A Note on Non-Interactive Zero-Knowledge from CDH

Geoffroy Couteau, Abhishek Jain, Zhengzhong Jin, Willy Quach

A Note on Non-Interactive Zero-Knowledge from CDH

Geoffroy Couteau*
Université Paris Cité, CNRS, IRIF

Abhishek Jain†
Johns Hopkins University

Zhengzhong Jin ‡
MIT

Willy Quach§
Northeastern University

Abstract

We build non-interactive zero-knowledge (NIZK) and ZAP arguments for all NP where soundness holds for infinitely-many security parameters, and against uniform adversaries, assuming the subexponential hardness of the Computational Diffie-Hellman (CDH) assumption. We additionally prove the existence of NIZK arguments with these same properties assuming the polynomial hardness of both CDH and the Learning Parity with Noise (LPN) assumption. In both cases, the CDH assumption does not require a group equipped with a pairing.

Infinitely-often uniform security is a standard byproduct of commonly used non-black-box techniques that build on disjunction arguments on the (in)security of some primitive. In the course of proving our results, we develop a new variant of this non-black-box technique that yields improved guarantees: we obtain explicit constructions (previous works generally only obtained existential results) where security holds for a relatively dense set of security parameters (as opposed to an arbitrary infinite set of security parameters). We demonstrate that our technique can have applications beyond our main results.

*Email: couteau@irif.fr
†Email: abhishek@cs.jhu.edu
‡Email: zzjin@mit.edu
§Email: quach.w@northeastern.edu

Contents

1 Introduction

Zero-knowledge (ZK) proofs [GMR85] allow a prover to convince a verifier about the validity of a statement without revealing any other information. They are studied in two flavors – interactive proofs, where the prover and the verifier exchange messages in a protocol, and non-interactive proofs, where the prover sends a single message to the verifier. The latter notion, referred to as *non-interactive zero knowledge* (NIZK) [DMP88, BFM88], has been central to the popularity of ZK proofs due to its wide-ranging applications, including advanced encryption schemes [NY90, DDN91], signature schemes [BMW03, BKM06] and anonymous blockchains [BCG+14].

NIZKs are a fascinating object in cryptography. Despite a long line of research starting more than three decades ago [DMP88, BFM88, FLS90, BY93, CHK03, GOS06b, GOS06a, GR13, SW14, CL18, CCRR18, CCH+19, PS19, CKU20, BKM20, JJ21], remarkably, the cryptographic complexity of NIZKs is not well understood. We do not yet know whether NIZKs are in Minicrypt or Cryptomania.[1] In fact, we do not even know how to construct NIZKs from all standard assumptions known to imply public-key encryption. Significant progress, however, has recently been made on this front: we now know NIZKs for NP from learning with errors [CCH+19, PS19] as well as the (sub-exponential) Decisional Diffie Hellman (DDH) assumption [JJ21], which substantially adds to the prior list of assumptions known to imply NIZKs.

DDH is the strongest assumption in the discrete-logarithm family of assumptions. A weaker assumption – known to imply public-key encryption – is the Computational Diffie Hellman (CDH) assumption. With the aim of further enhancing our understanding of the relationship between NIZKs and public-key encryption, we ask the following question:

Do there exist NIZKs for NP *based on CDH?*

A positive resolution to this question would also help diminish the gap between NIZKs and their designated-verifier[2] counterpart [PsV06]. Indeed, the latter are already known from CDH [QRW19, CH19, KNYY19] as well as all other assumptions known to imply NIZKs (see [LQR+19] and references therein).

ZAPs. ZAPs [DN00] are two-round public-coin proof systems in the plain model that guarantee witness indistinguishability (WI) [FS87], i.e., a proof for a statement with two or more witnesses does not reveal which of the witnesses was used in the computation of the proof.

Dwork and Naor [DN00] proved that ZAPs are equivalent to NIZK *proofs* in the common random string model. Thus, ZAPs are known from the same assumptions also known to imply NIZK proofs. Very recently, computationally-sound ZAPs, aka ZAP *arguments* were constructed based on quasi-polynomial LWE [BFJ+20, LVW19, GJJM20], and subexponential DDH (and variants) [JJ21, CKSU21]. As in the case of NIZKs, however, constructing ZAP arguments based on CDH remains an open problem.

CDH vs DDH. Our work follows a well-established line of research on building cryptographic primitives from CDH, when feasibility from DDH is already known. The motivation for this line of work stems from the relative gap between CDH and DDH and the difficulty of building cryptography from CDH.

CDH is a weaker assumption than DDH, and believed to be strictly weaker for some choices of groups, e.g. $\mathbb{G} = \mathbb{Z}_q^*$ or the source group $\mathbb{G}$ of a symmetric pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ (where, in both cases, DDH is broken). In fact, the hardness of CDH is closely related to that of the discrete logarithm assumption: [Mau94] proved that the non-uniform hardness of CDH in any group $\mathbb{G}$ of known order q is equivalent to the non-uniform hardness of computing discrete logarithms in $\mathbb{G}$, and [BL96] proved that the uniform versions are equivalent in the (large) subexponential regime, both results assuming a plausible and widely believed conjecture on the distribution of smooth numbers. In that sense, despite being a public-key assumption, CDH is morally equivalent to the hardness of computing discrete logarithms, while there are no such connections for DDH (unless computing discrete logarithm is easy in all groups where DDH is known not to hold).

Furthermore, CDH seems to be significantly less expressive than DDH in terms of enabling advanced functionalities. While there has been recent progress on building CDH counterparts to fundamental primitives known from DDH, (e.g. trapdoor functions [GH18], maliciously-secure oblivious transfer [DGH+20], or private information retrieval [BCM22]), there are still many fundamental primitives known from DDH that have no CDH counterpart (e.g. lossy trapdoor functions [PW08], somewhere statistically-binding hash functions [OPWW15], or 2-round private information retrieval [DGI+19], and more generally, most primitives that are built from the dual mode paradigm).

[1] Throughout this work, we focus on NIZKs in the common reference string (CRS) model. In the Random Oracle model, NIZKs are known to be in Minicrypt.

[2] In a designated-verifier NIZK, the verifier receives a private verification key that is sampled together with the CRS, which can be used to verify many proofs.

1.1 Our Main Result

In this work, we make progress towards resolving the above question. We demonstrate that NIZKs and ZAP arguments for NP with infinitely-often security against uniform adversaries do exist based on the subexponential CDH assumption, without requiring the existence of a pairing.

Theorem 1.1 (Informal)**.** *Under the subexponential CDH assumption, there exist:*

- *a subexponentially-often secure, uniform NIZK argument for all* NP *in the common random string model;*
- *a subexponentially-often secure, uniform ZAP argument for all* NP*.*

More precisely, our assumption in Theorem 1.1 states that no subexponential-time adversary can compute random Diffie-Hellman tuples, either over $\mathbb{Z}_q^*$ (or any subgroup, or, even more generally, over any group (family) with exponentiation computable in TC^0) or any (family of) elliptic curves (without requiring a pairing), with better than subexponential probability. Note that similar restrictions on cryptographic groups appear in the construction of NIZKs and ZAP arguments from DDH ([JJ21]).

Our NIZK satisfies (1) *infinitely-often* adaptive soundness against *uniform* efficient cheating provers, and (2) (standard, computational) adaptive, multi-theorem zero-knowledge. Our ZAP argument satisfies (1) *infinitely-often* non-adaptive soundness against *uniform* efficient cheating provers, and (2) (standard, computational) adaptive witness indistinguishability. Moreover, the set of security parameters where we argue soundness is at least *subexponentially dense*, in a sense we specify below.

1.2 On Infinitely-Often Security

Infinitely-often security refers to a setting where a primitive is secure on infinitely-many security parameters (as opposed to the traditional notion of almost everywhere security, where security holds for all large enough parameters). It shows up naturally as a consequence of a common non-black-box technique where the *insecurity* of a cryptographic primitive is used to argue the *security* of another primitive (where the attacker against the insecure primitive is used either explicitly in the construction, or implicitly in the security analysis), since (the standard notion of) insecurity of a primitive only guarantees the existence of an attacker successful on infinitely-many security parameters. This behavior shows up in many recent works, where it is sometimes explicitly pointed out, and sometimes disregarded as a minor technical subtlety. Early examples include the work of [RTV04] (which shows that if there is a reduction of key-agreement to OWFs, then there exists a mildly-blackbox reduction of infinitely often key-agreement to OWFs) and the work of [MPS10] (infinitely often one-way functions from constant-round weak coin-flipping protocol). There are also many recent examples, such as [HNO+18] (infinitely-often key agreement from nontrivially-correlated 2-party protocols), [Den17] (either the Feige-Shamir protocol is concurrent zero-knowledge, or injective one-way functions imply an infinitely often key agreement), [Zha19] (a post-quantum collision-resistant hash function is either "collapsing", or it implies infinitely often quantum lightning schemes), or the works of [KY18, RV22] (which construct (distributional and standard, respectively) collision-resistant hash functions from multi-collision-resistant hash functions). A recent work of [PV20] shows that hard-on-average NP languages imply infinitely-often hard-on-average *promise-true* NP search problems. Closer in spirit to our work, [CKU20] gives a construction of infinitely-often NIZK from an exponentially-strong KDM-style variant of the discrete logarithm assumption.

In all these works, the existential result is generally non-constructive (whenever the construction itself relies on the *existence* of an adversary against some primitive) and holds only for infinitely many security parameters, with no guarantee on the *density* of these parameters (*i.e.* the secure parameters could be separated by arbitrarily fast-growing gaps).

Our Work. In contrast, a surprising and interesting feature of our work is that we manage to improve significantly on both these caveats: while we employ a non-black-box technique similar in spirit to these works, our constructions

- are fully explicit (*i.e.*, our result is constructive)
- are proven secure on a set E of security parameters which can be shown to be *reasonably dense* in $\mathbb{N}$.

Concretely, in our main construction of NIZKs and ZAP arguments from the subexponential hardness of CDH, the set E of secure security parameters for our NIZKs and ZAPs is at least *subexponentially dense*: there exists a constant $0 < K < 1$ such that, for all $\lambda \in \mathbb{N}$, $\left[\lambda, 2^{\lambda^K}\right] \cap E \neq \emptyset$. In that sense, we say that the constructions of Theorem 1.1 are *subexponentially-often secure.*

A caveat of our technique is that soundness only holds against *uniform* cheating provers, while usual notions of security allow adversaries to use a non-efficiently computable advice. This seems an unavoidable consequence of aiming for uniform NIZK algorithms (so that honest parties do not require non-uniform advice to run our NIZK). Consequently, our construction can only use the existence of a *uniform* attacker against some primitive, which turns into building soundness on uniform computational assumptions. We refer to the technical overview for more details.

Looking ahead, we prove Theorem 1.1 by carefully combining two central ingredients. The first is a NIZK (resp. ZAP argument) which is secure assuming the subexponential hardness of DDH [JJ21]. The second is a template to build NIZKs from cryptographic groups, introduced in [CH19, QRW19, KNYY19] (resp. ZAPs, when combined with [DN00]). Our main technical tool is the construction of a *universal breaker*, which we believe to be of independent interest. In our setting, our universal breaker allows to somewhat efficiently test, given a security parameter λ, whether DDH is "secure" with respect to λ, for a specific definition of security. We refer to the technical overview and Section 5.1 for more precise statements.

On the generality of our approach. While we mostly focus on NIZKs and ZAP arguments, our approach is modular, and we believe that a similar technique could be used to refine the results of many of the previous works that achieved infinitely often security, such as those listed above. In general, when our approach can be applied, it should lead to explicit constructions with security on a dense set of security parameters, but with two caveats: it would only prove security against uniform adversaries, and would rely on superpolynomial hardness assumptions (because our techniques inherently require some mild use of complexity leveraging). Though the results stated in Theorem 1.1 are our main results, we believe that our new techniques are a conceptual contribution of independent interest. Below, we further illustrate the generality of our techniques and obtain some additional results, both within and outside the setting of NIZKs.

1.3 Further Results

NIZKs from CDH+LPN. First, replacing the NIZK of [JJ21] with the one of [BKM20], we directly obtain the following:

Theorem 1.2 (Informal). *Under both the superpolynomial CDH and polynomial Learning Parity with Noise (LPN) assumptions, there exists a NIZK argument for all* NP *satisfying (1) superpolynomially-often adaptive soundness against uniform efficient cheating provers, and (2) (standard, computational) adaptive, multi-theorem zero-knowledge.*

We also show, through a different argument, the following existential (non-constructive) result. We refer to Section 6 for more details.

Theorem 1.3 (Informal). *At least one of the following two statements holds:*

- *Under the polynomial LPN assumption, there exists a NIZK argument for all* NP *satisfying (1) infinitely-often non-adaptive soundness against uniform efficient cheating provers and (2) statistical zero-knowledge;*
- *Under the polynomial CDH assumption, there exists a NIZK proof for all* NP *satisfying (1) statistical adaptive soundness and (2) (standard, computational) adaptive, multi-theorem zero-knowledge.*

Unlike Theorem 1.1, neither Theorem 1.2 nor Theorem 1.3 suffer from restrictions over cryptographic groups supported.

Promise-true hard-on-average search problems from hard-on-average languages. Eventually, we revisit the recent work of [PV20], which showed that if there exists a hard-on-average NP language, then there also exists an (infinitely-often) hard-on-average *promise-true* distributional NP search problem – or, using their terminology, proving theorems that are guaranteed to be true is no easier than proving theorems in general. Applying our new technique, we obtain an explicit variant of their main theorem that starts from a (mildly superpolynomially) hard-on-average NP language, and builds a promise-true distributional NP search problem which is sound on a *superpolynomially dense* set of security parameters:

Theorem 1.4 (Informal). *Given any superpolynomially-secure uniformly hard-on-average* NP *language, there is an explicit construction of a promise-true distributional* NP *search problem which is uniformly superpolynomially-often hard-on-average.*

1.4 Roadmap

We present an overview of our techniques in Section 2. We introduce notations and recall useful results from prior work in Section 3. In Section 4, we present generic constructions related to DDH breakers and $\overline{\text{DDH}}$-based NIZKs. In Section 5, we present our main construction along with our main technical tools. In Section 6, we present a purely existential result corresponding to Theorem 1.3. In Section 7, we show how to adapt our construction to the setting of elliptic curves. In Section 8, we present a construction of promise-true NP search problem from a hard-on-average NP language.

2 Technical Overview

Designated-Verifier NIZKs from CDH. Our starting point is the construction of designated-verifier NIZKs for NP from CDH [CH19, QRW19, KNYY19]. In a nutshell, these works, assuming CDH, reduce building a NIZK for all NP to the task of building a NIZK for the DDH language: an instance $(g, g^\alpha, g^\beta, g^\gamma)$ belongs to the language if $\gamma = \alpha \cdot \beta \bmod p$, where p is the order of the group. Unfortunately, we do not know how to build NIZKs for the DDH language assuming only the hardness of CDH. Instead, [CH19, QRW19, KNYY19] observe that *designated-verifier* NIZKs for the DDH language can be constructed [CS02], which in turn, yields a designated-verifier NIZK for NP from CDH.

In fact, this approach can yield *publicly-verifiable* NIZKs for NP if the verifier can efficiently check whether group elements form DDH tuples. This observation already yields a NIZK for NP when the group is equipped with a (symmetric) bilinear map: the verifier can check whether an input from the source group is a DDH tuple by comparing the appropriate pairings $e(g, g^\gamma)$ and $e(g^\alpha, g^\beta)$ [CHK03, QRW19]. Notably, the bilinear map and the target group are only used in the verification algorithm, and security properties of the NIZK only rely on the hardness of CDH in the source group.

Alternatively, *if DDH were broken over the group* (without pairings), then we could also obtain NIZKs for NP based on CDH via this approach. In this case, the verifier could perform the required checks using the DDH breaker. For convenience, we refer to NIZKs obtained in this manner as $\overline{\text{DDH}}$-based NIZKs.

NIZKs from DDH, and a Disjunction Argument. Recently, [JJ21] provided a construction of NIZK for all NP from (subexponential) DDH.[3][4] For convenience, we will refer to this NIZK as a DDH-based NIZK.

This brings us to the following attempt for constructing NIZKs for NP from CDH. Fix a single (family of) cryptographic groups for which CDH holds. Then,

- either "DDH is secure", in which case the DDH-based NIZK of [JJ21] is secure,
- or "DDH is broken", which allows to build a $\overline{\text{DDH}}$-based NIZK, assuming CDH!

One could be tempted to conclude that this disjunction approach yields a NIZK for all of NP from CDH. Unfortunately, this conclusion does not directly hold, because the statements "DDH is secure" and "DDH is broken", as (imprecisely) stated above, are not negations of each other. Nevertheless, this dichotomy serves as the key starting point behind our result.

A Closer Look. There are several mismatches in the definitions of "secure" and "broken" above.

1. A first mismatch relates to the *success probabilities* of breakers. An adversary falsifying the security of DDH is only ensured to work with some small, non-negligible (or even subexponentially small) probability. In contrast, the breaker needed to instantiate the $\overline{\text{DDH}}$-based NIZK from CDH needs to work *with very high probability* on *worst-case* inputs (since the breaker is used by the verifier).
2. A second mismatch is that hardness assumptions are usually stated as to handle *non-uniform* adversaries. Consequently, an adversary falsifying the security of DDH would only yield a non-uniform DDH breaker, and thus the resulting verifier for the $\overline{\text{DDH}}$-based NIZK would be non-uniform.

[3] [JJ21] actually provides two NIZKs. The first one provides statistical zero-knowledge, but only non-adaptive soundness. The second is adaptively-sound and computationally zero-knowledge. Because our approach can only yield computational zero-knowledge, we will use the second version.

[4] Technically, [JJ21] imposes mild restrictions on the supported cryptographic groups, that we also inherit. We will ignore this for the sake of this overview, and refer to Section 3.1 for more details.

3. A third mismatch is that, in order to falsify DDH being secure in the usual sense, it suffices to exhibit an adversary that breaks DDH with sufficiently good advantage on *infinitely many* security parameters. In fact, it is not even clear, given such an adversary, how to efficiently determine on which security parameters the breaker works, without, say, a bound on its runtime. Such a bound could be provided as non-uniform advice to the verification algorithm, but would again result again in a $\overline{\text{DDH}}$-based NIZK with a non-uniform verifier. Furthermore, such an adversary would only help in constructing a $\overline{\text{DDH}}$-based NIZK on only infinitely many security parameters.

The first mismatch can be taken care of using the random self-reducibility of DDH,[5] which allows us to amplify the success probability of any "weak" breaker to a "strong" one. Given that the DDH-based NIZK of [JJ21] relies on the subexponential hardness of DDH, the resulting amplified breaker runs in subexponential time. Then, after relying on complexity leveraging for the resulting $\overline{\text{DDH}}$-based NIZK in order to make this breaker efficient, soundness follows from the (mildly stronger) subexponential hardness of CDH.

It is unfortunately less clear how to handle the two other issues. Still, the approach above already gives a NIZK with *non-uniform, non-explicit* algorithms, which is *infinitely-often* secure based on the subexponential hardness of CDH — an already interesting result.[6]

A Universal DDH Breaker. Towards tackling the drawbacks of the previous construction, our first step is to characterize more precisely subsets of security parameters such that DDH is secure, and ones such that DDH is broken. Doing so opens the hope of obtaining a new construction which, for every security parameter, uses either the DDH-based NIZK if the security parameter is secure, and the $\overline{\text{DDH}}$-based NIZK otherwise.

Our crucial observation is that, for a suitable notion of security, one can somewhat efficiently test whether a security parameter is secure. We do so through the construction of a *universal breaker* UnivBreak, which (with overwhelming probability) breaks DDH on every security parameter such that *some* good breaker exists, and fails only when no good enough breaker exists. Our construction is inspired by classic constructions of universal cryptographic objects. Namely, it iterates through all small Turing machines of size say $\leq \lfloor \log \lambda \rfloor$, and tests whether they efficiently break DDH. If a good breaker exists, it uses one of them to break DDH, and otherwise states that the security parameter is secure. Standard concentration bounds intuitively ensure that if a good breaker exists, then UnivBreak finds a good breaker (with overwhelming probability), and if no good breakers exist, then UnivBreak is not fooled into using a bad breaker (with overwhelming probability). Still, in order to fully define our universal breaker, we need to define more precisely the set of "small Turing machines" it will consider. Equivalently, we now seek to formally define a set SECURE of security parameters for which DDH is secure.

We observe that if SECURE relates to the *uniform* security of DDH, then breakers on $\overline{\text{SECURE}} := \mathbb{N} \setminus \text{SECURE}$ are also uniform. The intuition is then that, fixing any uniform breaker $\mathcal{A}$ on $\overline{\text{SECURE}}$, UnivBreak will eventually run $\mathcal{A}$ when given as input a large enough security parameter $\lambda \geq 2^{|\mathcal{A}|}$, allowing us to use the $\overline{\text{DDH}}$-based NIZK. Furthermore, we can show that the DDH-based NIZK is sound on SECURE, albeit only against *uniform* cheating provers. Ultimately, this is the reason our constructions are only sound against uniform cheating provers.

This is unfortunately not yet sufficient. The reason is that UnivBreak is called on a fixed security parameter λ, and that security on any *fixed* security parameter is an inherently *non-uniform* notion of security. For instance, there could exist a family of uniform breakers $\mathcal{A}_i$ that respectively break DDH on all *large enough* parameters in $\{\lambda \notin \text{SECURE} \mid \lambda \geq \lambda_i\}$ but such that λ_i grows with their size $|\mathcal{A}_i|$ as Turing machines, eg $\lambda_i = |\mathcal{A}_i|$. In particular, we cannot rule out that, for *all* $\lambda \notin$ SECURE, UnivBreak never runs any $\mathcal{A}_i$ on input $\lambda \geq \lambda_i$. This, in turn, could imply that for all security parameters, UnivBreak has low advantage, or even wrongly concludes that some input parameter is secure.

Our solution is to modify the definition of SECURE to bound the "non-uniformity" of breakers. Namely, whether some fixed security parameter λ belongs to SECURE now only depends on whether there exists an adversary with small description $\leq \lfloor \log \lambda \rfloor$ as a Turing machine that breaks DDH on λ.

A separate issue is that iterating over polynomial-time adversaries with non-negligible advantage is not well defined, because the notions of polynomial-time and negligible advantage are *asymptotic*. For instance, for any fixed λ, there will always exist a (uniform) polynomial-time machine breaking DDH on λ with non-negligible advantage. Instead, we define SECURE using $(t(\lambda), \varepsilon(\lambda))$-security (namely, considering adversaries running in time $t(\lambda)$ with advantage $\varepsilon(\lambda)$) with *fixed* functions $t = t(\lambda)$ and $\varepsilon = \varepsilon(\lambda)$, where t is superpolynomial, and ε is inverse superpolynomial, so that (t, ε)-security (asymptotically) implies standard polynomial security.

[5] Assuming the (family of) group is of prime order.

[6] Formalizing such a statement turns out to require quite a bit of care, because of subtleties specific to the precise soundness statement of [JJ21]. We will not develop these difficulties further here, as we will directly prove a stronger statement below.

Importantly, we can argue that DDH is uniformly hard on the set SECURE, which will allow us to argue *uniform* soundness of the DDH-based NIZK.

Summing up, our universal breaker UnivBreak, on input a security parameter λ, tests all $t(\lambda)$-time machines of size $\leq \lfloor \log \lambda \rfloor$, and checks whether their advantage in breaking DDH is larger than $\varepsilon(\lambda)$, using $\approx 1/\varepsilon^2(\lambda)$ runs. If some machine has enough advantage, UnivBreak uses this machine to compute its output; and if no such machine exists, UnivBreak indicates that λ is secure. Note that UnivBreak is somewhat efficient in that it runs in superpolynomial time $\approx t(\lambda)/\varepsilon^2(\lambda)$. Intuitively, UnivBreak indicates that λ is secure whenever $\lambda \in \mathsf{SECURE}$, and breaks DDH with advantage $\approx \varepsilon(\lambda)$ whenever $\lambda \notin \mathsf{SECURE}$. It turns out this intuition is slightly inaccurate, but will suffice for the purpose of this overview. We refer to Section 5.1 for more details, and a formal treatment.

A Subexponentially-Often NIZK from Subexponential CDH. We now use our universal breaker to build a (weak) NIZK from CDH. A proof in our scheme simply consists of both a DDH-based proof π_{DDH} and a $\overline{\mathsf{DDH}}$-based proof $\pi_{\overline{\mathsf{DDH}}}$. The verifier, given the security parameter λ, tests the universal breaker UnivBreak on λ. If the universal breaker fails to produce an output bit, the verifier verifies π_{DDH}. Otherwise, it amplifies the advantage of UnivBreak in order to verify $\pi_{\overline{\mathsf{DDH}}}$. Note that the construction is *fully explicit*, and features *uniform* algorithms.

Completeness and zero-knowledge follow from correctness of UnivBreak (which is ensured to produce outputs with good advantage whenever it produces an output) and the completeness and zero knowledge properties of the DDH-based NIZK and the $\overline{\mathsf{DDH}}$-based NIZK.

One could hope that the NIZK above satisfies *uniform* soundness on *all* security parameters, Indeed, the *uniform* hardness of DDH holds over the set of parameters SECURE by definition: given any PPT uniform adversary $\mathcal{A}$ of size s, thanks to $t(\lambda)$ (resp. $\varepsilon(\lambda)$) being superpolynomial (resp. inverse superpolynomial), we have that for all large enough $\lambda \geq 2^s$ such that $\lambda \in \mathsf{SECURE}$, the advantage of $\mathcal{A}$ on λ is at most $\varepsilon(\lambda)$. Conversely, if $\lambda \notin \mathsf{SECURE}$, then soundness holds thanks to guarantees on UnivBreak and soundness of the $\overline{\mathsf{DDH}}$-based NIZK.

Unfortunately, the argument above does not hold, because the (amplified) DDH breaker given by UnivBreak runs in super-polynomial time $\approx t(\lambda)/\varepsilon^2(\lambda)$. In fact, the security of the DDH-NIZK of [JJ21] requires assuming the *subexponential security* of DDH, which requires to set ε as inverse subexponential. Thus, the resulting verification algorithm building on UnivBreak actually runs in subexponential time. As a result, we need to rely on complexity leveraging whenever calling the $\overline{\mathsf{DDH}}$-based NIZK to make the verification algorithm efficient. Namely, our NIZK for security parameter λ calls the $\overline{\mathsf{DDH}}$-based NIZK on security parameter $\lambda' := \lfloor \log^{1/c}(\lambda) \rfloor$ for some constant $0 < c < 1$. First, this introduces the need to assume subexponential hardness of CDH (to argue zero-knowledge of the $\overline{\mathsf{DDH}}$-based NIZK). Second, this resulting mismatch of the security parameters used by the DDH-based NIZK and the $\overline{\mathsf{DDH}}$-based NIZK prevents us from arguing soundness on all security parameters: it could be that $\lambda \notin \mathsf{SECURE}$ and $\lambda' \in \mathsf{SECURE}$, in which case we do not know how to argue soundness of any of the two NIZKs, whenever running our NIZK on security parameter λ.

Still, we obtain a NIZK which is secure on infinitely-many security parameters. In fact, we can argue that the set of secure parameters for the NIZK is *subexponentially dense* in the following sense. For every security parameter λ, either $\lambda \in \mathsf{SECURE}$ or $\lambda \notin \mathsf{SECURE}$. Consequently, either our NIZK is sound on λ (corresponding to $\lambda \in \mathsf{SECURE}$), or it is sound on $\overline{\lambda} := 2^{\lambda^c}$ (corresponding to $\lambda \notin \mathsf{SECURE}$). This is because, in that latter case, on input $\overline{\lambda}$, our NIZK calls the $\overline{\mathsf{DDH}}$-based NIZK and UnivBreak on parameter $\lfloor \log^{1/c}\left(\overline{\lambda}\right) \rfloor = \lambda \notin \mathsf{SECURE}$, and therefore soundness of the $\overline{\mathsf{DDH}}$-based NIZK applies.[7] Overall, this ensures that the *relative gap* between two consecutive parameters for which our NIZK is secure is at most subexponential. We refer to Theorem 5.1 for a formal statement.

Variant: NIZK from CDH and LPN. Our approach is quite modular in the DDH-based NIZK we start from. In particular, starting with the construction of [BKM20] which is secure assuming both the polynomial hardness of DDH and LPN, we obtain a "superpolynomially dense" NIZK where the gap between secure parameters is only superpolynomial, and where security holds assuming both the superpolynomial hardness of CDH and the polynomial hardness of LPN.[8]

Variant: ZAP Arguments from CDH. We observe that, given a DDH breaker, the $\overline{\mathsf{DDH}}$-based NIZKs from [CH19, QRW19, KNYY19] use a uniform common random string, and are statistically sound. Thus, any efficient

[7] The actual statement we prove is slightly more technical, due to subtleties in the proof of soundness of [JJ21]. We refer to Theorem 5.1 for more details.

[8] We only know how to instantiate our universal breaker using a superpolynomial (resp. inverse superpolynomial) function t (resp. ε) so that $\lambda \in \mathsf{SECURE}$ implies that DDH is polynomially hard on λ against uniform adversaries. We therefore still need to rely on complexity leveraging, resulting in a superpolynomial gap.

DDH breaker implies a ZAP for all NP based on CDH via [DN00]. Moreover, [JJ21] builds ZAP arguments for all NP assuming the subexponential hardness of DDH. Thus, we can apply the same blueprint as for the construction of NIZK. The verifier message consists of verifier messages for both the DDH-based ZAP argument and the (complexity-leveraged) $\overline{\text{DDH}}$-based ZAP, and the prover replies with two proofs. The verifier then runs UnivBreak, and verifies one of the proofs accordingly. A similar analysis gives that the resulting ZAP argument is subexponentially often (non-adaptively) sound against uniform cheating provers,[9] and witness indistinguishable assuming the subexponential hardness of CDH.

3 Preliminaries

Notation. Throughout this paper, λ denotes the security parameter. A probabilistic polynomial time algorithm (PPT, also denoted *efficient* algorithm) runs in time polynomial in the (implicit) security parameter λ. A function f is *negligible* if for any positive polynomial p there exists a bound $B > 0$ such that, for any integer $k \geq B$, $|f(k)| \leq 1/|p(k)|$. Given a finite set S, the notation $x \xleftarrow{\$} S$ means a uniformly random assignment of an element of S to the variable x. For a positive integer n, m such that $n < m$, we denote by $[n]$ the set $\{1, \cdots, n\}$. We will sometimes explicitly refer to the random coins r used by a probabilistic algorithm M by writing $M(\cdot; r)$.

3.1 Diffie-Hellman Assumptions

Cryptographic Groups. Let DHGen be a deterministic algorithm which on input 1^λ returns a description $\mathcal{G} = (\mathbb{G}, p)$ where $\mathbb{G}$ is a cyclic group of prime order p. Throughout the paper, we will fix DHGen, and therefore a family of groups $\{\mathcal{G}_\lambda\}_{\lambda\in\mathbb{N}}$. Unless specified otherwise, we will assume throughout this work that the prime-order group $\mathbb{G}$ has exponentiation in TC^0. This notably includes (subgroups of) $\mathbb{Z}_q^*$ for $q \in \mathbb{N}$, which includes its subgroup of quadratic residues. Looking ahead, we consider in Section 7 a variant for elliptic curves.

As is usually (implicitly) assumed for cryptographic groups, we will suppose that, for all $\lambda \in \mathbb{N}$, there exists an efficient *oblivious sampling algorithm* $\mathsf{Sample}(1^\lambda; r) \mapsto g$ which we denote $g \xleftarrow{\$} \mathbb{G}_\lambda$. Formally, this requires that there exists an efficient algorithm Equivocate such that the two following distributions are within negligible statistical distance: $(r, \mathsf{Sample}(1^\lambda; r)) \approx_s (\mathsf{Equivocate}(g), g \leftarrow \mathbb{G})$, where r is uniformly random. Note that this follows whenever the description of a uniformly random group element is itself a uniformly random string. This allows to securely view uniformly random group elements as uniformly random strings (up to considering the internal random coins used by Sample).

The computational Diffie-Hellman assumption is defined as follows.

Definition 3.1 (CDH Assumption)**.** *We say that the computational Diffie-Hellman (CDH) assumption holds relative to DHGen if for all PPT adversaries $\mathcal{A}$ and all large enough security parameters λ,*

$$\Pr\left[\mathcal{G} = \mathit{DHGen}(1^\lambda), g \xleftarrow{\$} \mathbb{G}, \alpha, \beta \xleftarrow{\$} \mathbb{Z}_p : g^{\alpha\beta} \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta)\right] \leq \mathsf{negl}(\lambda).$$

We also define similarly the decisional Diffie-Hellman assumption:

Definition 3.2 (DDH Assumption)**.** *We say that the decisional Diffie-Hellman (DDH) assumption holds relative to DHGen if for all PPT adversaries $\mathcal{A}$ and all large enough security parameters λ,*

$$\Pr\left[\begin{array}{c}\mathcal{G} = \mathit{DHGen}(1^\lambda), g \xleftarrow{\$} \mathbb{G}, \alpha, \beta, \gamma \xleftarrow{\$} \mathbb{Z}_p, b \xleftarrow{\$} \{0,1\}, \\ \delta \leftarrow b\gamma + (1-b)\alpha\beta, b' \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\delta)\end{array} : b = b'\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Throughout the paper, whenever there are no ambiguities about DHGen, we will denote by $\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda)$, for any adversary $\mathcal{A}$ and security parameter λ, the probability

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda) := \Pr\left[\begin{array}{c}\mathcal{G} = \mathit{DHGen}(1^\lambda), g \xleftarrow{\$} \mathbb{G}, \alpha, \beta, \gamma \xleftarrow{\$} \mathbb{Z}_p, b \xleftarrow{\$} \{0,1\}, \\ \delta \leftarrow b\gamma + (1-b)\alpha\beta, b' \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\delta)\end{array} : b = b'\right] - \frac{1}{2}.$$

Subexponential security. In the definition of CDH (resp. DDH), if the inequality is strengthened to hold against all probabilistic 2^{λ^c}-time adversaries $\mathcal{A}$ with advantage at most $2^{-\lambda^c}$ for some constant $0 < c < 1$, we refer to the corresponding assumption as the (2^{λ^c}-)subexponential CDH (resp. DDH) assumption.

[9] This is because the ZAP argument of [JJ21] is only non-adaptively sound.

Infinitely-often security. In the definition of CDH (resp. DDH), if the inequality is instead required to hold only for (all large enough elements of) an infinite set of security parameters $E \subseteq \mathbb{N}$, we refer to the corresponding assumption as the infinitely-often CDH (resp. DDH) assumption with respect to E, and denote it io-CDH (resp. io-DDH).

Uniform security. By default, when we quantify over PPT adversaries $\mathcal{A}$, PPT refers to *non-uniform* adversaries: families $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$ of boolean circuits such that $|\mathcal{A}_\lambda| = \mathrm{poly}(\lambda)$ for every $\lambda \in \mathbb{N}$. In this work, we will in fact mostly consider a weaker, *uniform* notion of security, where the adversaries $\mathcal{A}$ are modelled as probabilistic Turing machines. When the CDH (resp. DDH) assumption is only required to hold against all uniform PPT adversaries, we call *uniform CDH* (resp. *uniform DDH*) the corresponding assumption.

3.2 Non-Interactive Zero-Knowledge

A (publicly-verifiable) non-interactive zero-knowledge (NIZK) argument system for an NP relation R, with associated language $\mathcal{L}(R) = \{x \mid \exists w, (x,w) \in R\}$ is a 3-tuple of efficient algorithms (Setup, Prove, Verify), where Setup outputs a common reference string, $\mathsf{Prove}(\mathsf{crs}, x, w)$, given the crs, a statement x, and a witness w, outputs a proof π, and $\mathsf{Verify}(\mathsf{crs}, x, \pi)$, on input the crs, a word x, and a proof π, outputs a bit indicating whether the proof is accepted or not. A NIZK argument system satisfies the following: completeness, adaptive soundness, and adaptive multi-theorem zero-knowledge properties:[10]

- A non-interactive argument system (Setup, Prove, Verify) for an NP relation R satisfies *completeness* if for every $(x,w) \in R$,

$$\Pr[\mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\lambda, 1^{|x|}), \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w) : \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1] \geq 1 - \mathsf{negl}(\lambda).$$

- A non-interactive argument system (Setup, Prove, Verify) for an NP relation R satisfies *adaptive soundness* if for any PPT $\mathcal{A}$ and any large enough security parameter λ,

$$\Pr\left[\begin{array}{l} \mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\lambda, 1^{|x|}), (x,\pi) \xleftarrow{\$} \mathcal{A}(\mathsf{crs}) \ : \\ \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \wedge x \notin \mathcal{L} \end{array}\right] \leq \mathsf{negl}(\lambda).$$

- A non-interactive argument system (Setup, Prove, Verify) for an NP relation R satisfies (computational, statistical) *adaptive multi-theorem zero-knowledge* if for all (computational, statistical) $\mathcal{A}$, there exists a PPT simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ such that if we run $\mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\lambda, 1^{|x|})$ and $\overline{\mathsf{crs}} \xleftarrow{\$} \mathsf{Sim}_1(1^\lambda, 1^{|x|})$, then we have $|\Pr[\mathcal{A}^{O_0(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}) = 1] - \Pr[\mathcal{A}^{O_1(\overline{\mathsf{crs}},\cdot,\cdot)}(\mathsf{crs}) = 1]| = \mathsf{negl}(\lambda)$, where $O_0(\mathsf{crs}, x, w)$ outputs $\mathsf{Prove}(\mathsf{crs}, x, w)$ if $(x,w) \in R$ and $\perp$ otherwise, and $O_1(\overline{\mathsf{crs}}, x, w)$ outputs $\mathsf{Sim}_2(\overline{\mathsf{crs}}, x)$ if $(x,w) \in R$ and $\perp$ otherwise.

Whenever $\mathsf{Setup}(1^\lambda)$ outputs a uniformly random string crs, we say that the NIZK is in the *common random string* model.

Infinitely-often, uniform, subexponential NIZKs. If we relax the definition of correctness (resp. adaptive soundness) to hold only for (all large enough elements of) an infinite set of security parameters $E \subseteq \mathbb{N}$, we say that the NIZK satisfies *infinitely-often correctness* (resp. *infinitely-often adaptive soundness*) *with respect to* E, and refer to the NIZK as an *infinitely-often NIZK*. One could analogously define infinitely-often zero-knowledge, but we will not need it in this work.

If soundness holds only against uniform PPT adversaries, we say that the NIZK is a *uniform NIZK* (similarly, we will not need to consider uniform zero-knowledge in this work).

Finally, if soundness and zero-knowledge hold against subexponential-time adversaries (resp. subexponential-time adversaries with subexponential advantage), we say that the NIZK is subexponentially secure (resp. strongly subexponentially secure).

NIZKs in the hidden-bits model. We use the following result regarding the existence of NIZKs in the hidden-bits model (HBM). Since the full definition of NIZK in the HBM will not be required in our work, we refer the readers to [FLS90] for more details.

Theorem 3.3 (NIZK for all of NP in the HBM [FLS90])**.** *Let λ denote the security parameter and let $k = k(\lambda)$ be any positive integer-valued function. Then, unconditionally, there exists NIZK proof systems for any* NP *language $\mathcal{L}$ in the HBM that uses* $\mathsf{hb} = k \cdot \mathrm{poly}(\lambda, |x|)$ *hidden bits with soundness error $\varepsilon \leq 2^{-k\cdot\lambda}$, where λ denotes the security parameter and* poly *is a function related to the* NP *language $\mathcal{L}$, and that are perfectly zero-knowledge.*

[10] Intuitively, multi-theorem zero-knowledge ensures that a simulator can provide *many* simulated proofs under a *common* simulated CRS.

3.3 Verifiable Pseudorandom Generators

Verifiable pseudorandom generators have been introduced in [DN00]. Their definition has been refined in [CH19, QRW19, KNYY19], and slightly relaxed in [CKU20]. Below, we recall the definition from [CKU20].

Definition 3.4 (Verifiable Pseudorandom Generator)**.** *Let $\delta(\lambda)$ and $s(\lambda)$ be positive valued polynomials. A $(\delta(\lambda), s(\lambda))$-verifiable pseudorandom generator (*VPRG*) is a four-tuple of efficient algorithms* (Setup, Stretch, Prove, Verify) *such that*

- $\mathsf{Setup}(1^\lambda, m)$*, on input the security parameter (in unary) and a polynomial bound $m(\lambda) \geq s(\lambda)^{1+\delta(\lambda)}$, outputs a set of public parameters* pp *(which contains 1^λ);*
- $\mathsf{Stretch}(\mathsf{pp})$*, on input the public parameters* pp*, outputs a triple* $(\mathsf{pvk}, x, \mathsf{aux})$*, where* pvk *is a public verification key of length $s(\lambda)$, x is an m-bit pseudorandom string, and* aux *is an auxiliary information;*
- $\mathsf{Prove}(\mathsf{pp}, \mathsf{aux}, i)$*, on input the public parameters* pp*, auxiliary informations* aux*, an index $i \in [m]$, outputs a proof π;*
- $\mathsf{Verify}(\mathsf{pp}, \mathsf{pvk}, i, b, \pi)$*, on input the public parameters* pp*, a public verification key* pvk*, an index $i \in [m]$, a bit b, and a proof π, outputs a bit β;*

which is in addition complete*,* hiding*, and* binding*, as defined below.*

Definition 3.5 (Completeness of a VPRG)**.** *For any $i \in [m]$, a* complete VPRG *scheme* (Setup, Stretch, Prove, Verify) *satisfies, for all large enough λ:*

$$\Pr\left[\begin{array}{l} \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda, m), \\ (\mathsf{pvk}, x, \mathsf{aux}) \xleftarrow{\$} \mathsf{Stretch}(\mathsf{pp}), \\ \pi \xleftarrow{\$} \mathsf{Prove}(\mathsf{pp}, \mathsf{aux}, i), \end{array} : \mathsf{Verify}(\mathsf{pp}, \mathsf{pvk}, i, x_i, \pi) = 1 \right] \geq 1 - \mathsf{negl}(\lambda).$$

Definition 3.6 (Statistical Binding Property of a VPRG)**.** *Let* (Setup, Stretch, Prove, Verify) *be a* VPRG*. A* VPRG *is* statistically binding *if there exists a (possibly inefficient)* extractor Ext *such that for any (potentially unbounded) $\mathcal{A}$ and for all large enough λ, it holds that*

$$\Pr\left[\begin{array}{l} \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda, m), \\ (\mathsf{pvk}, i, \pi) \xleftarrow{\$} \mathcal{A}(\mathsf{pp}), \\ x_i \leftarrow \mathsf{Ext}(\mathsf{pp}, \mathsf{pvk}) \end{array} : \mathsf{Verify}(\mathsf{pp}, \mathsf{pvk}, i, 1 - x_i, \pi) = 1 \right] \leq \mathsf{negl}(\lambda).$$

Definition 3.7 (Hiding Property of a VPRG)**.** *A* VPRG *scheme* (Setup, Stretch, Prove, Verify) *is* hiding *if for any $i \in [m]$ and any PPT adversary $\mathcal{A}$ that outputs bits, and for all large enough λ, it holds that:*

$$\Pr\left[\begin{array}{l} \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda, m), \\ (\mathsf{pvk}, x, \mathsf{aux}) \xleftarrow{\$} \mathsf{Stretch}(\mathsf{pp}), \\ (\pi_j \xleftarrow{\$} \mathsf{Prove}(\mathsf{pp}, \mathsf{aux}, j))_j \end{array} : \mathcal{A}(\mathsf{pp}, \mathsf{pvk}, i, (x_j, \pi_j)_{j \neq i}) = x_i \right] \leq 1/2 + \mathsf{negl}(\lambda).$$

Infinitely-often, subexponential VPRGs. If we relax the definition of completeness (resp. binding) to hold only for (all large enough elements of) an infinite set of security parameters $E \subseteq \mathbb{N}$, we say that the VPRG satisfies *infinitely-often completeness* (resp. *infinitely-often binding*) *with respect to E*, and refer to the VPRG as an *infinitely-often VPRG*. We note that one can analogously define infinitely-often hiding, but we will not need it in this work. We furthermore say that a VPRG is a subexponential VPRG, if (1) completeness error (resp. binding error, distinguishing advantage against hiding) are all inverse subexponential, and (2) if hiding holds against subexponential-time adversaries $\mathcal{A}$.

Statistical VS Computational Binding. In definition Definition 3.6, we define the binding notion of VPRGs to hold against any (potentially unbounded) adversaries $\mathcal{A}$. In [CH19, CKU20], VPRGs are defined with a more general *computational* binding requirement, which only holds against polynomial-time adversaries $\mathcal{A}$. And indeed, looking slightly ahead, Theorem 3.8 does extend to compile computationally binding VPRGs to *computationally sound* NIZK arguments for NP. In this work, we define VPRGs with statistical binding because (1) our constructions of VPRGs from DDH breakers in Section 4.2 will be statistically binding; and (2) we crucially use that our resulting NIZK from NP is statistically sound to obtain ZAPs for NP (Theorem 5.12).

From VPRGs to NIZKs for NP. The following shows that VPRG are sufficient to construct NIZKs for all of NP.

Theorem 3.8 ((δ, s)-VPRGs $\Rightarrow$ NIZKs for all of NP)**.** *Fix an NIZK proof system for any* NP *language $\mathscr{L}$ in the HBM that uses* $\mathsf{hb} = \mathsf{hb}(\lambda, |x|)$ *hidden bits with soundness error $\varepsilon \leq 2^{-\lambda}$ where* $\mathsf{hb} \geq \lambda$ *w.l.o.g. Suppose that a $(\delta(\lambda), s(\lambda))$-verifiable pseudorandom generator where $s(\lambda) \geq \max\{\lambda, (\mathsf{hb}^2/\lambda)^{1/\delta(\lambda)}\}$ exists. Then, there exist statistically adaptively sound with soundness error $2^{-\lambda}$ and adaptively multi-theorem zero-knowledge NIZK proofs for the* NP *relation $\mathscr{L}$.*

*If instead the $(\delta(\lambda), s(\lambda))$-*VPRG *satisfies infinitely-often completeness and infinitely-often binding with respect to some infinite subset $E \subseteq \mathbb{N}$, then there exists an infinitely-often NIZK which is statistically, adaptively sound with soundness error $2^{-\lambda}$ with respect to E, and adaptively multi-theorem zero-knowledge.*

Furthermore, if the public parameters of the verifiable pseudorandom generator are uniformly random, then the resulting NIZK is in the common random *string model.*

Proof. The proof follows readily from [FLS90] and [DN00, CH19]. It can be checked from Theorem 16 of [CH19] that we can combine the NIZK in the HBM for the NP relation $\mathscr{L}$ with any VPRG that satisfies $s^{1+\delta} > (1 + s/\lambda)\mathsf{hb} + \mathsf{hb}^2/\lambda$ in order to construct a statistically sound, adaptive single-theorem non-interactive witness indistinguishable (NIWI) proof for the NP relation $\mathscr{L}$. Working out the equation and taking into account that s needs to be at least λ-bits, the condition on s in our statement is sufficient. Then, by using [FLS90], we can convert an adaptive single-theorem NIWI proof into an adaptive multi-theorem NIZK proof assuming the existence of pseudorandom generators (which are by definition implied by VPRGs). To obtain a NIZK with soundness error $2^{-\lambda}$, we use a λ-wise parallel repetition, using that the construction above is statistically sound.

The proof extends directly to VPRGs that are correct and binding on any infinite subset $E \subseteq \mathbb{N}$, giving an infinitely-often NIZK with respect to E. □

Since the existence of an NIZK in the HBM for any NP language $\mathscr{L}$ is implied by Theorem 3.3, the above shows that VPRGs with some mild condition on $\delta(\lambda)$ and $s(\lambda)$ implies existence of NIZKs for any NP language $\mathscr{L}$.

Remark 3.9 (NIZKs for Large Statements)**.** Theorem 3.8 can be readily extended to the setting of NIZKs with statements of size *subexponential* in the security parameter, namely $|x| = 2^{\lambda^c}$ for some constant c satisfying $0 < c < 1$, assuming an appropriately strong VPRG. More precisely, assume the existence of a subexponential VPRG with quantitatively stronger completeness, binding, and hiding $2^{-\lambda^{c'}}$, where $c < c' < 1$ is a constant. Then there exists a NIZK with honest algorithms running in time $\mathsf{poly}(\lambda, |x|)$, with completeness error (resp. zero-knowledge distinguishing advantage) $2^{-O(\lambda^{c'})} = \mathsf{negl}(\lambda, |x|)$, and statistical soundness error $2^{-\lambda}$. Moreover, if hiding of the VPRG hiding holds against $2^{-\lambda^{c'}}$-time adversaries, then the resulting NIZK is zero-knowledge against adversaries running in time $2^{O(\lambda^{c'})}$.

The statement above is directly obtained by adapting the proof of Theorem 3.8, starting instead with a VPRG with subexponential-length output $s^{1+\delta} \geq (1 + s/\lambda)\mathsf{hb}(\lambda, |x|) + \mathsf{hb}^2(\lambda, |x|)/\lambda$, where we recall that hb is a polynomial in $\lambda, |x|$.

3.4 NIZKs and ZAP arguments from DDH

We recall here the result of [JJ21], which we adapt to our setting. Recall that we assume our cryptographic groups to have exponentiation in TC^0 (see Section 3.1). We refer to Section 7 for elliptic curve counterparts (without requiring a pairing).

Theorem 3.10 (NIZK from DDH [JJ21])**.** *There exists a constant $L > 0$ such that the following holds. For any constant $0 < c < 1$, and for all $\lambda \in \mathbb{N}$, define the set* $\mathsf{TOWER}_\lambda = \mathsf{TOWER}_\lambda(c, L) := \{\lambda\} \cup \{\lambda^{(c/2)^{i/2}}\}_{i \in [L]}$*. For any infinite set $E \subseteq \mathbb{N}$, define :*

$$E_{\mathsf{TOWER}} = \bigcup_{\lambda \in E} \mathsf{TOWER}_\lambda.$$

Suppose that, for any (uniform) PPT adversary $\mathcal{A}$, there exists λ^* such that for all $\lambda_{\mathsf{TOWER}} \in E_{\mathsf{TOWER}}$ satisfying $\lambda_{\mathsf{TOWER}} \geq \lambda^*$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^{\lambda_{\mathsf{TOWER}}}) \leq 2^{-(\lambda_{\mathsf{TOWER}})^c}.$$

Then:

- *there exists a NIZK for all* NP *satisfying perfect completeness, infinitely-often adaptive soundness w.r.t. E (against uniform cheating provers),*[11] *and computational zero-knowledge against non-uniform verifiers;*

[11] See the paragraph on infinitely-often security in Section 3.2 for a definition of soudness w.r.t. an infinite set E.

- *there exists a ZAP argument for all* NP *satisfying perfect completeness, infinitely-often non-adaptive soundness w.r.t. E, and statistical adaptive witness indistinguishability.*

In other words, [JJ21] builds, given a (uniform) cheating prover on security parameter λ, a (uniform) DDH breaker for at least one security parameter in TOWER_λ. Then, the theorem above captures the resulting security statement associated to infinitely-often soundness.

4 DDH Breakers and VPRGs

In this section, we introduce building blocks that we use in our constructions. Throughout this section, we only assume that our cryptographic groups are of prime order, and we do not assume that exponentiation can be computed in TC^0. We mainly prove the following result, which intuitively states that algorithms breaking DDH can be turned into an approriate NIZK:

Lemma 4.1. *Let $t = t(\lambda)$ be any positive integer-valued function, and $0 < \varepsilon = \varepsilon(\lambda) < 1/2$ be any function such that, for all λ, $t(\lambda)/\varepsilon^2(\lambda) \leq 2^{\lambda^c}$ for some constant $0 < c < 1$.*

Assume $\mathcal{A}$ is a Turing machine running in time $t(\lambda)$, such that there exists an infinite set $E \subseteq \mathbb{N}$ such that for all $\lambda \in E$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda) \geq \varepsilon(\lambda).$$

Let $E' = \{2^{\lambda^c}\}_{\lambda \in E}$, and let c' be any constant such that $c < c' < 1$.

Then, assuming the $2^{\lambda^{c'}}$-subexponential hardness of CDH (Section 3.1), there exists a NIZK in the common random string model, which is infinitely-often correct and statistically adaptively sound with respect to E', and satisfying computational, adaptively, multi-theorem zero-knowledge.

Furthermore, if for all λ, $t(\lambda)/\varepsilon^2(\lambda) = \mathsf{poly}(\lambda)$, then assuming the polynomial hardness of CDH, there exists a NIZK in the common random string model, which is infinitely-often correct and statistically adaptively sound with respect to E, and computationally adaptively, multi-theorem zero-knowledge.

Remark 4.2 (Non-Uniform Security)**.** We note here that Lemma 4.1 results in a (subexponentially-often) NIZK with security holding against non-uniform adversaries: adaptive soundness holds against computationally unbounded provers, and adaptive, multi-theorem zero-knowledge hold against efficient non-uniform verifiers.

Remark 4.3 (Large Statement Sizes)**.** Similar to Remark 3.9, the resulting NIZK, when ran over input statements of size up to $|x| = 2^{\lambda^c}$, remains correct, statistically sound, and subexponentially zero-knowledge, and where the running time of the honest algorithms is $\mathsf{poly}(\lambda, |x|)$. Looking ahead, this is done by combining the subexponential VPRG of Lemma 4.6 with the proof of Theorem 3.8.

In Section 4.1, we show how to amplify the success probability of weak DDH breakers. In Section 4.2, we show to build a VPRG from strong DDH breakers.

4.1 Amplification of DDH Breakers

First, we prove a generic result on amplifying the success probability of (weak) DDH breakers.

Given a group description $\mathcal{G} = (\mathbb{G}, p) = \mathsf{DHGen}(1^\lambda)$ and a security parameter λ, let $\mathcal{S}_{\mathsf{DH}}(\lambda)$ be the set of DDH four-tuples: $\mathcal{S}_{\mathsf{DH}}(\lambda) = \{(g, g^\alpha, g^\beta, g^\gamma) : \gamma = \alpha\beta\}$. Let $T(\lambda) = \mathsf{poly}(\lambda)$ be such that $|\mathbb{G}| < 2^{T(\lambda)}$.

Lemma 4.4 (Amplification of DDH Breakers.)**.** *Let $t = t(\lambda)$ be any positive integer-valued function, and $\varepsilon = \varepsilon(\lambda)$ be any function such that $0 < \varepsilon(\lambda) < 1/2$ for all λ.*

Assume $\mathcal{A}$ is a Turing machine running in time $t(\lambda)$, such that there exists an infinite set $E \subseteq \mathbb{N}$ such that for all $\lambda \in E$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda) \geq \varepsilon(\lambda).$$

Then there exists a Turing machine $\overline{\mathcal{A}} = \overline{\mathcal{A}}(t, \varepsilon)$ running in time $t(\lambda)/\varepsilon^2(\lambda) \cdot \mathsf{poly}(\lambda)$ such that, for all large enough security parameters $\lambda \in E$ and all DDH tuples $(g, g^\alpha, g^\beta, g^\gamma) \in \mathcal{S}_{\mathsf{DH}}(\lambda)$:

$$\Pr_r\left[b \xleftarrow{\$} \overline{\mathcal{A}}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\gamma; r) : \ b = 1\right] \geq 1 - \mathsf{negl}(\lambda) \cdot 2^{-4T(\lambda)}.$$

Furthermore, for all large enough security parameters $\lambda \in E$, and all non-DDH tuples $(g, g^\alpha, g^\beta, g^\gamma) \in \mathbb{G}^4 \setminus \mathcal{S}_{\mathsf{DH}}(\lambda)$,

$$\Pr_r\left[b \xleftarrow{\$} \overline{\mathcal{A}}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\gamma; r) : \ b = 0\right] \geq 1 - \mathsf{negl}(\lambda) \cdot 2^{-4T(\lambda)},$$

We call any machine satisfying these properties a strong DDH breaker *with respect to E.*

Brief overview. The proof is slightly more involved than the standard DDH amplification approach. We start from the default strategy: we run the weak DDH breaker $\mathcal{A}$ on many rerandomized versions of the input to $\overline{\mathcal{A}}$ (using an appropriate rerandomization such that a DDH tuple becomes a fresh DDH tuple, and a non-DDH tuple becomes a fresh random tuple). However, the inputs to $\mathcal{A}$ are now either all random, or all DDH tuples. So we further randomize the inputs, by randomly switching them to a freshly uniform tuple, with probability $1/2$, and check whether $\mathcal{A}$ correctly guesses whether the input was switched. We then check whether these guesses deviate significantly from the distribution of uniform bits using concentration bounds: they should not deviate when starting with a random DDH tuple, as the output to $\mathcal{A}$ is then independent of the switch, and should deviate significantly otherwise by assumption on $\mathcal{A}$. The proof follows.

Proof. Set $B(\lambda) = 100 \cdot \frac{T(\lambda)+\omega(\log\lambda))}{\varepsilon^2}$. On input $(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\delta)$, the machine $\overline{\mathcal{A}}$ proceeds as follows:

- For $i = 1$ to $B(\lambda)$, it samples $(u_i, v_i, w_i, z_i, t_i) \xleftarrow{\$} \mathbb{Z}_p^5$ and a fresh random bit $s_i \xleftarrow{\$} \{0,1\}$.
- It computes
$$\sigma_i \leftarrow \mathcal{A}(1^\lambda, \mathcal{G}, g^{z_i}, g^{z_i(\alpha w_i+u_i)}, g^{z_i(\beta+v_i)}, g^{z_i(\delta w_i+\alpha v_i w_i+\beta u_i+u_i v_i+s_i t_i)}).$$
- It outputs 1 if and only if
$$\sum_{i=1}^{B(\lambda)} s_i \oplus \sigma_i < B(\lambda) \cdot \frac{1}{2}(1-\varepsilon).$$

$\overline{\mathcal{A}}$ requires $B(\lambda) = \mathrm{poly}(\lambda)/\varepsilon^2(\lambda)$ executions of the time-$t(\lambda)$ algorithm $\mathcal{A}$, and therefore runs in time $t(\lambda)/\varepsilon^2(\lambda) \cdot \mathrm{poly}(\lambda)$. For the rest of the proof, suppose that $\lambda \in E$.

For any i, let us write $h_i \leftarrow g^{z_i}$, $a_i \leftarrow \alpha w_i + u_i$, and $b_i \leftarrow \beta + v_i$. We have

$$(g^{z_i}, g^{z_i(\alpha w_i+u_i)}, g^{z_i(\beta+v_i)}, g^{z_i(\delta w_i+\alpha v_i w_i+\beta u_i+u_i v_i+s_i t_i)}) = (h_i, h_i^{a_i}, h_i^{b_i}, h_i^{a_i b_i-(\alpha\beta-\delta) w_i+s_i t_i}),$$

where h_i and (a_i, b_i) are distributed as fresh random elements of $\mathbb{G}$ and $\mathbb{Z}_p^2$ respectively, independently of w_i. Therefore,

1. if $\alpha\beta - \delta = 0$, *i.e.* $(g, g^\alpha, g^\beta, g^\delta) \in \mathcal{S}_{\mathrm{DH}}$, then $(h_i, h_i^{a_i}, h_i^{b_i}, h_i^{a_i b_i-(\alpha\beta-\delta) w_i+s_i t_i})$ is distributed as a fresh random DDH tuple whenever $s_i = 0$, and as a fresh random four-tuple whenever $s_i = 1$.
2. otherwise, when $(g, g^\alpha, g^\beta, g^\delta) \in \mathbb{G}^4 \setminus \mathcal{S}_{\mathrm{DH}}$, $(h_i, h_i^{a_i}, h_i^{b_i}, h_i^{a_i b_i-(\alpha\beta-\delta) w_i+s_i t_i})$ is always distributed as a fresh random four-tuple, independently of the value of s_i.

Note that the statement of case 2 crucially relies on the fact that the output $\mathbb{G}$ of DHGen is a group of prime order p: otherwise, $w_i(\alpha\beta - \delta)$ would not be distributed uniformly over $\mathbb{Z}_p$ in general. In case 2, the distribution of the i-th input to $\mathcal{A}$ is perfectly independent of s_i (as the $s_i t_i$ is additively masked by the uniformly random term $w_i(\alpha\beta - \delta)$), hence for every i, $\Pr[s_i = \sigma_i] = \Pr[s_i \oplus \sigma_i = 0] = 1/2$. Therefore, the bits $(\sigma_i \oplus s_i)_i$ are independent uniformly random bits. Hence, $\mathbb{E}[\sum_i s_i \oplus \sigma_i] = B(\lambda)/2$, and by a standard Chernoff bound,

$$\Pr\left[\sum_i (\sigma_i \oplus s_i) < \frac{B(\lambda)}{2} \cdot (1-\varepsilon)\right] < \exp\left(-\frac{\varepsilon^2 \cdot B(\lambda)}{4}\right) < 2^{-4T(\lambda)} \cdot \mathrm{negl}(\lambda),$$

by assumption on $\mathcal{A}$ and where $B(\lambda) = 100 \cdot \frac{T(\lambda)+\omega(\log\lambda))}{\varepsilon^2(\lambda)}$. Therefore, in case 2 ($(g, g^\alpha, g^\beta, g^\delta) \in \mathbb{G}^4 \setminus \mathcal{S}_{\mathrm{DH}}$), $\overline{\mathcal{A}}$ outputs 0 with probability at least $1 - 2^{-4T(\lambda)} \cdot \mathrm{negl}(\lambda)$.

In case 1, on the other hand, the tuples $(s_i, h_i, h_i^{a_i}, h_i^{b_i}, h_i^{a_i b_i-(\alpha\beta-\delta) w_i+s_i t_i})_i$ are distributed exactly as independent random samples from $\{(b, g, g^\alpha, g^\beta, g^\delta) : g \xleftarrow{\$} \mathbb{G}, \alpha, \beta, \gamma \xleftarrow{\$} \mathbb{Z}_p, b \xleftarrow{\$} \{0,1\}, \delta \xleftarrow{\$} \gamma + (1-b)\alpha\beta\}$. Hence, by assumption on $\mathcal{A}$, for every $i \le B(\lambda)$,

$$\Pr\left[\sigma_i \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathcal{G}, h_i, h_i^{a_i}, h_i^{b_i}, h_i^{a_i b_i-(\alpha\beta-\delta) w_i+s_i t_i}) : \sigma_i = s_i\right] > \frac{1}{2} + \varepsilon,$$

Therefore, $\mathbb{E}\left[\sum_i (\sigma_i \oplus s_i)\right] < (1/2 - \varepsilon) \cdot B(\lambda)$. This implies that

$$B(\lambda) \cdot \frac{1}{2}(1-\varepsilon) > \mathbb{E}\left[\sum_i (\sigma_i \oplus s_i)\right] \cdot \frac{1-\varepsilon}{1-2\cdot\varepsilon} > \mathbb{E}\left[\sum_i (\sigma_i \oplus s_i)\right] \cdot (1 + \varepsilon/2),$$

for a large enough λ. Then, by a standard Chernoff inequality,

$$\Pr\left[\sum_i (\sigma_i \oplus s_i) \geq B(\lambda)\cdot\frac{1}{2}(1-\varepsilon)\right] < \exp\left(-\frac{\varepsilon^2\cdot B(\lambda)}{24}\right) < 2^{-4T(\lambda)}\cdot \mathsf{negl}(\lambda),$$

again by definition of ε, hence $\overline{\mathcal{A}}$ outputs 1 with probability at least $1 - 2^{-4T(\lambda)}\cdot\mathsf{negl}(\lambda)$. This concludes the proof. □ □

Next, we make a simple observation:

Claim 4.5. *For all sufficiently large security parameter λ, denoting $\mathcal{G} = (\mathbb{G}, p) \leftarrow DHGen(1^\lambda)$, and under the same assumption on $\mathcal{A}$ as in Lemma 4.4:*

$$\Pr_r\left[\exists(g, g^\alpha, g^\beta, g^\gamma) \in \mathcal{S}_{\mathsf{DH}} : \overline{\mathcal{A}}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\gamma; r)\right] = 0] \leq \mathsf{negl}(\lambda), \text{ and}$$
$$\Pr_r\left[\exists(g, g^\alpha, g^\beta, g^\gamma) \in \mathbb{G}^4 \setminus \mathcal{S}_{\mathsf{DH}} : \overline{\mathcal{A}}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\gamma; r)\right] = 1] \leq \mathsf{negl}(\lambda).$$

Proof. This follows immediately from a straightforward union bound over all elements of $\mathcal{S}_{\mathsf{DH}}$ and of $\mathbb{G}^4 \setminus \mathcal{S}_{\mathsf{DH}}$, using the fact that $|\mathcal{S}_{\mathsf{DH}}| < |\mathbb{G}^4 \setminus \mathcal{S}_{\mathsf{DH}}| < 2^{4T(\lambda)}$ since $|G| < 2^{T(\lambda)}$. □ □

4.2 VPRGs from Strong DDH Breakers

Next, we show that the existence of the strong DDH breaker $\overline{\mathcal{A}}$, with the specifications of Lemma 4.4, suffices to construct a verifiable pseudorandom generator under the CDH assumption.

Lemma 4.6 (a VPRG from subexponential CDH)**.** *Let $0 < c < 1$ be a constant. Assume $\overline{\mathcal{A}}$ is a Turing machine running in time $2^{\lambda^c}\cdot\mathrm{poly}(\lambda)$, and an $E \subseteq \mathbb{N}$ is an infinite set such that $\overline{\mathcal{A}}$ is a strong DDH breaker with respect to E (Lemma 4.4), and let $E' = \{2^{\lambda^c}\}_{\lambda\in E}$.*

Then, assuming the subexponential $2^{\lambda^{c'}}$-hardness of CDH for any constant $c < c' < 1$, there exists a subexponential, infinitely-often statistically binding VPRG *with respect to E'.*

Furthermore, if $\overline{\mathcal{A}}$ runs in polynomial time, then assuming the polynomial hardness of CDH, there exists an infinitely-often statistically binding VPRG *with respect to E.*

Let $B : \mathbb{G}^3 \mapsto \{0,1\}$ be a predicate satisfying the following property: given (g^a, g^b, g^c), computing $B(g^a, g^{ab}, g^{ac})$ should be as hard (up to polynomial factors) as computing (g^a, g^{ab}, g^{ac}). Note that this implies that distinguishing $B(g^a, g^{ab}, g^{ac})$ from a random bit given a random triple (g^a, g^b, g^c) is as hard as solving CDH. There are standard method to build this predicate using e.g. the Goldreich-Levin construction [GL89], see e.g. [CKS08] for an illustration in the specific case of CDH. Our construction proceeds as follows.

Let $\lambda' = \lambda$ if $\overline{\mathcal{A}}$ runs in polynomial time, and $\lambda' = \lfloor \log^{1/c}(\lambda)\rfloor$ if $\overline{\mathcal{A}}$ runs in time 2^{λ^c} for some constant $0 < c < 1$. [12]

- $\mathsf{Setup}(1^\lambda, m)$: Sample $\mathcal{G} = \mathcal{G}_{\lambda'} = \mathsf{DHGen}(1^{\lambda'})$ and $g \xleftarrow{\$} \mathbb{G}$. For $i = 1$ to m, pick $a_i \xleftarrow{\$} \mathbb{Z}_p$ and set $h_i \leftarrow g^{a_i}$. Pick a random tape R for $\overline{\mathcal{A}}$. Set $\mathsf{pp} = (1^\lambda, \mathcal{G}, g, (h_i)_{i\leq m}, R)$.
- $\mathsf{Stretch}(\mathsf{pp})$: pick $r \xleftarrow{\$} \mathbb{Z}_p$, set $\mathsf{pvk} \leftarrow g^r$, and for $i \leq m$, set $x_i \xleftarrow{\$} B(\mathsf{pvk}, h_i^r)$. Output $(\mathsf{pvk}, x, \mathsf{aux} = r)$.
- $\mathsf{Prove}(\mathsf{pp}, \mathsf{aux}, i)$: output $\pi \leftarrow h_i^r$.
- $\mathsf{Verify}(\mathsf{pp}, \mathsf{pvk}, \mathcal{T}, i, \sigma, \pi)$: output 1 iff $(B(\mathsf{pvk}, \pi) = \sigma) \wedge (\overline{\mathcal{A}}(1^{\lambda'}, \mathcal{G}, g, \mathsf{pvk}, h'; R) = 1)$.

Theorem 4.7. *If the subexponential CDH assumption holds relative to DHGen, then the above construction is a computationally, subexponentially hiding and statistically binding* VPRG.

Proof. Observe that $\lambda \in E'$ implies $\lambda' \in E$. Completeness on E' and efficiency $\mathrm{poly}(\lambda, m)$ follows easily by inspection, noting that $\overline{\mathcal{A}}$, on input λ', runs in time $\mathrm{poly}(\lambda)$ by assumption. Furthermore, the VPRG can have arbitrary polynomial stretch $m(\lambda)$, independently of the length of pvk (the latter is a single element of $\mathbb{G}$).

We now show that the construction is infinitely-often statistically binding with respect to E'. Let $\lambda \in E'$. Let $\mathcal{B}$ be an adversary against the binding property: on input pp, $\mathcal{B}$ outputs a triple (pvk, i, π). We must exhibit an extractor Ext that finds bit x_i such that $\mathsf{Verify}(\mathsf{pp}, \mathsf{pvk}, i, 1 - x_i, \pi) = 0$ with overwhelming probability. Ext

[12] Recall that in any case, $\overline{\mathcal{A}}$ is a strong DDH breaker.

extracts x_i as follows: it parses pp as $(1^\lambda, \mathcal{G}, g, (h_i)_{i\le m})$, computes $r \leftarrow \mathsf{dlog}_g(\mathsf{pvk})$ and sets $x_i \leftarrow B(\mathsf{pvk}, \pi)$. To make Verify accept, $\mathcal{B}$ must find a triple (pvk, h_i, π) such that $\overline{\mathcal{A}}(1^\lambda, \mathcal{G}, g, \mathsf{pvk}, h_i, \pi; R) = 1$, yet $B(\mathsf{pvk}, \pi) = 1 - x_i$. The latter implies in particular that $(g, \mathsf{pvk}, h_i, \pi) \notin \mathcal{S}_{\mathsf{DH}}$. But with overwhelming probability over the choice of R, there cannot exist an element of $\mathbb{G}^4 \setminus \mathcal{S}_{\mathsf{DH}}$ where $\overline{\mathcal{A}}$ outputs 1, which concludes the proof.

We now discuss the hiding property. We show that a 2^{λ^c}-time adversary $\mathcal{B}$ against the hiding property with advantage greater than $2^{-\lambda^c}$ of the above scheme contradicts the subexponential CDH assumption. Let $\lambda \in E'$. Given a position i, the reduction receives a CDH challenge on security parameter λ', of the form $(1^{\lambda'}, \mathcal{G}, g, g^\alpha, g^\beta)$ and attempts to guess the predicate $x = B(g^\alpha, g^{\alpha\beta})$. It defines $h_i \leftarrow g^\beta$ and samples the rest of pp honestly, picking $a_j \xleftarrow{\$} \mathbb{Z}_p$ and setting $h_j \leftarrow g^{a_j}$ for $j \neq i$. Then, it sets $\mathsf{pvk} \leftarrow g^\alpha$, and computes π_j as $(g^\alpha)^{a}_{j}$ and x_j as $B(\mathsf{pvk}, \pi_j)$ for every $j \neq i$. Observe that the input $(\mathsf{pp}, \mathsf{pvk}, (x_j, \pi_j)_{j\neq i})$ to $\mathcal{B}$ is distributed exactly as in the hiding game. The reduction outputs whatever $\mathcal{B}$ outputs. Observe that $x_i = B(g^\alpha, g^{\alpha\beta})$ by construction, hence the advantage of the reduction in this game is exactly the advantage of $\mathcal{B}$ against the hiding property of the VPRG. Since the reduction runs in time $\mathsf{poly}(\lambda) = \mathsf{poly}(2^{\lambda'^c}) < 2^{\lambda^{c'}}$ for any $c < c' < 1$ for all large enough λ and recovers the hardcore predicate of the CDH challenge with subexponential advantage $2^{-\lambda^c}$, this contradicts the $2^{\lambda^{c'}}$-subexponential CDH assumption. The argument extends directly to the setting where $\lambda' = \lambda$ assuming only the polynomial hardness of CDH. □

□

Combining Lemma 4.4 and Lemma 4.6 with Theorem 3.8 concludes the proof of Lemma 4.1, where the resulting CRS is a uniformly random string thanks to the oblivious samplability of the group. We note that the construction above is not new: the works of [CHK03, QRW19] constructed a NIZK by compiling a NIZK in the HBM under the CDH assumption over pairing-friendly groups. Our construction can be viewed as abstracting out their compiler as a VPRG, and replacing the pairing (which is used solely to check a DDH relation in their construction) by the efficient DDH breaker $\overline{\mathcal{A}}$.

5 A Subexponentially-Often NIZK from Subexponential CDH

In this section we prove our main theorem:

Theorem 5.1. *Assume the subexponential hardness of CDH. Then for any* NP *language $\mathcal{L}$, there exists a non-interactive zero-knowledge proof for $\mathcal{L}$ which is infinitely-often secure in the following sense:*

- *Subexponentially-often uniform soundness: There exists a constant $0 < K < 1$, and an infinite set $E \subseteq \mathbb{N}$, such that the following properties hold:*
  - *(Relative density of E): For all $\lambda \in \mathbb{N}$, $\left[\lambda, 2^{\lambda^K}\right] \cap E \neq \emptyset$.*
  - *(Infinitely-often uniform soundness w.r.t. E): For all uniform PPT adversary $\mathcal{A}$ and all $\lambda_i \in E$:*

$$\Pr\left[\begin{array}{l}\mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^{\lambda_i}), (x,\pi) \xleftarrow{\$} \mathcal{A}(\mathsf{crs}) \ : \\ \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \wedge x \notin \mathcal{L}\end{array}\right] \le \mathsf{negl}(\lambda_i).$$

- *Standard adaptive, computational, multi-theorem zero-knowledge against non-uniform efficient verifiers.*

In particular, defining $E = \{\lambda_i\}_{i\in\mathbb{N}}$ as an increasing sequence (namely $\lambda_i < \lambda_j$ whenever $i < j$), we have that for all $i \in \mathbb{N}$: $\lambda_{i+1} \le 2^{(\lambda_i+1)^K}$. We note that the construction from Theorem 5.1 is fully explicit, and satisfies a standard (namely non-uniform) notion of (adaptive, computational, multi-theorem) zero-knowledge.

Remark 5.2 (Restriction on cryptographic groups)**.** We recall that we consider here cryptographic groups with exponentiation in TC^0, typically including $\mathbb{Z}_q^*$ or its subgroup of quadratic residues (see also Section 3.1). This is a similar restriction to the one made in [JJ21]. We sketch in Section 7 how to extend Theorem 5.1 to families of elliptic curves (without requiring a pairing).

Remark 5.3 (Subexponential security)**.** The proof of Theorem 5.1 can be directly modified to achieve various forms of subexponential soundness and zero-knowledge. Soundness with *subexponential advantage* follows from [JJ21], by relying on an appropriately stronger subexponential hardness of DDH (which in turns requires a stronger CDH assumption). Zero-knowledge against *subexponential time* verifiers follows from relying on an appropriately stronger subexponential hardness of CDH. The constant K (representing the density of "secure" parameters in Theorem 5.1) will increase with the strength of the subexponential soundness claim, and the exact subexponential hardness of CDH needed will both grow with K and the strength of the subexponential zero-knowledge claim.

In Section 5.1, we build a universal DDH breaker. We then present our NIZK construction in Section 5.2. We discuss additional results in Section 5.3.

5.1 A Universal DDH Breaker

In order to prove Theorem 5.1, our main building block is a *universal DDH breaker*. Very imprecisely, the universal breaker (1) efficiently breaks DDH on all security parameters such that *some* efficient breaker exists, and such that (2) DDH holds otherwise. For technical reasons (briefly discussed in Remark 5.7), we split the construction of a universal breaker into two procedures: a *tester* which tests whether some input security parameter is secure or broken (Lemma 5.5), and a universal DDH breaker which breaks DDH with large probability whenever any weak breaker exists (Lemma 5.6).

Notation. Throughout the section, $t = t(\lambda)$, $\varepsilon = \varepsilon(\lambda)$ will denote functions such that $t(\lambda)$ is positive-integer-valued, and $0 < \varepsilon(\lambda) < 1/2$. Define the two following machines:

- $M_{(t)}$: on input $(1^\lambda, x)$, run $M(1^\lambda, x)$ for up to $t(\lambda)$ steps. If M terminates and outputs a bit, define that bit as the output of $M_{(t)}(1^\lambda, x)$; otherwise output a random bit $b \leftarrow \{0,1\}$. Note that by definition, $M_{(t)}$ runs in time at most $t(\lambda)$.
- $\overline{M_{(t)}} = \overline{M_{(t)}}(t, \varepsilon/6)$ is the machine defined in Lemma 4.4 starting with $M_{(t)}$, with functions t and $\varepsilon/6$. In particular, if $\mathsf{Adv}^{\mathsf{DDH}}_{M_{(t)}} \geq \varepsilon/6$, then $\mathsf{Adv}^{\mathsf{DDH}}_{\overline{M_{(t)}}} \geq 1 - \mathsf{negl}(\lambda) \cdot 2^{-4T(\lambda)}$.

Next, we define our sets of secure and broken security parameters.

Definition 5.4 (Secure and Broken DDH parameters)**.** *Let $t = t(\lambda)$, $\varepsilon = \varepsilon(\lambda)$ be functions such that $t(\lambda)$ is positive-integer-valued, and $0 < \varepsilon(\lambda) < 1/2$.*

We define $\mathsf{SECURE} = \mathsf{SECURE}(t, \varepsilon) \subseteq \mathbb{N}$ as the set of security parameters λ such that, for all uniform Turing machines $\mathcal{A}$ of size at most $\lfloor \log \lambda \rfloor$ running in time at most $t(\lambda)$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda) < \varepsilon(\lambda),$$

where $\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda)$ is defined in Definition 3.2.

We define $\mathsf{BROKEN} = \mathsf{BROKEN}(t, \varepsilon) \subseteq \mathbb{N}$ as the set of security parameters λ, such that there exists a uniform machine $\mathcal{A}^*$ of size at most $\lfloor \log \lambda \rfloor$ such that:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}^*_{(t)}}(1^\lambda) \geq \frac{\varepsilon(\lambda)}{2},$$

where $\mathcal{A}^*_{(t)}$ is defined above.

Let us make a few comments on this definition. First, $\mathsf{SECURE} \cup \mathsf{BROKEN} = \mathbb{N}$. This is because if $\lambda \notin \mathsf{SECURE}$, then any machine $\mathcal{A}^*$ contradicting $\lambda \in \mathsf{SECURE}$ is a "witness" for $\lambda \in \mathsf{BROKEN}$. However, SECURE and BROKEN are not necessarily complementary sets. This is because (1) the advantage requirements are potentially compatible, and (2) BROKEN quantifies over a slightly larger set of Turing machines, as $\mathcal{A}^*_{(t)}$ can have description size (slightly) larger than $\lfloor \log \lambda \rfloor$.

The following gives an algorithm which, on input a security parameter, efficiently determines whether DDH is secure or not, in the sense of Definition 5.4.

Lemma 5.5 (Security Parameter Tester)**.** *Let $t = t(\lambda)$, $\varepsilon = \varepsilon(\lambda)$ be functions such that $t(\lambda)$ is positive-integer-valued, and $0 < \varepsilon(\lambda) < 1/2$.*

Then there exists an algorithm $\mathsf{Test} = \mathsf{Test}(t, \varepsilon)$ which takes as input 1^λ where $\lambda \in \mathbb{N}$, runs in time $t(\lambda)/\varepsilon^2(\lambda) \cdot \mathsf{poly}(\lambda)$, and satisfying the following properties:

- *For any $\lambda \in \mathbb{N}$:*
$$\Pr\left[\lambda \notin \mathsf{SECURE} \wedge \mathsf{Test}(1^\lambda) = 1\right] \leq 2^{-\lambda},$$
over the randomness of Test*;*
- *For any $\lambda \in \mathbb{N}$:*
$$\Pr\left[\lambda \notin \mathsf{BROKEN} \wedge \mathsf{Test}(1^\lambda) = 0\right] \leq \lambda \cdot 2^{-\lambda},$$
over the randomness of Test*.*

Intuitively, the algorithm Test can ensure, with overwhelming probability, that some input security parameter is secure (corresponding to output 1) or broken (corresponding to output 0) with respect to Definition 5.4. Note that Test can potentially produce both outcomes with large probability whenever $\lambda \in \mathsf{SECURE} \cap \mathsf{BROKEN}$.

Proof. We define Test as follows. On input (1^λ):

- For $M \in \{0,1\}^{\lfloor \log \lambda \rfloor}$, parse M as the description of a Turing Machine. Let $C(\lambda) = \left\lceil 100 \cdot \frac{\lambda}{\varepsilon^2(\lambda)} \right\rceil$
  - For $i = 1$ to $C(\lambda)$, sample $\alpha_i, \beta_i, \gamma_i \leftarrow \mathbb{Z}_p$, and $b_i \leftarrow \{0,1\}$. Set $\delta_i = b\gamma_i + (1-b)\alpha_i\beta_i$. Compute $b_i' \leftarrow M_{(t)}((1^\lambda, \mathcal{G}, g, g^{\alpha_i}, g^{\beta_i}, g^{\delta_i})$. Let

    $$c_M = \sum_{i=1}^{C(\lambda)} 1 \oplus b_i \oplus b_i'$$

    be the number of indices i such that $b_i = b_i'$.

    If $c_M \geq \left(\frac{1}{2} + \frac{3\varepsilon(\lambda)}{4}\right) \cdot C(\lambda)$, then output 0.

    Otherwise continue to the next $M \in \{0,1\}^{\lfloor \log \lambda \rfloor}$.
- If no output has been produced so far, output 1.

Note that Test runs in time $t(\lambda)/\varepsilon^2(\lambda) \cdot \mathsf{poly}(\lambda)$.
We first prove that, for any $\lambda \in \mathbb{N}$:

$$\Pr\left[\lambda \notin \mathsf{SECURE} \wedge \mathsf{Test}(1^\lambda) = 1\right] \leq 2^{-\lambda},$$

over the randomness of Test.

Suppose $\lambda \notin \mathsf{SECURE}$. Then there exists a uniform adversary $\mathcal{A}^*$ with size at most $\lfloor \log \lambda \rfloor$ running in time $t(\lambda)$ such that:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}^*}(1^\lambda) \geq \varepsilon(\lambda).$$

Because $\mathcal{A}^*$ runs in time $t(\lambda)$, note that $\mathcal{A}^*_{(t)} \equiv \mathcal{A}^*$. In order to output $\perp$, Test has to loop through $M = \mathcal{A}^*$. When $M = \mathcal{A}^*$, all the b_i and b_i' for all $1 \leq i \leq C(\lambda)$ are independent from each other, and $\Pr[b_i = b_i'] \geq \frac{1}{2} + \varepsilon(\lambda)$ by assumption on $\mathcal{A}^*$, so that $\mathbb{E}[c_M] \geq \left(\frac{1}{2} + \varepsilon(\lambda)\right) \cdot C(\lambda)$. A standard Chernoff bound gives:

$$\Pr\left[c_M \leq \left(\frac{1}{2} + \frac{3\varepsilon(\lambda)}{4}\right) \cdot \left\lceil \frac{\lambda}{\varepsilon^2(\lambda)} \right\rceil\right] \leq \exp\left(-\frac{100\lambda}{64}\right) \leq 2^{-\lambda},$$

so with probability at least $1 - \exp(-\lambda/64)$, Test outputs 0 when looping on $\mathcal{A}^*$ (or some other machine, if it produces an output bit before reaching $\mathcal{A}^*$).

Next, we prove that for all $\lambda \in \mathbb{N}$:

$$\Pr\left[\lambda \notin \mathsf{BROKEN} \wedge \mathsf{Test}(1^\lambda) = 0\right] \leq \lambda \cdot 2^{-\lambda}$$

over the randomness of Test.

Suppose $\lambda \notin \mathsf{BROKEN}$, that is, for all Turing machines M of size at most $\lfloor \log \lambda \rfloor$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{M_{(t)}}(1^\lambda) < \frac{\varepsilon(\lambda)}{2}.$$

Then $\mathbb{E}[c_M] \leq \left(\frac{1}{2} + \frac{\varepsilon(\lambda)}{2}\right) \cdot \left\lceil \frac{\lambda}{\varepsilon^2(\lambda)} \right\rceil$. A standard Chernoff bound gives:

$$\Pr\left[c_M \geq \left(\frac{1}{2} + \frac{3\varepsilon(\lambda)}{4}\right) \cdot \left\lceil \frac{\lambda}{\varepsilon^2(\lambda)} \right\rceil\right] \leq \exp\left(-\frac{100\lambda}{20}\right) \leq 2^{-\lambda}.$$

Using a union bound, the probability that Test outputs 1 on any machine M of size at most $\lfloor \log \lambda \rfloor$ is at most $\lambda \cdot 2^{-\lambda}$. □ □

Next, we build a universal breaker that breaks DDH on any $\lambda \in \mathsf{BROKEN}$:

Lemma 5.6 (Universal DDH Breaker)**.** *Let $t = t(\lambda)$ and $\varepsilon = \varepsilon(\lambda)$ be positive functions defined in the beginning of the section.*

Then there exists an algorithm $\mathsf{UnivBreak} = \mathsf{UnivBreak}(t, \varepsilon)$ *which runs in time* $t(\lambda)/\varepsilon^2(\lambda) \cdot \mathsf{poly}(\lambda)$*, such that for all* $\lambda \in \mathsf{BROKEN}$*:*

$$\Pr\left[\begin{array}{l} \mathcal{G} = \mathit{DHGen}(1^\lambda), g \xleftarrow{\$} \mathbb{G}, \alpha, \beta, \gamma \xleftarrow{\$} \mathbb{Z}_p, \\ \quad b \xleftarrow{\$} \{0,1\}, \delta \leftarrow b\gamma + (1-b)\alpha\beta, \quad : \ b = b' \\ b' \xleftarrow{\$} \mathsf{UnivBreak}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\delta) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda) \cdot 2^{-4T(\lambda)}.$$

Proof. We define UnivBreak as follows. On input $(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\delta)$):

- For $M \in \{0,1\}^{\lfloor \log \lambda \rfloor}$, parse M as the description of a Turing Machine. Let $C'(\lambda) = \left\lceil 400 \cdot \frac{T(\lambda)+\lambda}{\varepsilon^2(\lambda)} \right\rceil$
  - For $i = 1$ to $C(\lambda)$, sample $\alpha_i, \beta_i, \gamma_i \leftarrow \mathbb{Z}_p$, and $b_i \leftarrow \{0,1\}$. Set $\delta_i = b\gamma_i + (1-b)\alpha_i\beta_i$. Compute $b_i' \leftarrow M_{(t)}((1^\lambda, \mathcal{G}, g, g^{\alpha_i}, g^{\beta_i}, g^{\delta_i})$. Let

    $$c_M = \sum_{i=1}^{C(\lambda)} 1 \oplus b_i \oplus b_i'$$

    be the number of indices i such that $b_i = b_i'$.

    If $c_M \geq \left(\frac{1}{2} + \frac{\varepsilon(\lambda)}{3}\right) \cdot C(\lambda)$, then output $\overline{M_{(t)}}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\delta)$.

    Otherwise continue to the next $M \in \{0,1\}^{\lfloor \log \lambda \rfloor}$.
- If no output has been produced so far, output a random bit $b \leftarrow \{0,1\}$.

Note that UnivBreak runs in time $t(\lambda)/\varepsilon^2(\lambda) \cdot \mathsf{poly}(\lambda)$.

Let $\lambda \in \mathsf{BROKEN}$, and let M^* be a machine of size at most $\lfloor \log \lambda \rfloor$ such that:

$$\mathsf{Adv}^{\mathsf{DDH}}_{M_{(t)}}(1^\lambda) \geq \frac{\varepsilon(\lambda)}{2}.$$

We first argue that the probability that UnivBreak outputs a random bit (because of skipping all Turing machines of size at most $\lfloor \log \lambda \rfloor$) is at most $2^{-\lambda} \cdot 2^{-4T(\lambda)}$. This only occurs whenever UnivBreak ignores M^*, which happens with probability at most $2^{-\lambda} \cdot 2^{-4T(\lambda)}$ by a standard Chernoff bound similar to Lemma 5.5.

Next, we claim that the probability that UnivBreak produces an output using a machine M such that $\mathsf{Adv}^{\mathsf{DDH}}_{M_{(t)}} < \varepsilon/6$ is at most $\lambda \cdot 2^{-\lambda} \cdot 2^{-4T(\lambda)}$, again using a standard Chernoff bound similar to Lemma 5.5. Finally, by definition of $\overline{M_{(t)}} = \overline{M_{(t)}}(t, \varepsilon/6)$ (defined at the beginning of Section 5.1 and Lemma 4.4), we obtain:

$$\Pr\left[\begin{array}{l} \mathcal{G} = \mathsf{DHGen}(1^\lambda), g \xleftarrow{\$} \mathbb{G}, \alpha, \beta, \gamma \xleftarrow{\$} \mathbb{Z}_p, \\ \quad b \xleftarrow{\$} \{0,1\}, \delta \leftarrow b\gamma + (1-b)\alpha\beta, \quad : \ b = b' \\ b' \xleftarrow{\$} \mathsf{UnivBreak}(1^\lambda, \mathcal{G}, g, g^\alpha, g^\beta, g^\delta) \end{array}\right] \geq 1 - (2^{-\lambda} + \lambda \cdot 2^{-\lambda} + \mathsf{negl}(\lambda)) \cdot 2^{-4T(\lambda)}.$$

$$\geq 1 - \mathsf{negl}(\lambda) \cdot 2^{-4T(\lambda)}.$$

□ □

Remark 5.7 (Splitting tester and universal breaker)**.** We introduced separate constructions of testers and breakers, even though the algorithms are very similar: this is done so that they can use different thresholds. Then, the behavior of UnivBreak becomes fully characterized by whether its input security parameter belongs to BROKEN. Using the same threshold for the tester and the breaker would instead result in a universal breaker which, on input some security parameters in SECURE∩BROKEN, could potentially "fail" with high (say constant) probability and use some good enough breaker with also high constant probability.

Remark 5.8 (Generality of Universal Breakers)**.** The definitions of secure and broken sets (Definition 5.4) and the blueprints of Lemma 5.5 and Lemma 5.6 are quite general, and should readily extend to any falsifiable computational security definition, not just DDH. The only difference is that the resulting breaker analog to Lemma 5.6 would only be guaranteed to have advantage $\varepsilon(\lambda)/6 - (\lambda+1) \cdot 2^{-\lambda}$. In Lemma 5.6, we amplify directly the breaker internally using Lemma 4.4 instead.

5.2 A Subexponentially-Often NIZK

We now prove Theorem 5.1. We first provide an outline of our construction. We use both a DDH-based NIZK, and a VPRG-based NIZK based on the universal breaker of Section 5.1. Given a fixed security parameter λ, a proof consists of both proofs (which does not hurt zero-knowledge, as both constructions are zero-knowledge almost-everywhere), where we use complexity leveraging on the VPRG-based one (namely, we run it on a smaller security parameter λ'). In order to verify a proof, we use our "universal tester" from Section 5.1 to check whether (the complexity leveraged version of) DDH is broken; if it is, then the VPRG-based NIZK, using the universal breaker of Section 5.1, ensures completeness and (statistical) soundness. Note that the testing step, and the verification algorithm of the VPRG-based NIZK, are made efficient thanks to complexity leveraging. Otherwise, we do not know how to directly argue that the DDH-based NIZK provides soundness; instead, we argue that there exists a "relatively close" security parameter $\overline{\lambda}$ for which the DDH-based NIZK allows to argue soundness. The formal construction and proof follow.

Let $\varepsilon = \varepsilon(\lambda) = 2^{-\lambda^c}$ be an inverse subexponential function, where $0 < c < 1$ is a constant and $t = t(\lambda)$ be any superpolynomial, positive-integer-valued function such that $t(\lambda) \leq 2^{\lambda^c}$.[13]

We use the following building blocks:

- A DDH-based NIZK (DDH.Setup, DDH.Prove, DDH.Verify) given by Theorem 3.10 using the constant c.
- A VPRG-based NIZK (VPRG.Setup, VPRG.Prove, VPRG.Verify) from Lemma 4.1, instantiated with the universal breaker UnivBreak from Lemma 5.6.[14]
- A DDH tester Test given by Lemma 5.5.

Construction. Define the following NIZK (Setup, Prove, Verify):

- $\mathsf{Setup}(1^\lambda, 1^{|x|})$: Define $\lambda' = \left\lfloor \log^{1/c}(\lambda) \right\rfloor$. Compute $\mathsf{crs}_{\mathsf{DDH}} \leftarrow \mathsf{DDH.Setup}(1^\lambda, 1^{|x|})$, $\mathsf{crs}_{\mathsf{VPRG}} \leftarrow \mathsf{VPRG.Setup}(1^{\lambda'}, 1^{|x|})$,[15] and output $\mathsf{crs} = (\mathsf{crs}_{\mathsf{DDH}}, \mathsf{crs}_{\mathsf{VPRG}})$.
- $\mathsf{Prove}(\mathsf{crs}, x, w)$: Compute $\pi_{\mathsf{DDH}} \leftarrow \mathsf{DDH.Prove}(\mathsf{crs}_{\mathsf{DDH}}, x, w)$ and $\pi_{\mathsf{VPRG}} \leftarrow \mathsf{VPRG.Prove}(\mathsf{crs}_{\mathsf{VPRG}}, x, w)$. Output $\pi = (\pi_{DDH}, \pi_{\mathsf{VPRG}})$.
- $\mathsf{Verify}(\mathsf{crs}, x, \pi)$: Compute $b \leftarrow \mathsf{Test}(1^{\lambda'})$. If $b = 0$, output $\mathsf{VPRG.Verify}(\mathsf{crs}_{\mathsf{VPRG}}, x, \pi_{\mathsf{VPRG}})$ using UnivBreak. If $b = 1$, output $\mathsf{DDH.Verify}(\mathsf{crs}_{\mathsf{DDH}}, x, \pi_{\mathsf{DDH}})$.

We first tie the definitions of Definition 5.4 with security properties of the NIZKs above. Recall that for any $\lambda \in \mathbb{N}$, we defined in Theorem 3.10 the set $\mathsf{TOWER}_\lambda = \mathsf{TOWER}_\lambda(c, L) := \{\lambda\} \cup \{\lambda^{(c/2)^{i/2}}\}_{i \in [L]}$, where $L > 0$ is a constant given by Theorem 3.10.

Lemma 5.9 (Security of the DDH-based NIZK)**.** *Define the set*

$$E_{DDH} := \bigcup_{\lambda \in \mathbb{N}} \{\lambda \mid \mathsf{TOWER}_\lambda \subseteq \mathsf{SECURE}\},$$

where $\mathsf{SECURE} = \mathsf{SECURE}(t, \varepsilon)$ *is defined in Definition 5.4. Then* (*DDH*.Setup, *DDH*.Prove, *DDH*.Verify) *satisfies perfect completeness, infinitely-often soundness w.r.t.* E_{DDH} *(against uniform cheating provers), and statistical zero-knowledge against non-uniform verifiers.*

Proof. First, observe that the construction of E_{DDH} implies:

$$\bigcup_{\lambda \in E_{\mathsf{DDH}}} \mathsf{TOWER}_\lambda \subseteq \mathsf{SECURE}.$$

In particular, for any $\lambda \in E_{\mathsf{DDH}}$ and any $\lambda_i \in \mathsf{TOWER}_\lambda$, $\lambda_i \in \mathsf{SECURE}$.

[13] Taking any other subexponential upper-bound for t would suffice for us, but would result in additional unnecessary notation.

[14] The universal breaker from Lemma 5.6 is already a strong breaker, so the proof of Lemma 4.1 can directly argued combining Lemma 4.6 with Theorem 3.8, without explicitly using Lemma 4.4. This is because we internally amplified the success probability of UnivBreak in Lemma 5.6 (using Lemma 4.4).

[15] We use the VPRG-based NIZK to prove statements of size $|x| = \mathsf{poly}(\lambda)$ which are subexponential in its internal security parameter λ'. The VPRG-based NIZK of Lemma 4.1 remains subexponentially secure against adversaries (namely, verifiers, as soundness is statistical) running in time $\mathsf{poly}(|x|)$ in that setting. See Remarks 3.9 and 4.3.

Therefore, by Theorem 3.10, it suffices to check that, for any uniform PPT adversary $\mathcal{A}$, there exists λ^* such that for all $\lambda \in \mathsf{SECURE}$ such that $\lambda \geq \lambda^*$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda) \leq 2^{-\lambda^c}. \tag{1}$$

Let $\mathcal{A}$ be a uniform adversary, and let $q(\lambda) = \mathsf{poly}(\lambda)$ denote its runtime, and s its size as a Turing machine. By construction of SECURE (Definition 5.4), Equation (1) holds for all $\lambda \in \mathsf{SECURE}$ such that $t(\lambda) \geq q(\lambda)$ and $\lambda \geq 2^s$, which in turn hold for all large enough $\lambda \in \mathsf{SECURE}$ as t is a super-polynomial function. □ □

Lemma 5.10 (Security of the VPRG-based NIZK). *Define, for all $\lambda \in \mathbb{N}$, $\mathsf{VPRG}.\overline{\mathsf{Setup}}(1^\lambda) := \mathsf{VPRG}.\mathsf{Setup}(1^{\lambda'})$, where $\lambda' = \left\lfloor \log^{1/c}(\lambda) \right\rfloor$. Define the set*

$$E_{\mathsf{VPRG}} = \left\{ \lambda \;\middle|\; \left\lfloor \log^{1/c}(\lambda) \right\rfloor \in \mathsf{BROKEN} \right\}.$$

Let c' be any constant such that $c < c' < 1$. Assuming the $2^{\lambda^{c'}}$-subexponential hardness of CDH (Section 3.1), $(\mathsf{VPRG}.\overline{\mathsf{Setup}}, \mathsf{VPRG}.\mathsf{Prove}, \mathsf{VPRG}.\mathsf{Verify})$ is infinitely often correct and statistically adaptively sound with respect to E_{VPRG}[16], and computationally adaptively, multi-theorem zero-knowledge.

Proof. By definition of λ', and by assumption on the functions t, ε, $\mathsf{UnivBreak}$ on input $(1^{\lambda'}, x)$ for any x runs in time $t(\lambda')/\varepsilon^2(\lambda') \cdot \mathsf{poly}(\lambda') = \mathsf{poly}(\lambda)$, and therefore the algorithms $(\mathsf{VPRG}.\overline{\mathsf{Setup}}, \mathsf{VPRG}.\mathsf{Prove}, \mathsf{VPRG}.\mathsf{Verify})$ run in polynomial time.

The rest follows by instantiating Lemma 4.1 starting with $\mathcal{A} = \mathsf{UnivBreak}$, which runs on security parameter $\lambda' \in \mathsf{BROKEN}$ by definition of E_{VPRG}, and using that $\mathsf{Adv}^{\mathsf{DDH}}_{\mathsf{UnivBreak}}(1^{\lambda'}) \geq 1 - \mathsf{negl}(\lambda') \cdot 2^{-4T(\lambda')}$ thanks to Lemma 5.6. □ □

Next, we tie Lemma 5.9 and Lemma 5.10 together thanks to the properties of our tester Test. Let $\lambda \in \mathbb{N}$. We distinguish several cases:

Case 1: If $\lambda \notin E_{\mathsf{DDH}}$, there exists some $\lambda_i \in \mathsf{TOWER}_\lambda$ such that $\lambda_i \notin \mathsf{SECURE}$, so that $\lambda_i \in \mathsf{BROKEN}$ and $\overline{\lambda} := 2^{\lambda_i^c} \in E_{\mathsf{VPRG}}$ by definition. Furthermore, because $\lambda_i \notin \mathsf{SECURE}$, $\mathsf{Test}(1^{\lambda_i})$ outputs 0 except with probability $2^{-\lambda_i}$ by Lemma 5.5, and therefore $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ on security parameter $\overline{\lambda} = 2^{\lambda_i^c}$ will call $\mathsf{VPRG}.\mathsf{Verify}$ with overwhelming probability. Soundness on $\overline{\lambda}$ then follows by soundness of the VPRG-based NIZK $(\mathsf{VPRG}.\overline{\mathsf{Setup}}, \mathsf{VPRG}.\mathsf{Prove}, \mathsf{VPRG}.\mathsf{Verify})$ on E_{VPRG} (Lemma 5.10).

Case 2: If $\lambda \in E_{\mathsf{DDH}}$, namely if $\mathsf{TOWER}_\lambda \subseteq \mathsf{SECURE}$, we distinguish two subcases:

- **Case 2.1:** $\lambda' \notin \mathsf{BROKEN}$ then $\mathsf{Test}(1^{\lambda'})$ outputs 1 except with probability at most $\lambda \cdot 2^{-\lambda}$ by Lemma 5.5, and therefore $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ on security parameter λ will call $\mathsf{DDH}.\mathsf{Verify}$ with overwhelming probability. Soundness on λ then follows by soundness of the DDH-based NIZK $(\mathsf{DDH}.\mathsf{Setup}, \mathsf{DDH}.\mathsf{Prove}, \mathsf{DDH}.\mathsf{Verify})$ on E_{DDH} (Lemma 5.9).
- **Case 2.2:** Last, if $\lambda' \in \mathsf{BROKEN}$, then $\lambda \in E_{\mathsf{DDH}} \cap E_{\mathsf{VPRG}}$, and therefore $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ is sound regardless of the outcome of $\mathsf{Test}(1^{\lambda'})$ by soundness of both NIZKs $(\mathsf{VPRG}.\overline{\mathsf{Setup}}, \mathsf{VPRG}.\mathsf{Prove}, \mathsf{VPRG}.\mathsf{Verify})$ and $(\mathsf{DDH}.\mathsf{Setup}, \mathsf{DDH}.\mathsf{Prove}, \mathsf{DDH}.\mathsf{Verify})$.

Summing up, define a set $E \subseteq \mathbb{N}$ as follows. For all $\lambda \in \mathbb{N}$, define $\lambda \in E$ if either Case 2.1 or Case 2.2 occurs, and define $\overline{\lambda} = 2^{\lambda_i^c} \in E$ if Case 1 or Case 2.2 occurs, for all $\lambda_i \in \mathsf{TOWER}_\lambda$ such that $\lambda_i \in \mathsf{BROKEN}$. We obtain that $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ is infinitely-often adaptively sound with respect to E, and satisfies adaptive, multi-theorem zero-knowledge against non-uniform verifiers. Finally, by construction of E, for all $\lambda \in \mathbb{N}$, $E \cap \left(\{\lambda\} \cup 2^{\mathsf{TOWER}_{\lambda^c}}\right) \neq \emptyset$, and therefore, by construction of TOWER_λ, $E \cap [\lambda, 2^{\lambda^c}] \neq \emptyset$. Setting $K = c$, we obtain the relative density of E as stated in Theorem 5.1. This concludes the proof. □

5.3 Additional Results

Modifying specific building blocks directly yields the following theorems.

Theorem 5.11 (NIZKs from CDH and LPN). *Assume the superpolynomial hardness of CDH and the polynomial hardness of LPN.*[17] *Then for any* NP *language $\mathcal{L}$, there exists a superpolynomially-often uniform non-interactive zero-knowledge proof for $\mathcal{L}$.*

[16] See the paragraph on infinitely-often security in Section 3.2 for a definition of soudness w.r.t. an infinite set E.

[17] The noise rate of the LPN assumption is $1/n^c$ for some constant $1/2 < c < 1$. We refer to [BKM20] for a definition of LPN.

Note that the construction above can be instantiated in *any* (candidate) prime-order groups (that is, without restrictions similar to [JJ21]).

Sketch of proof. This follows from the same construction as in Section 5.2, but instantiating the DDH-based NIZK with the construction of [BKM20] which is secure under the polynomial hardness of both DDH and LPN. We then set t (resp. ε) as any superpolynomial (resp. inverse superpolynomial) function. The only notable differences in the proof are (1) the complexity leveraging is consequently milder, and we therefore only need to rely on the superpolynomial hardness of CDH, and (2) $E_{\mathsf{DDH}} = \mathsf{SECURE}$. □

Theorem 5.12 (ZAP Arguments from Subexponential CDH)**.** *Assume the subexponential hardness of CDH. Then for any* NP *language $\mathscr{L}$, there exists a ZAP argument for $\mathscr{L}$ satisfying (1) subexponentially-often, non-adaptive soundness against uniform efficient cheating provers and (2) (standard) adaptive witness indistinguishability.*

Sketch of proof. We start with the existence of DDH-based ZAPs which is secure assuming DDH (Theorem 3.10; note that it only satisfies non-adaptive soundness), and VPRG-ZAPs built on any DDH breaker, which are secure assuming CDH (this follows noting that Lemma 4.1 gives a statistically sound NIZK with a common random string, which yields a ZAP by [DN00]). The construction simply defines the first (resp. second) message of the ZAP as the concatenation of the first (resp. second) messages two ZAPs. Verification proceeds as in Section 5.2. The analysis is identical to the one in Section 5.2. □

6 An Infinitely-Often NIZK from CDH+LPN

In this section, we prove the following theorem:

Theorem 6.1 (io-NIZK from CDH+LPN)**.** *Assume that CDH and (uniform) LPN both hold.*[18] *Then for any* NP *language $\mathscr{L}$, there exists an infinitely-often uniform non-interactive zero-knowledge proof for $\mathscr{L}$.*

Compared to Theorem 5.11, Theorem 6.1 only requires the *polynomial* hardness of CDH and LPN. This is, however, at the cost of having a non-constructive result, and losing superpolynomial-oftenness. In fact, we prove the following statement:

Theorem 6.2. *At least one of the following statements is necessarily true:*

- *the (uniform) LPN assumption implies the existence of an infinitely-often non-interactive zero-knowledge argument system for* NP *with uniform adaptive soundness and standard zero-knowledge, or*
- *the CDH assumption implies the existence of a non-interactive zero-knowledge proof for* NP *with statistical adaptive soundness, and adaptive multi-theorem zero-knowledge.*

In order to prove Theorems 6.1 and 6.2, consider the following hypothesis H:

For all uniform PPT adversary $\mathcal{A}$, all polynomials q, and for infinitely many security parameters $\lambda \in \mathbb{N}$,

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda) \leq 1/q(\lambda).$$

A note on the choice of H. Hypothesis H is carefully defined so that the resulting NIZK is uniform. Defining H over *infinitely-many* parameters (as opposed to almost all parameters) is crucial: if we started with a standard DDH assumption, its negation would only give us an infinitely-often DDH breaker. Because one cannot (uniformly) test whether this breaker is successful on a given security parameter (attempting to run it until it works leads to halting problems), completeness would be lost. Similarly, if we made the case disjunction over the standard non-uniform hardness of DDH (*i.e.* replacing "uniform" by "non-uniform" in H), the resulting NIZK would again have a non-uniform verifier, as building on a non-uniform DDH breaker.

Theorems 6.1 and 6.2 follow directly from the combination of Lemma 6.3 and Lemma 6.4 below, which show the existence of NIZKs when H holds and when $\neg H$ holds, respectively.

Lemma 6.3. *If H holds, then assuming the (uniform) hardness of LPN, there exists an infinitely-often non-interactive zero-knowledge argument system with uniform non-adaptive soundness and statistical zero-knowledge.*

[18] Again, the noise rate of the LPN assumption is $1/n^c$ for some constant $1/2 < c < 1$, and we refer to [BKM20] for a definition of LPN. We use uniform LPN to denote the setting where adversaries are restricted to be uniform algorithms.

Proof. This is directly implied by the NIZK of [BKM20] which is statistically zero-knowledge, and (uniformly, non-adaptively) sound assuming the polynomial (uniform) hardness of LPN and DDH; their claim extends directly to the infinitely-often, uniform setting. □

Lemma 6.4. *If H does not hold, then assuming the hardness of CDH, there exists a non-interactive zero-knowledge proof with statistical adaptive soundness, and adaptive multi-theorem zero-knowledge.*

Proof. Assuming $\neg H$, there exists a uniform PPT $\mathcal{A}$, along with a polynomial q, such that for all $\lambda \in \mathbb{N}$, $\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda) > 1/q(\lambda)$. The lemma then follows directly from Lemma 4.1. □

7 Instantiation from Elliptic Curves

We sketch here how to adapt our construction to be compatible with using elliptic curves as base cryptographic groups, without requiring the existence of a pairing. Namely, we show that Theorems 5.1 and 5.12 hold for (families of) elliptic curves, thus obtaining NIZKs and ZAP arguments (with the same caveats as Theorems 5.1 and 5.12) assuming the subexponential hardness of CDH over elliptic curves (without requiring the existence of a pairing).

Theorem 7.1. *Under the subexponential CDH assumption over any (family of) elliptic curves, there exists:*

- *a subexponentially-often secure, uniform NIZK argument for all* NP *in the common random string model. The construction satisfies (1) subexponentially-often adaptive, computational soundness against uniform efficient provers, and (2) standard adaptive, computational multi-theorem zero-knowledge against non-uniform verifiers;*
- *a subexponentially-often secure, uniform ZAP argument for all* NP*. The construction satisfies (1) subexponentially-often adaptive, computational soundness against uniform efficient provers and (2) standard adaptive, computational witness-indistinguishability against non-uniform verifiers.*

However, even if Theorem 5.1 and Theorem 5.12 hold over elliptic curves, the resulting constructions ensure slightly weaker guarantees. Indeed, while Theorem 5.1 and Theorem 5.12 can easily be extended to ensure subexponential security for almost free (Remark 5.3), their elliptic curve counterparts only ensure *super-polynomial* soundness, regardless of the strength of the subexponential CDH assumption we start from (Remark 7.4).

The technical difficulty of extending our result to other groups with exponentiation not known to be in TC^0 is inherited from existing NIZKs from DDH [JJ21].[19] Still, [JJ21] shows how to extend their result to elliptic curves using complexity leveraging: while elliptic curves are not known to have exponentiation in TC^0, curves associated to a sufficiently small security parameter have exponentiation in a low enough complexity class to enable the construction of [JJ21]. In our case, we have to delicately argue that this additional use of complexity leveraging can be made compatible with our (already complexity-leveraging-heavy) approach.

We start with the analogue of Theorem 3.10, ported to the setting of elliptic curves (rephrased from [JJ21, Appendix B]).

Theorem 7.2 (NIZK and ZAP Arguments from DDH over Elliptic Curves [JJ21])**.** *There exists a constant $L > 0$ such that the following holds. For any constant $0 < c < 1$ and $M > 1$, and for all $\lambda \in \mathbb{N}$, define the set $\overline{\mathsf{TOWER}}_\lambda = \overline{\mathsf{TOWER}}_\lambda(c, L) := \{\lambda^{(c/2)^{i/2}}\}_{i\in[L]}$. For any infinite set $E \subseteq \mathbb{N}$, define the sets:*

$$E_{\overline{\mathsf{TOWER}}} = \bigcup_{\lambda\in E} \overline{\mathsf{TOWER}}_\lambda,$$
$$E_{\mathsf{com}} = \bigcup_{\lambda\in E} \{\lfloor \log^{M/c} \lambda \rfloor\}$$

Suppose that, for any (uniform) PPT adversary $\mathcal{A}$, there exists λ^* such that for all $\lambda_{\overline{\mathsf{TOWER}}} \in E_{\overline{\mathsf{TOWER}}}$ satisfying $\lambda_{\overline{\mathsf{TOWER}}} \geq \lambda^*$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^{\lambda_{\overline{\mathsf{TOWER}}}}) \leq 2^{-(\lambda_{\overline{\mathsf{TOWER}}})^c}.$$

Furthermore, for all $\lambda_{\mathsf{com}} \in E_{\mathsf{com}}$, define $\lambda = 2^{\lambda_{\mathsf{com}}{}^{c/M}}$. Suppose that, for all (uniform) adversary $\mathcal{A}$ running in time $\mathsf{poly}(\lambda)$, and for all $\lambda_{\mathsf{com}} \in E_{\mathsf{com}}$ satisfying $\lambda_{\mathsf{com}} \geq \lambda^*$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^{\lambda_{\mathsf{com}}}) \leq \mathsf{negl}(\lambda).$$

Then:

[19] For readers familiar with [JJ21], this is so that trapdoor decryption for the sigma-protocol commitment can be evaluated by the interactive trapdoor hashing protocol.

- *there exists a NIZK for all* NP *satisfying perfect completeness, infinitely-often adaptive soundness w.r.t. E (against uniform cheating provers), and computational zero-knowledge against non-uniform verifiers;*
- *there exists a ZAP argument for all* NP *satisfying perfect completeness, infinitely-often non-adaptive soundness w.r.t. E, and statistical adaptive witness indistinguishability.*

In other words, the construction of NIZKs from DDH on elliptic curves from [JJ21] uses, on input security parameter λ, all groups corresponding to security parameters in $\overline{\mathsf{TOWER}}_\lambda$, which are all assumed to be subexponentially secure, and a group with security parameter λ_{com}, which is assumed to be mildly subexponentially secure.[20]

For readers familiar with the construction of [JJ21], $\overline{\mathsf{TOWER}}_\lambda$ corresponds to the set of security parameters used, in the NIZK construction from DDH for security parameter λ, to instantiate the interactive trapdoor hashing / correlation-intractable hash function. In the construction from $\mathbb{Z}_q^*$ (or more generally, in any cryptographic group with exponentiation in TC^0), the commitment for the trapdoor sigma-protocol is instantiated with security parameter λ. In the construction from elliptic curves, further complexity leveraging is required, and this latter commitment is instead instantiated using some poly-logarithmic security parameter $\log^{M/c}\lambda$ for some constant $M > 1$.

We now argue that our construction and proof in Section 5.2 can be adapted to use Theorem 7.2 instead of Theorem 3.10. Similar changes directly extend to the setting of ZAP arguments, extending Theorem 5.12 to elliptic curves.

The construction is almost identical to the one in Section 5.2, except with the following differences:

- We use as our DDH-based NIZK (DDH.Setup, DDH.Prove, DDH.Verify) the one given by Theorem 7.2 for some arbitrary constant $M > 1$, as opposed to Theorem 3.10.
- We require $t = t(\lambda)$ to be a subexponential function. For convenience of notation, we will set $t(\lambda) = 1/\varepsilon(\lambda) = 2^{\lambda^c}$, where we recall that $0 < c < 1$ is a constant.

Therefore, in terms of correctness and security, the only difference with Section 5.2 appears in Lemma 5.9, which is replaced by the following.

Lemma 7.3 (Security of the DDH-based NIZK — Elliptic Curve Version)**.** *Define the set*

$$\overline{E}_{DDH} := \bigcup_{\lambda\in\mathbb{N}} \left\{\lambda \;\middle|\; \overline{\mathsf{TOWER}}_\lambda \cup \{\lfloor\log^{M/c}\lambda\rfloor\} \subseteq \mathsf{SECURE}\right\},$$

where $\mathsf{SECURE} = \mathsf{SECURE}(t,\varepsilon)$ *is defined in Definition 5.4, and* $\overline{\mathsf{TOWER}}_\lambda$ *is defined in Theorem 7.2. Then* (*DDH*.Setup, *DDH*.Prove, *DDH*.Verify) *satisfies perfect completeness, infinitely-often soundness w.r.t.* $\overline{E}_{DDH}$ *(against uniform cheating provers), and statistical zero-knowledge against non-uniform verifiers.*

Proof. First, observe that the construction of $\overline{E}_{\mathsf{DDH}}$ implies:

$$\bigcup_{\lambda\in\overline{E}_{\mathsf{DDH}}} \overline{\mathsf{TOWER}}_\lambda \cup \{\lfloor\log^{M/c}\lambda\rfloor\} \subseteq \mathsf{SECURE}.$$

In particular, for any $\lambda \in \overline{E}_{\mathsf{DDH}}$ and any $\lambda_i \in \overline{\mathsf{TOWER}}_\lambda \cup \{\lfloor\log^{M/c}\lambda\rfloor\}$, $\lambda_i \in \mathsf{SECURE}$.

Therefore, by Theorem 7.2, it suffices to check that that the two following statements hold. First, that, for any uniform PPT adversary $\mathcal{A}$, there exists λ^* such that for all $\lambda \in \mathsf{SECURE}$ such that $\lambda \geq \lambda^*$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^\lambda) \leq 2^{-\lambda^c}. \tag{2}$$

Second, define the function $\tau(\lambda') := 2^{\lambda'^{(c/M)}}$. We also check that for all uniform adversaries $\mathcal{A}$ running in time $\mathsf{poly}(\tau)$:

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A}}(1^{\lambda'}) \leq \mathsf{negl}(\tau(\lambda')). \tag{3}$$

Note that, renaming $\lambda' = \lfloor\log^{M/c}\lambda\rfloor$, Equation (3) requires the advantage of $\mathsf{poly}(\lambda)$ adversaries against DDH with security parameter λ' to be $\mathsf{negl}(\lambda)$.

Let $\mathcal{A}$ be a uniform adversary, and let $q(\lambda)$ denote its runtime, and s its size as a Turing machine. By construction of SECURE (Definition 5.4), if q is a polynomial $\mathsf{poly}(\lambda)$, then Equation (2) holds for all λ such that

[20] Technically, the groups in $\overline{\mathsf{TOWER}}_\lambda$ are only required to be subexponentially-secure with respect to polynomial-time adversaries, while the group for λ_{com} is required to be secure with respect to mildly-subexponential time $\lambda = 2^{\lambda_{\mathsf{com}}{}^{c/M}}$ adversaries. We will ignore this distinction, as it will not affect our final result.

$t(\lambda) \geq q(\lambda)$ and $\lambda \geq 2^s$, which in turn hold for all large enough λ (as t is a super-polynomial function and q is assumed to be polynomial).

Similarly, if $q(\lambda)$ is polynomial in $\tau(\lambda) = 2^{\lambda^{(c/M)}}$, then for large enough λ, we have $t(\lambda) = 2^{\lambda^c} \geq \mathsf{poly}(2^{\lambda^{(c/M)}}) = q(\lambda)$ and $\varepsilon(\lambda) = 2^{\lambda^{-c}} = \mathsf{negl}(q(\lambda))$, as $M > 1$. □ □

Similarly to Section 5, we define a set $\overline{E}$ as follows. Let $\lambda \in \mathbb{N}$. Then either $\lambda \in \overline{E}_{\mathsf{DDH}}$, in which case we define $\lambda \in \overline{E}$, or $\lambda \notin \overline{E}_{\mathsf{DDH}}$, in which case we define $2^{\lambda_i^c} \in \overline{E}$ for all $\lambda_i \in \overline{\mathsf{TOWER}}_\lambda$ such that $\lambda_i \in \mathsf{BROKEN}$. A similar analysis as in Section 5 shows that our construction is secure for all large enough $\lambda \in \overline{E}$. Furthermore, by construction, $\overline{E}$ always contains an element in $\{\lambda\} \cup \{2^{\overline{\mathsf{TOWER}}_\lambda{}^c}\} \subset [\lambda, 2^{(\lambda^{c/2})^c}] \in E$, and setting $K = c^2/2$ concludes the proof.

Remark 7.4 (Superpolynomial Security)**.** Similar to Remark 5.3, the proof above can be directly adapted to achieve stronger than polynomial security. Soundness with *superpolynomial advantage* follows from [JJ21], by relying on an appropriately stronger subexponential hardness of DDH (which in turns requires a stronger CDH assumption). Note that because of the additional layer of complexity leveraging in the elliptic curve setting, we can only ensure super-polynomial hardness for Equation (3) even if we assume strong subexponential security of DDH, and we are therefore stuck to super-polynomial security. This is one main difference with the setting of groups with exponentiation in TC^0.

Zero-knowledge against *subexponential-time* verifiers again follows from relying on an appropriately stronger subexponential hardness of CDH. The constant K (representing the density of "secure" parameters) will increase with the strength of the superpolynomial soundness claim, and the exact subexponential hardness of CDH needed will both grow with K and the strength of the subexponential zero-knowledge claim.

8 On Promise-True Distributional Search NP Hardness from Average-Case NP Hardness

We show here that our new technique for handling disjunction arguments seem flexible, and seem to apply to other similar disjunction arguments that previously appeared in the litterature. We leave a more precise characterization of compatible prior work, along with an associated abstraction of our technique, for future work.

We focus here on the work of [PV20], which shows that, using their terminology, proving theorems that are guaranteed to be true is no easier than proving theorems in general. More formally, [PV20] prove the following result (we refer to Section 8.2 for definitions):

Theorem 8.1. *Suppose that there exists a distributional* NP *problem that is (almost-everywhere) hard on average. Then, there exists an infinitely-often hard-on-average promise-true distributional* NP *search problem.*

In contrast, we use our techniques to obtain the following theorem:

Theorem 8.2. *Assume the existence of a superpolynomially-secure uniformly hard-on-average distributional* NP *problem* $(\mathscr{L}, \mathcal{D})$*. Then there is an explicit construction of a promise-true distributional* NP *search problem which is uniformly superpolynomially-often hard-on-the-average.*

Compared with Theorem 8.1, Theorem 8.2 strengthens the security requirement on the language, which is now required to be *superpolynomially* hard on average (where the superpolynomial bound can be arbitrarily close to polynomial). In exchange for this stronger requirement, it obtains a much stronger conclusion: that there exists a promise-true distributional NP search problem that is secure on a set of superpolynomial density (as opposed to just secure on infinitely-many security parameters). Our construction is also in the uniform setting (hence we start from the weaker notion of uniformly secure HOA languages, and obtain the weaker notion of uniformly secure promise-true distributional NP search problem).

8.1 A Universal One-Way Function Tester

Definition 8.3. *Let $f : \{0,1\}^* \mapsto \{0,1\}^*$ be a polynomial-time computable function. f is said to be a (t, ε)-one-way function (OWF) if for every $t(\lambda)$-time algorithm $\mathcal{A}$, for all large enough $\lambda \in \mathbb{N}$, it holds that*

$$Pr[x \xleftarrow{\$} \{0,1\}^\lambda, y = f(x) : \mathcal{A}(1^\lambda, y) \in f^{-1}(f(x))] = \mathsf{Adv}^f_{\mathcal{A}}(1^\lambda) \leq \varepsilon(\lambda).$$

If the above holds only for infinitely many $\lambda \in \mathbb{N}$, we say that f is an infinitely-often (t, ε)*-OWF ((t, ε)-ioOWF).*

Now, fix a superpolynomial function $t(\lambda) = \lambda^{\omega(1)}$. As in Definition 5.4, for any candidate one-way function family $f = \{f_\lambda : \{0,1\}^\lambda \mapsto \{0,1\}^\lambda\}$, we can define $\mathsf{SECURE}_f = \mathsf{SECURE}_f(t, 1/t) \subseteq \mathbb{N}$ as the set of security parameters λ such that, for all uniform Turing machines $\mathcal{A}$ of size at most $\lfloor \log \lambda \rfloor$ running in time at most $t(\lambda)$:

$$\mathsf{Adv}^f_{\mathcal{A}}(1^\lambda) < 1/t(\lambda).$$

Similarly, we define $\mathsf{BROKEN}_f = \mathsf{BROKEN}_f(t, 1/t) \subseteq \mathbb{N}$ as the set of security parameters λ, such that there exists a uniform machine $\mathcal{A}$ of size at most $\lfloor \log \lambda \rfloor$ such that:

$$\mathsf{Adv}^f_{\mathcal{A}_{(t)}}(1^\lambda) \geq 1/2t$$

(recall that $\mathcal{A}_{(t)}$ is the machine that runs $\mathcal{A}$ for exactly t steps, outputs whatever outputs if $\mathcal{A}$ terminates in t steps, and outputs a random bit otherwise). The same considerations as in the case of DDH holds for these sets: $\mathbb{N} = \mathsf{SECURE}_f \cup \mathsf{BROKEN}_f$, but these do not form a partition of the integers (they can overlap).

Testing for all one-way functions with bounded runtime. We will go one step further, as we want our tester to identify, for any input parameter λ, whether there *exists* any polynomial-time computable function f that is $(t, 1/t)$-one way. To do so, we now fix two parameters, $t'(\lambda) = \lambda^{\omega(1)}$ and $t(\lambda) = (t'(\lambda))^{\omega(1)}$. Note that now, t' is superpolynomial in λ, while t is superpolynomial in t'. Then, for any adversary $\mathcal{A}$, we denote by $\mathsf{Adv}^{t'}_{\mathcal{A}}(1^\lambda)$ the largest advantage of $\mathcal{A}$ against any candidate $(t, 1/t)$-OWF f which can be computed by a uniform Turing machine of size at most $\lfloor \log \lambda \rfloor$ running in time at most $t'(\lambda)$ (note that this includes in particular all candidate OWF with description size at most $\lfloor \log \lambda \rfloor$ running in time $\mathsf{poly}(\lambda)$ for large enough λ). That is,

$$\mathsf{Adv}^{t'}_{\mathcal{A}}(1^\lambda) = \max_f \mathsf{Adv}^f_{\mathcal{A}}(1^\lambda),$$

where the maximum is taken over all uniform Turing machines f of size at most $\lfloor \log \lambda \rfloor$ running in time at most $t'(\lambda)$. Then, we define

$$\mathsf{SECURE} = \mathsf{SECURE}(t, t') = \bigcup_f \mathsf{SECURE}_f(t, 1/t) \subseteq \mathbb{N}$$

as the set of security parameters λ such that there *exists* a function f with description size at most $\lfloor \log \lambda \rfloor$ running in time at most $t'(\lambda)$ such that, for all uniform Turing machines $\mathcal{A}$ of size at most $\lfloor \log \lambda \rfloor$ running in time at most $t(\lambda)$:

$$\mathsf{Adv}^f_{\mathcal{A}}(1^\lambda) < 1/t(\lambda).$$

Similarly, we define $\mathsf{BROKEN} = \mathsf{BROKEN}(t, t') = \bigcap_f \mathsf{BROKEN}_f(t, 1/t) \subseteq \mathbb{N}$ as the set of security parameters λ, such that *for all* functions f with description size at most $\lfloor \log \lambda \rfloor$ running in time at most $t'(\lambda)$, there exists a uniform machine $\mathcal{A}$ of size at most $\lfloor \log \lambda \rfloor$ such that:

$$\mathsf{Adv}^f_{\mathcal{A}_{(t)}}(1^\lambda) \geq 1/2t.$$

The same considerations as in the case of DDH holds for these sets: $\mathbb{N} = \mathsf{SECURE} \cup \mathsf{BROKEN}$, but these do not form a partition of the integers (they can overlap). Then, similar to our construction of tester for DDH, we can prove the existence of a security parameter tester for general one-way functions (we will not need this tester in our actual construction, though, only in the proof of security).

Lemma 8.4 (Universal Security Parameter Tester)**.** *Let $t = t(\lambda)$, $t' = t'(\lambda)$ be positive-integer-valued functions. Then there exists an algorithm $\mathsf{Test}_{\mathsf{OWF}} = \mathsf{Test}_{\mathsf{OWF}}(t, t')$ which takes as input 1^λ where $\lambda \in \mathbb{N}$, runs in time $t' \cdot t^3(\lambda) \cdot \mathsf{poly}(\lambda)$, and satisfying the following properties:*

- *For any $\lambda \in \mathbb{N}$,* $\Pr\left[\lambda \notin \mathsf{SECURE} \wedge \mathsf{Test}_{\mathsf{OWF}}(1^\lambda) = 1\right] \leq 2^{-\lambda}$.
- *For any $\lambda \in \mathbb{N}$,* $\Pr\left[\lambda \notin \mathsf{BROKEN} \wedge \mathsf{Test}_{\mathsf{OWF}}(1^\lambda) = 0\right] \leq \lambda \cdot 2^{-\lambda}$.

Proof. We define $\mathsf{Test}_{\mathsf{OWF}}$ as follows. On input (1^λ), let $C(\lambda) = \lceil 100 \cdot \lambda \cdot t^2 \rceil$.

- for all $f \in \{0,1\}^{\lfloor \log \lambda \rfloor}$, parse f as the description of a Turing Machine. Denote by $f_{(t')}$ the Turing machine that, on input $x \in \{0,1\}^\lambda$, runs f for t' steps and if f terminates with output $y \in \{0,1\}^\lambda$, it outputs y; otherwise, it outputs 0^λ.
- For all $M \in \{0,1\}^{\lfloor \log \lambda \rfloor}$, parse M as the description of a Turing Machine.

- For $i = 1$ to $C(\lambda)$, sample $x_i \xleftarrow{\$} \{0,1\}^\lambda$ and compute $y_i = f_{(t')}(x_i)$. Compute $x'_i \leftarrow M_{(t)}((1^\lambda, y_i)$. Let $c_{M,f}$ denote the number of indices i such that $f_{(t')}(x'_i) = y_i$.

- If for each f there exists M such that $c_{M,f} \geq \frac{3}{4t(\lambda)} \cdot C(\lambda)$, output 0; else, output 1.

From there, the security analysis of the tester is essentially identical to the analysis of Lemma 5.5. □

8.2 The Pass-Venkitasubramaniam Construction

Below, we recall some definitions, adapted to our setting. A distributional NP problem is a pair $(\mathcal{L}, \mathcal{D})$ where $\mathcal{L}$ is an NP language, and $\mathcal{D}$ is a polytime-samplable distribution.

Definition 8.5 (hardness on average)**.** *A distributional problem $(\mathcal{L}, \mathcal{D})$ is (uniformly) (t, ε)-*hard-on-the-average *((t, ε)-HOA) if there exists no t-time (uniform) adversary $\mathcal{A}$ such that for infinitely many $\lambda \in \mathbb{N}$,*

$$\Pr[x \xleftarrow{\$} \mathcal{D}(1^\lambda) \ : \ \mathcal{A}(1^\lambda, x) = \mathcal{L}(x)] > 1 - \varepsilon,$$

where we define $\mathcal{L}(x) = 1$ if $x \in \mathcal{L}$ and $\mathcal{L}(x) = 0$ otherwise.

When $t = \lambda^{\omega(1)}$ and $\varepsilon = 1/2 - \lambda^{-\omega(1)}$, we say that the problem is *superpolynomially* (uniformly) HOA. The definition above can be extended to (uniformly) (t, ε)-HOA *search problems.* A distributional search NP problem is a pair $(\mathcal{R}, \mathcal{D})$ where $\mathcal{R}$ is a search NP problem and $\mathcal{D}$ a polytime-samplable distribution. We recall the notion of infinitely-often hard-on-average search problem from [PV20]:

Definition 8.6 (search ioHOA)**.** *A distributional* NP *search problem $(\mathcal{R}, \mathcal{D})$ is (uniformly) infinitely-often (t, ε)-*hard-on-the-average *((t, ε)-ioHOA) if there exists no t-time (uniform) adversary $\mathcal{A}$ such that for all large enough $\lambda \in \mathbb{N}$,*

$$\Pr[x \xleftarrow{\$} \mathcal{D}(1^\lambda), \mathcal{A}(1^\lambda, x) = w \ : \ \mathcal{L}_\mathcal{R}(x) = 1 \implies (x, w) \in \mathcal{R}] > 1 - \varepsilon.$$

A *promise-true* distributional NP search problem is such that every x in the support of $\mathcal{D}(1^\lambda)$ satisfies $\mathcal{L}_\mathcal{R}(x) = 1$. Eventually, we define HOA search problems which are secure on an superpolynomially dense set of security parameters:

Definition 8.7 (search isoHOA)**.** *A distributional* NP *search problem $(\mathcal{R}, \mathcal{D})$ is (uniformly)* superpolynomially-often (t, ε)-hard-on-the-average *((t, ε)-isoHOA) if for every t-time (uniform) adversary $\mathcal{A}$, the set $S \subset \mathbb{N}$ of security parameters λ such that*

$$\Pr[x \xleftarrow{\$} \mathcal{D}(1^\lambda), \mathcal{A}(1^\lambda, x) = w \ : \ \mathcal{L}_\mathcal{R}(x) = 1 \implies (x, w) \in \mathcal{R}] > 1 - \varepsilon$$

satisfies $[\lambda, \lambda^{\omega(1)}] \not\subset S$ for every $\lambda \in \mathbb{N}$.

Brief overview of the construction of [PV20]. We briefly recall the main steps used in [PV20] to prove Theorem 8.1.

The proof of [PV20] uses the intermediate notion of *interactive puzzle*, which we recall below.

Definition 8.8 (Informal)**.** *An* interactive puzzle *is a 2-player interactive protocol between a challenger C and an attacker $\mathcal{A}$ such that, when running on joint input 1^λ,*

- *(S-computational soundness) there exists no PPT attacker $\mathcal{A}^*$ such that $\mathcal{A}^*(1^\lambda)$ succeeds in making $C(1^\lambda)$ output 1 for every $\lambda \in S$, and*

- *(completeness) there exists a negligible function μ and an* inefficient *attacker $\mathcal{A}$ which, on input 1^λ, succeeds in making $C(1^\lambda)$ output 1 with probability $1 - \mu(\lambda)$ for all $\lambda \in \mathbb{N}$.*

The proof of Theorem 8.1 uses the following sequence of steps:

- A hard-on-average distributional NP problem implies a two-round public-coin puzzle Π.

- A two-round public-coin puzzle implies a 3-round public-coin puzzle with perfect completeness via [PV20, Theorem 6.1].

- If infinitely-often OWFs do not exist, then a 3-round public-coin puzzle with perfect completeness can be converted into a 2-round public-coin puzzle with perfect completeness Π' using the round-collapse lemma [PV20, Lemma 4.4]. Such a puzzle is in particular (syntactically equivalent to) a promise-true (almost-everywhere) distributional NP search problem.

- Else, if infinitely-often OWFs do exist, they imply the existence of a 2-round private-coin puzzle with perfect completeness, which is syntactically equivalent to an infinitely-often promise-true distributional NP search problem.

The heart of the argument is a proof that the Babai-Moran round-reduction theorem [BM88] works also in the setting of computationally sound interactive puzzles. The proof proceeds by showing that given an adversary $\mathcal{A}$ against the round-collapsed protocol, one can construct a new machine M (that incorporates the code of $\mathcal{A}$) and an adversary $\mathcal{B}$ against the original k-round protocol, which internally runs an algorithm Inv which samples from a distribution which is $1/\lambda$ close to the distribution of preimages of M, and succeeds in the k-round puzzle with constant probability $\geq 1/64$. At this stage, the authors conclude that if infinitely-often OWFs do not exist, then via the standard black-box reduction between (infinitely-often) distributional OWFs and OWFs, there exists an efficient implementation of Inv which succeeds on every large enough security parameter λ.

8.3 A New Reduction from Distributional NP Problems to Promise-True Distributional NP Problems

Using our techniques and building upon the sketch above, we prove Theorem 8.2. In particular we can actually describe an explicit construction of a hard-on-average promise-true distributional NP search problem with stronger hardness guarantees.

Proof. Let f be Levin's universal one-way function. Let h_f denote the universal one-way hash function built from f using Rompel's construction [NY89, Rom90] (note that the adjective universal has a different meaning for f and h_f here). Let Π be a superpolynomially-secure two-round public-coin puzzle with imperfect completeness, whose existence is implied by that of a superpolynomially-secure (uniformly) hard-on-average distributional NP problem (see [PV20, Lemma 3.3]). Our (straightforward) construction proceeds as follows:

- On input the security parameter 1^λ, both parties run the 2-round private-coin puzzle Π' with perfect completeness obtained by applying the Babai-Moran round-reduction technique to the 3-round public-coin puzzle with perfect completeness constructed from Π via [PV20, Theorem 6.1] (note that this is a fully explicit construction: only its *soundness analysis* relies on whether a OWF exist).
- In parallel, both parties run the 2-round public-coin perfectly complete puzzle constructed from h_f via [PV20, Proposition 3.2], on joint input 1^λ.
- The challenger accepts if and only if both executions accept.

This protocol simply runs both candidate constructions of 2-round public-coin perfectly complete puzzles from [PV20], making the OWF-based construction explicit by means of a universal one-way function. However, the analysis of [PV20] only guarantees that this combined puzzle is sound on *infinitely many* security parameters. We provide below a sketch of an alternative analysis which shows, starting from slightly stronger premises (the existence of *superpolynomially* hard-on-average distributional NP problems, where the superpolynomial gap can be an arbitrary $\lambda^{\omega(1)}$), that the puzzle is sound on a set of security parameters of superpolynomial density.

Completeness and soundness. Perfect completeness follows immediately from the perfect completeness of each puzzle. We now turn our attention to soundness. arbitrary parameters $t'(\lambda) = \lambda^{\omega(1)}$ and $t(\lambda) = (t'(\lambda))^{\omega(1)}$.

Let $\lambda \in \mathbb{N}$. Assume for now that $\lambda \in \mathsf{BROKEN}(t, t')$. By definition, this means that for every function $f : \{0,1\}^\lambda \mapsto \{0,1\}^\lambda$ with description at most $\lceil \log \lambda \rceil$ running in time at most $t'(\lambda)$, there exists a machine M running in time at most $t(\lambda)$ that, on input $y = f(x)$ for a random x, outputs $x' \in f^{-1}(f(x))$ with probability at least $1/t(\lambda)$. Now, let $\mathcal{A}^*$ be a PPT adversary against Π'. We follow directly the soundness analysis of Lemma 4.1 from [PV20]. From $\mathcal{A}^*$, the authors show how to build a specific polytime-computable function F such that, given an algorithm Inv that sample from a distribution $1/\lambda$-close to random preimages for F, one can construct from Inv an adversary $\mathcal{B}$ against the original protocol Π (in fact again the 3-round protocol , but this directly implies an attack on Π via the reduction of Theorem 6.1 in [PV20]). Because $\lambda \in \mathsf{BROKEN}(t, t')$, we can apply a similar sequence of steps when $\mathcal{A}^*$ is uniform: we assume without loss of generality that the function F constructed from $\mathcal{A}^*$ has description size at most $\lceil \log \lambda \rceil$ (this holds for large enough λ). Let M be the t-time machine that inverts F with probability $1/t$. Using the standard reduction from one-way functions to distributional one-way function, we can build a $\mathrm{poly}(t)$-time machine M' that samples from a distribution $1/t$-close from the distribution of random preimages of F. Plugging M' in the construction of the adversary $\mathcal{B}$ from [PV20], we get an algorithm

that runs in time $\mathrm{poly}(t) = \lambda^{\omega(1)}$ and breaks Π with probability $\lambda^{-\omega(1)}$, which is a contradiction to our assumption that Π is superpolynomially secure.

Now, let $\lambda' \in \mathbb{N}$ be a security parameter such that $t'(\lambda') = \lambda$. Assume that $\lambda' \in \mathsf{SECURE}(t, t')$: this means that there exists a $\lambda = t'(\lambda')$-time function g that cannot be inverted in time $t(\lambda') = \lambda^{\omega(1)}$ with probability better than $\lambda^{\omega(1)}$ – in particular, g is a one-way function against all adversaries running in time $\mathrm{poly}(\lambda)$. Because f is a universal one-way function, it follows that no $\mathrm{poly}(\lambda)$ uniform adversary inverts f with non-negligible advantage, hence that the puzzle constructed from h_f is sound (against uniform adversaries).

We conclude by showing that the set of security parameters such that either $\lambda \in \mathsf{BROKEN}(t, t')$ or $\lambda' \in \mathsf{SECURE}(t, t')$ is superpolynomially dense. This follows immediately from the fact that $\lambda' = \lambda^{1/\omega(1)}$: for each λ, either $\lambda \in \mathsf{BROKEN}(t, t')$, hence the protocol is sound on input 1^λ, or $\lambda \in \mathsf{SECURE}(t, t')$, and the protocol is sound on input $1^{t'(\lambda)} = 1^{\lambda^{\omega(1)}}$. This shows that the gap between two parameters where the protocol is sound is an arbitrarily small superpolynomial gap. □

Acknowledgements

G. Couteau is supported by the French Agence Nationale de la Recherche (ANR), under grant ANR-20-CE39-0001 (project SCENE), and the France 2030 ANR Project ANR22-PECY-003 SecureCompute. The second author was supported in part by NSF CNS-1814919, NSF CAREER 1942789, Johns Hopkins University Catalyst award, JP Morgan Faculty Award, and research gifts from Ethereum, Stellar and Cisco. Zhengzhong Jin was supported in part by DARPA under Agreement No. HR00112020023 and by an NSF grant CNS-2154149. Willy Quach was supported by NSF grant CNS-1750795, CNS-2055510.

References

[BCG+14] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, pages 459–474. IEEE Computer Society, 2014.

[BCM22] E. Boyle, G. Couteau, and P. Meyer. Sublinear secure computation from new assumptions. In *TCC 2022, Part II*, LNCS, pages 121–150. Springer, Heidelberg, November 2022.

[BFJ+20] S. Badrinarayanan, R. Fernando, A. Jain, D. Khurana, and A. Sahai. Statistical ZAP arguments. In *EUROCRYPT 2020, Part III*, *LNCS* 12107, pages 642–667. Springer, Heidelberg, May 2020.

[BFM88] M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pages 103–112. ACM Press, May 1988.

[BKM06] A. Bender, J. Katz, and R. Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. In *TCC 2006*, *LNCS* 3876, pages 60–79. Springer, Heidelberg, March 2006.

[BKM20] Z. Brakerski, V. Koppula, and T. Mour. NIZK from LPN and trapdoor hash via correlation intractability for approximable relations. In *CRYPTO 2020, Part III*, *LNCS* 12172, pages 738–767. Springer, Heidelberg, August 2020.

[BL96] D. Boneh and R. J. Lipton. Algorithms for black-box fields and their application to cryptography (extended abstract). In *CRYPTO'96*, *LNCS* 1109, pages 283–297. Springer, Heidelberg, August 1996.

[BM88] L. Babai and S. Moran. Arthur-merlin games: a randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36(2):254–276, 1988.

[BMW03] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT 2003*, *LNCS* 2656, pages 614–629. Springer, Heidelberg, May 2003.

[BY93] M. Bellare and M. Yung. Certifying cryptographic tools: The case of trapdoor permutations. In *CRYPTO'92*, *LNCS* 740, pages 442–460. Springer, Heidelberg, August 1993.

[CCH+19] R. Canetti, Y. Chen, J. Holmgren, A. Lombardi, G. N. Rothblum, R. D. Rothblum, and D. Wichs. Fiat-Shamir: from practice to theory. In *51st ACM STOC*, pages 1082–1090. ACM Press, June 2019.

[CCRR18] R. Canetti, Y. Chen, L. Reyzin, and R. D. Rothblum. Fiat-Shamir and correlation intractability from strong KDM-secure encryption. In *EUROCRYPT 2018, Part I*, *LNCS* 10820, pages 91–122. Springer, Heidelberg, April / May 2018.

[CH19] G. Couteau and D. Hofheinz. Designated-verifier pseudorandom generators, and their applications. In *EUROCRYPT 2019, Part II*, *LNCS* 11477, pages 562–592. Springer, Heidelberg, May 2019.

[CHK03] R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In *EUROCRYPT 2003*, *LNCS* 2656, pages 255–271. Springer, Heidelberg, May 2003.

[CKS08] D. Cash, E. Kiltz, and V. Shoup. The twin Diffie-Hellman problem and applications. In *EUROCRYPT 2008*, *LNCS* 4965, pages 127–145. Springer, Heidelberg, April 2008.

[CKSU21] G. Couteau, S. Katsumata, E. Sadeghi, and B. Ursu. Statistical ZAPs from group-based assumptions. In *TCC 2021, Part I*, *LNCS* 13042, pages 466–498. Springer, Heidelberg, November 2021.

[CKU20] G. Couteau, S. Katsumata, and B. Ursu. Non-interactive zero-knowledge in pairing-free groups from weaker assumptions. In *EUROCRYPT 2020, Part III*, *LNCS* 12107, pages 442–471. Springer, Heidelberg, May 2020.

[CL18] R. Canetti and A. Lichtenberg. Certifying trapdoor permutations, revisited. In *TCC 2018, Part I*, *LNCS* 11239, pages 476–506. Springer, Heidelberg, November 2018.

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

[DDN91] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography (extended abstract). In *23rd ACM STOC*, pages 542–552. ACM Press, May 1991.

[Den17] Y. Deng. Magic adversaries versus individual reduction: Science wins either way. In *EUROCRYPT 2017, Part II*, *LNCS* 10211, pages 351–377. Springer, Heidelberg, April / May 2017.

[DGH+20] N. Döttling, S. Garg, M. Hajiabadi, D. Masny, and D. Wichs. Two-round oblivious transfer from CDH or LPN. In *EUROCRYPT 2020, Part II*, *LNCS* 12106, pages 768–797. Springer, Heidelberg, May 2020.

[DGI+19] N. Döttling, S. Garg, Y. Ishai, G. Malavolta, T. Mour, and R. Ostrovsky. Trapdoor hash functions and their applications. In *CRYPTO 2019, Part III*, *LNCS* 11694, pages 3–32. Springer, Heidelberg, August 2019.

[DMP88] A. De Santis, S. Micali, and G. Persiano. Non-interactive zero-knowledge proof systems. In *CRYPTO'87*, *LNCS* 293, pages 52–72. Springer, Heidelberg, August 1988.

[DN00] C. Dwork and M. Naor. Zaps and their applications. In *41st FOCS*, pages 283–293. IEEE Computer Society Press, November 2000.

[FLS90] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *31st FOCS*, pages 308–317. IEEE Computer Society Press, October 1990.

[FS87] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO'86*, *LNCS* 263, pages 186–194. Springer, Heidelberg, August 1987.

[GH18] S. Garg and M. Hajiabadi. Trapdoor functions from the computational Diffie-Hellman assumption. In *CRYPTO 2018, Part II*, *LNCS* 10992, pages 362–391. Springer, Heidelberg, August 2018.

[GJJM20] V. Goyal, A. Jain, Z. Jin, and G. Malavolta. Statistical zaps and new oblivious transfer protocols. In *EUROCRYPT 2020, Part III*, *LNCS* 12107, pages 668–699. Springer, Heidelberg, May 2020.

[GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989.

[GMR85] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *17th ACM STOC*, pages 291–304. ACM Press, May 1985.

[GOS06a] J. Groth, R. Ostrovsky, and A. Sahai. Non-interactive zaps and new techniques for NIZK. In *CRYPTO 2006*, *LNCS* 4117, pages 97–111. Springer, Heidelberg, August 2006.

[GOS06b] J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. In *EUROCRYPT 2006*, *LNCS* 4004, pages 339–358. Springer, Heidelberg, May / June 2006.

[GR13] O. Goldreich and R. D. Rothblum. Enhancements of trapdoor permutations. *Journal of Cryptology*, 26(3):484–512, July 2013.

[HNO+18] I. Haitner, K. Nissim, E. Omri, R. Shaltiel, and J. Silbak. Computational two-party correlation: A dichotomy for key-agreement protocols. In *59th FOCS*, pages 136–147. IEEE Computer Society Press, October 2018.

[JJ21] A. Jain and Z. Jin. Non-interactive zero knowledge from sub-exponential DDH. In *EUROCRYPT 2021, Part I*, *LNCS* 12696, pages 3–32. Springer, Heidelberg, October 2021.

[KNYY19] S. Katsumata, R. Nishimaki, S. Yamada, and T. Yamakawa. Designated verifier/prover and preprocessing NIZKs from Diffie-Hellman assumptions. In *EUROCRYPT 2019, Part II*, *LNCS* 11477, pages 622–651. Springer, Heidelberg, May 2019.

[KY18] I. Komargodski and E. Yogev. On distributional collision resistant hashing. In *CRYPTO 2018, Part II*, *LNCS* 10992, pages 303–327. Springer, Heidelberg, August 2018.

[LQR+19] A. Lombardi, W. Quach, R. D. Rothblum, D. Wichs, and D. J. Wu. New constructions of reusable designated-verifier NIZKs. In *CRYPTO 2019, Part III*, *LNCS* 11694, pages 670–700. Springer, Heidelberg, August 2019.

[LVW19] A. Lombardi, V. Vaikuntanathan, and D. Wichs. 2-message publicly verifiable WI from (subexponential) LWE. Cryptology ePrint Archive, Report 2019/808, 2019. https://eprint.iacr.org/2019/808.

[Mau94] U. M. Maurer. Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete algorithms. In *CRYPTO'94*, *LNCS* 839, pages 271–281. Springer, Heidelberg, August 1994.

[MPS10] H. K. Maji, M. Prabhakaran, and A. Sahai. On the computational complexity of coin flipping. In *51st FOCS*, pages 613–622. IEEE Computer Society Press, October 2010.

[NY89] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *21st ACM STOC*, pages 33–43. ACM Press, May 1989.

[NY90] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pages 427–437. ACM Press, May 1990.

[OPWW15] T. Okamoto, K. Pietrzak, B. Waters, and D. Wichs. New realizations of somewhere statistically binding hashing and positional accumulators. In *ASIACRYPT 2015, Part I*, *LNCS* 9452, pages 121–145. Springer, Heidelberg, November / December 2015.

[PS19] C. Peikert and S. Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In *CRYPTO 2019, Part I*, *LNCS* 11692, pages 89–114. Springer, Heidelberg, August 2019.

[PsV06] R. Pass, a. shelat, and V. Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In *CRYPTO 2006*, *LNCS* 4117, pages 271–289. Springer, Heidelberg, August 2006.

[PV20] R. Pass and M. Venkitasubramaniam. Is it easier to prove theorems that are guaranteed to be true? In *61st FOCS*, pages 1255–1267. IEEE Computer Society Press, November 2020.

[PW08] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *40th ACM STOC*, pages 187–196. ACM Press, May 2008.

[QRW19] W. Quach, R. D. Rothblum, and D. Wichs. Reusable designated-verifier NIZKs for all NP from CDH. In *EUROCRYPT 2019, Part II*, *LNCS* 11477, pages 593–621. Springer, Heidelberg, May 2019.

[Rom90] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd ACM STOC*, pages 387–394. ACM Press, May 1990.

[RTV04] O. Reingold, L. Trevisan, and S. P. Vadhan. Notions of reducibility between cryptographic primitives. In *TCC 2004, LNCS* 2951, pages 1–20. Springer, Heidelberg, February 2004.

[RV22] R. D. Rothblum and P. N. Vasudevan. Collision-resistance from multi-collision-resistance. In *CRYPTO 2022, Part III*, LNCS, pages 503–529. Springer, Heidelberg, August 2022.

[SW14] A. Sahai and B. Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014.

[Zha19] M. Zhandry. Quantum lightning never strikes the same state twice. In *EUROCRYPT 2019, Part III*, *LNCS* 11478, pages 408–438. Springer, Heidelberg, May 2019.