho, \widetilde{r}$ make $\mathsf{V}'(x'; \widetilde{r})$ accept. (In particular, the simulated prover messages $\widetilde{\mathsf{p}}_1, \ldots, \widetilde{\mathsf{p}}_\rho$ are ignored altogether, and only the simulated coins $\widetilde{r}$ are used).

The probability that $\mathsf{V}(x)$ accepts in this hybrid is at least as large as the probability it accepts in the previous hybrid, as we have only removed a verifier test.

Hyb_4**:** In this hybrid, instead of sampling simulated coins $\widetilde{r}$ using $\mathsf{Sim}'(x')$, V samples truly random coins r.

The probability that $\mathsf{V}(x)$ accepts in this hybrids is negligibly close to that in the previous hybrid. This follows from zero knowledge of $(\mathsf{P}', \mathsf{V}')$. Indeed, since $x' \in \mathcal{L}_{OR}$, the simulated honest verifier coins $\widetilde{r}$ are pseudorandom.

Hyb_5**:** In this hybrid, $\mathsf{V}(x)$ samples a no-instance $\widetilde{x} \leftarrow N_{\mathcal{L}_{\mathsf{hard}}}$ instead of a yes-instance. By the indistinguishability of $Y_{\mathcal{L}_{\mathsf{hard}}}$ and $N_{\mathcal{L}_{\mathsf{hard}}}$, the probability that P^* convinces $\mathsf{V}(x)$ to accept in this hybrid is negligibly close to that in Hyb_4.

We now argue that the probability that P^* convinces $\mathsf{V}(x)$ to accept in Hyb_5 is negligible. Note that in Hyb_5 it holds that both $x \notin \mathcal{L}$ and $\widetilde{x} \notin \mathcal{L}_{\mathsf{hard}}$ and thus $x' = (x, \widetilde{x}) \notin \mathcal{L}_{OR}$. For P^* to convince $\mathsf{V}(x)$ of accepting in Hyb_5, it must convince $\mathsf{V}'(x'; r)$ of accepting, when V' uses truly random coins. By the soundness of $(\mathsf{P}', \mathsf{V}')$ this occurs with negligible probability. Soundness follows. □