Exploring Constructions of Compact NIZKs from Various Assumptions

Shuichi Katsumata[1,2], Ryo Nishimaki[3], Shota Yamada[1], and Takashi Yamakawa[3]

[1] AIST, Tokyo, Japan
{shuichi.katsumata,yamada-shota}@aist.go.jp
[2] The University of Tokyo, Tokyo, Japan
[3] NTT Secure Platform Laboratories, Tokyo, Japan
{ryo.nishimaki.zk, takashi.yamakawa.ga}@hco.ntt.co.jp

Abstract. A non-interactive zero-knowledge (NIZK) protocol allows a prover to non-interactively convince a verifier of the truth of the statement without leaking any other information. In this study, we explore shorter NIZK proofs for all **NP** languages. Our primary interest is NIZK proofs from falsifiable pairing/pairing-free group-based assumptions. Thus far, NIZKs in the common reference string model (CRS-NIZKs) for **NP** based on falsifiable pairing-based assumptions all require a proof size at least as large as $O(|C|\kappa)$, where C is a circuit computing the **NP** relation and κ is the security parameter. This holds true even for the weaker designated-verifier NIZKs (DV-NIZKs). Notably, constructing a (CRS, DV)-NIZK with proof size achieving an *additive*-overhead $O(|C|)+\mathsf{poly}(\kappa)$, rather than a multiplicative-overhead $|C|\cdot\mathsf{poly}(\kappa)$, based on any falsifiable pairing-based assumptions is an open problem.

In this work, we present various techniques for constructing NIZKs with *compact* proofs, i.e., proofs smaller than $O(|C|) + \mathsf{poly}(\kappa)$, and make progress regarding the above situation. Our result is summarized below.

- We construct CRS-NIZK for all **NP** with proof size $|C| + \mathsf{poly}(\kappa)$ from a (non-static) falsifiable Diffie-Hellman (DH) type assumption over pairing groups. This is the first CRS-NIZK to achieve a compact proof without relying on either lattice-based assumptions or non-falsifiable assumptions. Moreover, a variant of our CRS-NIZK satisfies universal composability (UC) in the erasure-free adaptive setting. Although it is limited to **NP** relations in $\mathbf{NC}^1$, the proof size is $|w| \cdot \mathsf{poly}(\kappa)$ where w is the witness, and in particular, it matches the state-of-the-art UC-NIZK proposed by Cohen, shelat, and Wichs (CRYPTO'19) based on lattices.
- We construct (multi-theorem) DV-NIZKs for **NP** with proof size $|C|+\mathsf{poly}(\kappa)$ from the computational DH assumption over *pairing-free* groups. This is the first DV-NIZK that achieves a compact proof from a standard DH type assumption. Moreover, if we further assume the **NP** relation to be computable in $\mathbf{NC}^1$ and assume hardness of a (non-static) falsifiable DH type assumption over *pairing-free* groups, the proof size can be made as small as $|w| + \mathsf{poly}(\kappa)$.

Another related but independent issue is that all (CRS, DV)-NIZKs require the running time of the prover to be at least $|C| \cdot \mathsf{poly}(\kappa)$. Considering that there exists NIZKs with efficient verifiers whose running time is

strictly smaller than $|C|$, it is an interesting problem whether we can construct *prover-efficient* NIZKs. To this end, we construct prover-efficient CRS-NIZKs for **NP** with compact proof through a generic construction using laconic functional evaluation schemes (Quach, Wee, and Wichs (FOCS'18)). This is the first NIZK in any model where the running time of the prover is strictly smaller than the time it takes to compute the circuit C computing the **NP** relation.
Finally, perhaps of an independent interest, we formalize the notion of *homomorphic equivocal commitments*, which we use as building blocks to obtain the first result, and show how to construct them from pairing-based assumptions.

1 Introduction

1.1 Background

Zero-knowledge (ZK) protocols, introduced by Goldwasser, Micali, and Rackoff [40], allow a prover to convince a verifier of the truth of a statement without leaking any knowledge other than the fact that the statement is indeed true. A practically useful and theoretically alluring feature for a ZK protocol to have is *non-interactiveness*, where a prover simply outputs a single message (called a proof) and convinces the verifier of the truth of the statement. Unfortunately, it is known that non-interactive ZK (NIZK) for non-trivial languages do not exist in the plain model where there is no trusted setup [39]. However, Blum, Feldman, and Micali [11] showed how to construct a NIZK in a setting where the prover and verifier have access to a shared *common reference string* (as known as CRS-NIZK). Since then, NIZKs have been used as a ubiquitous building block for cryptography ranging from the early chosen-ciphertext secure public key encryption schemes [71,30,81], advanced signature schemes [22,78,6], and multi-party computation [38].

Compact NIZK. One of the important research topics for NIZK is making the proof size as small as possible. So far, CRS-NIZK for all of **NP** in the standard model is known to exist from (doubly-enhanced) trapdoor permutation [31,7,37], pairing [45,46,62,48,49,33], indistinguishability obfuscation (iO) [82,9,10,18], or correlation intractable hash function [51,16,15]. Among these, CRS-NIZKs that have proof size independent of the size of the circuit C computing the **NP** relation are limited to those based on either a knowledge assumption [46,62,33] or iO [82]. There also exist generic conversions from standard CRS-NIZKs to CRS-NIZKs with proof size independent of $|C|$. However, they rely on fully homomorphic encryption (FHE) [34,35] or homomorphic trapdoor functions (HTDF) [23] whose existence is only implied from lattice-based assumptions. Put differently, the classical CRS-NIZKs based on trapdoor permutations or (falsifiable [68,36]) pairing-based assumptions all require a large proof size that is polynomially related to the circuit size $|C|$. Notably, even the most well-known Groth-Ostrovsky-Sahai NIZK (GOS-NIZK) [48] based on the decisional linear or subgroup decision assumptions over pairing groups requires the proof size

to be as large as $O(|C|\kappa)$, where κ is the security parameter. In fact, the CRS-NIZK with the shortest proof that does not rely on any of the above strong tools is the NIZK of Groth [45] based on the security of Naccache-Stern public key encryption scheme [67] which achieves proof size $|C| \cdot \mathsf{polylog}(\kappa)$. Therefore, it remains an interesting open problem to construct CRS-NIZKs with proof size smaller than the current state-of-the-art while avoiding to rely on strong tools such as knowledge assumptions, iO, FHE, and HTDF. Specifically, in this paper, one of the primary interest is to obtain a CRS-NIZK with proof size achieving an *additive*-overhead $O(|C|) + \mathsf{poly}(\kappa)$, rather than a multiplicative-overhead $|C| \cdot \mathsf{poly}(\kappa)$ (or $|C| \cdot \mathsf{polylog}(\kappa)$), based on any falsifiable pairing-based assumptions. Hereafter, we call such NIZKs with proof size $O(|C|) + \mathsf{poly}(\kappa)$ as NIZKs with *compact* proofs for simplicity.

Designated Verifier NIZKs and Compact Proofs. A relaxation of CRS-NIZKs called the *designated verifier* NIZKs (DV-NIZKs) [73,27] retain most of the useful properties of CRS-NIZKs and in some applications can be used as a substitute for CRS-NIZKs. The main difference between CRS and DV-NIZKs is that the latter limits the proof to only be verifiable by a designated party in possession of a verification key; the proof can still be generated by anybody as in CRS-NIZKs. Due to this extra secret information possessed by the verifier, DV-NIZKs suffer from the so-called verifier rejection attack. Specifically, a prover may learn partial information of the secret verification key and break soundness if the verifier uses the same verification key for verifying multiple statements. In this paper, our primary interest is *multi-theorem* DV-NIZKs (also known as *reusable* or *unbounded-soundness* DV-NIZKs) where the verification key can be reused for multiple statements without compromising soundness. Surprisingly, most DV-NIZKs [73,27,84,21,64,20] (that are not a simple downgrade of CRS-NIZKs) are known to either suffer from the verifier rejection attack or to be limited to specific **NP** languages. It was not until recently that the first multi-theorem DV-NIZK for all **NP** languages was (concurrently and independently) shown by Couteau and Hofheinz [24], Katsumata et al. [53], and Quach et al. [75]. They proposed a tweak to the classical Feige-Lapidot-Shamir (FLS) NIZK protocol [31] and showed for the first time how to construct DV-NIZKs from the computational Diffie-Hellman (CDH) assumption over *pairing-free* groups; an assumption which is not yet known to imply CRS-NIZKs. However, one drawback of their DV-NIZK is that the CRS size and proof size are huge, i.e., $\mathsf{poly}(\kappa, |C|)$. This is due to the fact that the FLS NIZK, which they base their construction on, is highly specific to the **NP**-complete Hamiltonicity problem. It is unclear if we can make their scheme compact since all other (CRS-)NIZKs following the footsteps of FLS NIZK such as [54,56,45] suffer from the same problem of having large CRS and proof size. Therefore, it is unclear whether such a weak assumption as CDH over pairing-free groups can be used to construct a DV-NIZK with compact proofs. In fact, constructing DV-NIZKs with compact proof from any *pairing/pairing-free* group assumptions remains open.

Prover-Efficient NIZKs. Continuing the line of NIZKs with compact proofs, it is very natural and appealing to consider NIZKs that enjoy *efficient provers*, i.e.,

the running time of the prover is small. We say the prover is efficient if its running time is strictly smaller than the time it takes to compute $C(x, w)$ for statement x and witness w, where recall C was the circuit computing the **NP** relation. As an example, we can imagine a case where a user (acting as a prover) is given some sort of credential w as a witness by a trusted authority and is required to prove in zero-knowledge the fact that it possesses a valid credential to make some action. More concretely, in group signatures [6] a trusted authority will provide users with a credential which allows them to sign anonymously on behalf of the group. In such a case, it would be appealing if the user could generate a proof without requiring to invest computational time-dependent of $|C|$, since if zero-knowledge was not required, the prover could have simply output the credential w in the clear and completely outsourced the computation of $C(x, w)$ to the verifier. Since the authority is providing a valid credential w to the user, in principle, the user should never need to compute $C(x, w)$ to check whether w is valid.

As far as our knowledge goes, all NIZKs, regardless of CRS or DV, have a prover with running time at least $|C| \cdot \mathsf{poly}(\kappa)$ which is much larger than the time it takes to simply compute the circuit C. We emphasize that solutions to the counterpart notion of *efficient verifiers* are well known and studied. Specifically, NIZKs with compact proofs with the additional property of having efficient verifiers are known as ZK-succinct non-interactive arguments (ZK-SNARGs) or ZK-succinct non-interactive arguments of knowledge (ZK-SNARKs).[4] They have been the subject of extensive research, e.g., [45,62,8,33,63,28,72,47], where constructions are known to exist either in the random oracle model or based on non-falsifiable assumptions. We also note that it would be impossible to construct a NIZK where both the prover *and* the verifier are efficient since the circuit C representing the **NP** relation must be computed by at least one of the parties to check the validity of the witness w. Therefore, it is an interesting question of whether there exists an opposite flavor of the current NIZKs where we have an efficient prover instead of an efficient verifier.

1.2 Our Contribution

In this paper, we provide new constructions of CRS-NIZK and DV-NIZK with compact proofs. The former is instantiated on a pairing group and the latter on a paring-free group. The tools and techniques which we use for our CRS-NIZK can be slightly modified to construct universally composable NIZK (UC-NIZK) [48] with compact proofs over pairing groups. Finally, we provide a generic construction of a CRS-NIZK with an efficient prover using as a building block the recently proposed laconic functional evaluation (LFE) scheme of Quach, Wee, and Wichs [76]. We summarize our results below and refer to Table 1, 2, and 3 for a comparison between prior works. We note that we only include multi-theorem NIZKs supporting all of **NP** based on falsifiable assumptions in the table.

[4] We note that in ZK-SNARG/SNARK, it is conventional to require an efficient verifier to have running time that is only poly-logarithmic dependent of $|C|$, rather than being just strictly smaller than $|C|$.

1. We construct CRS-NIZKs for **NP** with compact proof from a (non-static) assumption over pairing groups, namely, the (n, m)-computational Diffie-Hellman exponent and ratio (CDHER) assumption introduced by [53]. This is the first CRS-NIZK to achieve a compact proof without relying on either lattice-based assumptions, knowledge assumptions, or indistinguishability obfuscation. The proof size has an additive-overhead $|C|+\mathsf{poly}(\kappa)$, rather than a multiplicative-overhead $|C| \cdot \mathsf{poly}(\kappa)$, where C is the circuit that computes the **NP** relation (See Table 1). Moreover, if we assume the **NP** relation to be computable in $\mathbf{NC}^1$, we can make the proof size as small as $|w| + \mathsf{poly}(\kappa)$, where w is the witness. This matches the proof size of the CRS-NIZK of Gentry et al. [35] based on fully-homomorphic encryption.

2. We construct UC-NIZKs for **NP** relations in $\mathbf{NC}^1$ with compact proof from the (n, m)-CDHER assumption. Although it is limited to **NP** relations in $\mathbf{NC}^1$, it matches the smallest proof size among all the UC-NIZKs secure against adaptive corruptions in the erasure-free setting (See Table 2). The proof size is small as $|w| \cdot \mathsf{poly}(\kappa)$, and in particular, matches the recent UC-NIZK of Cohen, shelat, and Wichs [23] based on lattice-assumptions. Here, note that for $\mathbf{NC}^1$ circuits, the dependence on the depth d they have can be ignored, since asymptotically d is smaller than κ.

3. We construct (multi-theorem) DV-NIZKs for **NP** with compact proof from the CDH assumption over *pairing-free* groups. This is the first DV-NIZK that achieves a compact proof from a weak and static Diffie-Hellman type assumption such as CDH. Specifically, similarly to the above CRS-NIZK, the proof size of our DV-NIZK is $|C| + \mathsf{poly}(\kappa)$, whereas all previous DV-NIZKs had proof size $\mathsf{poly}(|C|, \kappa)$ (See Table 3). Moreover, if we further assume the **NP** relation to be computable in $\mathbf{NC}^1$ and assume the hardness of the parameterized ℓ-computational Diffie-Hellman inversion (CDHI) assumption over *pairing-free* groups [66,19], we can make the proof size as small as $|w| + \mathsf{poly}(\kappa)$.

4. Finally, we construct prover-efficient CRS-NIZKs for **NP** through a generic construction using LFE schemes [76]. This is the first NIZK in any model (e.g., CRS, DV) where the running time of the prover is strictly smaller than the time it takes to compute the circuit C computing the **NP** relation. Using any non-prover-efficient CRS-NIZK, we generically construct a CRS-NIZK where the running time of the prover (and the proof size) is $\mathsf{poly}(\kappa, |x|, |w|, d)$, independent of the circuit size $|C|$, by instantiating the LFE scheme by the sub-exponential security of the learning with errors (LWE) assumption with sub-exponential modulus-to-noise ratio, where x is the statement and d is the depth of C. Moreover, if we use as building block a CRS-NIZK whose prover running time is smaller than $|C| \cdot \mathsf{poly}(\kappa)$ (e.g., [48]), the running time and proof size can be made as small as $\tilde{O}(|x| + |w|) \cdot \mathsf{poly}(\kappa, d)$ by instantiating the LFE scheme by the *adaptive* LWE assumption with sub-exponential modulus-to-noise ratio introduced in [76].

Along the way of obtaining our first and second results, we formalize a new tool called *homomorphic equivocal commitments* (HEC)[5], which may be of independent interest. An HEC is a commitment with two additional properties called *equivocality* and *homomorphism*. The equivocality enables one to generate a commitment that can be opened to any message by using a master secret key. The homomorphism for a circuit family $\mathcal{C} = \{C : \mathcal{X} \to \mathcal{Z}\}$ informally requires that one can commit to a message $\mathbf{x} \in \mathcal{X}$, where its commitment com can be further publicly modified to a commitment com_C on the message $C(\mathbf{x}) \in \mathcal{Z}$ for any circuit $C \in \mathcal{C}$. Here, a decommitment for com_C can be computed by the knowledge of the message $\mathbf{x}$, decommitment of com, and the circuit C. To the knowledgeable readers, we note that HEC is a strictly weaker primitive compared to homomorphic trapdoor functions [42]. Previously, an HEC supporting the family of all polynomial-sized circuits were only (implicitly) known from lattice-based assumptions [42]. Apart from their construction, known (implicit) constructions of HEC only support linear functions [74] or group operations on a pairing group [2]. In this paper, we provide the first instantiation of HEC supporting $\mathbf{NC}^1$ based on any pairing-based assumptions, namely, the (n, m)-CDHER assumption introduced in [53]. The construction is inspired by the recent construction of compact homomorphic signatures of Katsumata et al. [53]. The proposed HEC enjoys a particular form of compactness which is especially useful for generically converting CRS-NIZKs with non-compact proofs to CRS-NIZKs with compact proofs. Concretely, for any polynomially-sized circuit C, the evaluated commitment com_C and its decommitment of our HEC are of size $\mathsf{poly}(\kappa)$ independent of $|C|$, and one can verify the validity of the decommitment in time $\mathsf{poly}(\kappa)$ independent of $|C|$. Somewhat surprisingly, we also construct another instantiation of HEC supporting $\mathbf{NC}^1$ based on the CDH assumption over pairing groups. Although this HEC does not enjoy compactness, and hence cannot be used for our compact CRS-NIZK conversion, we believe it to be an interesting primitive on its own since we achieve homomorphic computations in $\mathbf{NC}^1$ from such a weak assumption as CDH.

1.3 Technical Overview

Our results can be broken up into three parts. The first two results concerning CRS and UC-NIZKs with short proof are obtained through a generic conversion from NIZKs with non-compact proofs to NIZKs with compact proofs using homomorphic equivocal commitments (HEC); a primitive which we formalize and provide instantiations in this work. The third result concerning DV-NIZKs with short proof size based on pairing-free groups, that is, CDH and ℓ-CDHI, are obtained by extending the recent result of Katsumata et al. [53] which constructs the first NIZKs in the preprocessing model (PP-NIZKs) with short proof size from

[5] This primitive was already informally mentioned in [42] and we do not take credit for proposing the concept of HEC. We note that Abe et al. [2] also introduced a similar primitive with the name *homomorphic trapdoor commitments*.

Table 1. Comparison of CRS-NIZKs for **NP**.

Reference	CRS size	Proof size	Assumption (Misc.)
FLS [31]	$\mathsf{poly}(\kappa, \lvert C\rvert)$	$\mathsf{poly}(\kappa, \lvert C\rvert)$	trapdoor permutation†
Groth [45]	$\lvert C\rvert \cdot k_{\mathsf{tpm}} \cdot \mathsf{polylog}(\kappa) + \mathsf{poly}(\kappa)$	$\lvert C\rvert \cdot k_{\mathsf{tpm}} \cdot \mathsf{polylog}(\kappa) + \mathsf{poly}(\kappa)$	trapdoor permutation†
Groth [45]	$\lvert C\rvert \cdot \mathsf{polylog}(\kappa) + \mathsf{poly}(\kappa)$	$\lvert C\rvert \cdot \mathsf{polylog}(\kappa) + \mathsf{poly}(\kappa)$	Naccache-Stern PKE
GOS [48]	$\mathsf{poly}(\kappa)$	$O(\lvert C\rvert\kappa)$	DLIN/SD
CHK, Abusalah [17,3]	$\mathsf{poly}(\kappa, \lvert C\rvert)$	$\mathsf{poly}(\kappa, \lvert C\rvert)$	CDH (pairing group)
GGIPSS [35]	$\mathsf{poly}(\kappa)$	$\lvert w\rvert + \mathsf{poly}(\kappa)$	FHE and CRS-NIZK circular security
Sec. 3	$\mathsf{poly}(\kappa, \lvert C\rvert)$	$\lvert C\rvert + \mathsf{poly}(\kappa)$	(n, m)-CDHER
Sec. 3	$\mathsf{poly}(\kappa, \lvert C\rvert, 2^d)$	$\lvert w\rvert + \mathsf{poly}(\kappa)$	(n, m)-CDHER (limited to $\mathbf{NC}^1$ relation)
Sec. 5	$\mathsf{poly}(\kappa, \lvert x\rvert, \lvert w\rvert, d)$	$\mathsf{poly}(\kappa, \lvert x\rvert, \lvert w\rvert, d)$	LFE and CRS-NIZK (prover-efficient, implied by sub-exp. LWE)
Sec. 5	$(\lvert x\rvert + \lvert w\rvert) \cdot \mathsf{poly}(\kappa, d)$	$\tilde{O}(\lvert x\rvert + \lvert w\rvert) \cdot \mathsf{poly}(\kappa, d)$	LFE and CRS-NIZK‡ (prover-efficient, implied by adaptive LWE)

In column "CRS size" and "Proof size", κ is the security parameter, $|x|, |w|$ is the statement and witness size, $|C|$ and d are the size and depth of the circuit computing the **NP** relation, and k_{tpm} is the length of the domain of the trapdoor permutation. In column "Assumption", DLIN stands for the decisional linear assumption, SD stands for the subgroup decision assumption, (n, m)-CDHER stands for the (parameterized) computational DH exponent and ratio assumption, LFE stands for laconic functional evaluation, and sub-exp. LWE stands for sub-exponentially secure learning with errors (LWE).

† If the domain of the permutation is not $\{0,1\}^n$, we further assume they are doubly enhanced [37].

‡ We additionally require a mild assumption that the prover run time is linear in the size of the circuit computing the **NP** relation.

Table 2. Comparison of UC-NIZKs for **NP**.

Reference	Security (erasure-free)	CRS size	Proof size	Assumption (Misc.)
GOS [48]	adaptive (✓)	$\mathsf{poly}(\kappa)$	$O(\lvert C\rvert\kappa)$	DLIN/SD
GGIPSS [35]	adaptive (✗)	$\mathsf{poly}(\kappa)$	$\lvert w\rvert + \mathsf{poly}(\kappa)$	FHE and UC-NIZK (circular security)
CsW [23]	adaptive (✓)	$\mathsf{poly}(\kappa, d)$	$\lvert w\rvert \cdot \mathsf{poly}(\kappa, d)$	HTDF and UC-NIZK
Sec. 3	adaptive (✓)	$\mathsf{poly}(\kappa, \lvert C\rvert, 2^d)$	$\lvert w\rvert \cdot \mathsf{poly}(\kappa)$	(n, m)-CDHER (limited to $\mathbf{NC}^1$ relation)

In column "CRS size" and "Proof size", κ is the security parameter, $|w|$ is the witness size, $|C|$ and d are the size and depth of circuit computing the **NP** relation. In column "Assumption", DLIN stands for the decisional linear assumption, SD stands for the subgroup decision assumption, HTDF stands for homomorphic trapdoor functions, and (n, m)-CDHER stands for the (parameterized) computational DH exponent and ratio assumption.

pairing-free groups. As explained later, PP-NIZK is a strictly weaker primitive compared to DV-NIZK. Finally, the fourth result concerning prover-efficient NIZK is obtained by a generic construction based on the recently developed

Table 3. Comparison of DV-NIZKs for **NP**.

Reference	CRS size	Proof size	Verification key size	Assumption (Misc.)
CH, KNYY, QRW [24,53,75]	$\mathsf{poly}(\kappa, \lvert C\rvert)$	$\mathsf{poly}(\kappa, \lvert C\rvert)$	$\mathsf{poly}(\kappa, \lvert C\rvert)$	CDH (pairing-free group)
Sec. 4	$\mathsf{poly}(\kappa)$	$\lvert C\rvert + \mathsf{poly}(\kappa)$	$\mathsf{poly}(\kappa)$	CDH (pairing-free group)
Sec. 4	$2^d \cdot \mathsf{poly}(\kappa)$	$\lvert w\rvert + \mathsf{poly}(\kappa)$	$\mathsf{poly}(\kappa)$	ℓ-CDHI (pairing-free group, limited to $\mathbf{NC}^1$ relation)

In the columns concerning sizes, κ is the security parameter, $|w|$ is the witness-size, $|C|$ and d are the size and depth of the circuit computing the **NP** relation. In column "Assumption", ℓ-CDHI stands for the ℓ-computational Diffie-Hellman inversion assumption.

laconic function evaluation scheme of Quach et al. [76]. In the following, we explain these approaches in more detail.

Generic Construction of Compact (CRS, UC)-NIZK from HEC. Here, we explain our construction of compact CRS-NIZK. Our starting point is the recent result by Katsumata et al. [53], who constructed a *designated prover* NIZK (DP-NIZK) with compact proof, where DP-NIZK is an analogue of DV-NIZK where the prover requires secret information to generate proofs and anybody can publicly verify the proofs. Since the construction of Katsumata et al. is an instantiation of the generic conversion from homomorphic signature to DP-NIZK proposed by Kim and Wu [57], we first briefly review Kim and Wu's conversion. Recall that in homomorphic signature, a signature $\boldsymbol{\sigma}$ on a message $\mathbf{m} \in \{0,1\}^\ell$ generated by a secret key sk, can be homomorphically evaluated to a signature σ on $C(\mathbf{m})$ for a circuit $C : \{0,1\}^\ell \to \{0,1\}$. Anybody can verify the validity of the signature by using a public verification key vk and the circuit C. As for the security requirements, we need that given a verification key vk and a signature $\boldsymbol{\sigma}$ on $\mathbf{m}$, it is computationally hard to forge a signature σ^* on z such that $z \neq C(\mathbf{m})$ (unforgeability) and an honestly evaluated signature σ on z does not reveal information about $\mathbf{m}$ beyond the fact that it was derived from a signature on $\mathbf{m}$ such that $C(\mathbf{m}) = z$ (context-hiding). Furthermore, as an efficiency requirement, we need that the size of σ is independent of the size of the circuit C. In Kim and Wu's construction of DP-NIZK, the prover is given a signature $\boldsymbol{\sigma}$ on a secret key $\mathbf{k}$ of a secret key encryption (SKE) scheme as the secret proving key. When the designated prover proves that x is in some language $\mathcal{L}$ that is specified by a relation $\mathcal{R}$, it generates an encryption ct of the witness w such that $(x, w) \in \mathcal{R}$ and homomorphically evaluates the signature $\boldsymbol{\sigma}$ with respect to a circuit that computes $f_{x,\mathsf{ct}}$, where $f_{x,\mathsf{ct}}$ is a function that takes as input $\mathbf{k}'$ and outputs whether $(x, \mathsf{SKE.Dec}(\mathbf{k}', \mathsf{ct})) \in \mathcal{R}$. The proof for DP-NIZK is then set as ct and the homomorphically evaluated signature σ. The verifier prepares the function $f_{x,\mathsf{ct}}$ from ct and x, and simply checks σ is a correct signature on 1 with respect to the evaluated function $f_{x,\mathsf{ct}}$. The soundness

of the protocol follows from the unforgeability of the homomorphic signature since $f_{x,\mathsf{ct}}(\mathbf{k}') = 0$ for any $\mathbf{k}'$ when $\mathbf{x}$ is not in the language induced by the relation $\mathcal{R}$. Furthermore, the zero-knowledge property of the protocol follows from the security of SKE and the context-hiding property of the homomorphic signature. Katsumata et al. [53] gave a new homomorphic signature scheme with short evaluated signature σ that supports the function class of $\mathbf{NC}^1$ circuits based on a newly introduced (non-static) pairing-based assumption called the (n, m)-computational Diffie-Hellman exponent and ratio (CDHER) assumption. Plugging this homomorphic signature into the Kim-Wu conversion, they obtained the first compact DP-NIZK for all **NP** based on any pairing-based assumptions.[6]

The aim of our work is to modify the Kim-Wu conversion and remove the necessity of the prover keeping secret information to generate a proof so that we can convert the compact DP-NIZK of Katsumata et al. into a compact CRS-NIZK. The main reason why their construction cannot be used as a CRS-NIZK is because the prover cannot generate the signature $\boldsymbol{\sigma}$ on the fly without knowing the signing key sk of the homomorphic signature. To this end, our first idea is to let the prover choose vk, sk, and $\mathbf{k}$ on its own. This would allow the prover to generate a proof as in the designated prover setting since it can generate the signature $\boldsymbol{\sigma}$ on $\mathbf{k}$ on its own by using the signing key sk. The proof for the CRS-NIZK will then consist of the verification key vk and a proof of the DP-NIZK. Unfortunately, there are multiple of problems with this naive approach. The first problem is that the size of the verification key vk used in Katsumata et al. [53] is polynomially dependent on the size of the circuit that computes the relation to be proven, and thus, this ruins the compactness property of the original DP-NIZK proof. The second problem is that we can no longer invoke the unforgeability of the homomorphic signature to prove soundness since unforgeability holds against adversaries who only has access to a verification key vk and a signature $\boldsymbol{\sigma}$. Indeed, in the specific case of Katsumata et al.'s homomorphic signature scheme, an adversary will be able to completely break the soundness of the resulting scheme if it is further given the signing key sk. Therefore, to resolve these problems, we make use of the special structure that the homomorphic signature scheme of Katsumata et al. has and abstract it to a primitive which we call homomorphic equivocal commitments (HEC).

Our key observation is that in the Katsumata et al.'s homomorphic signature scheme, the reverse direction of the signing procedure is possible *without* the knowledge of the secret signing key sk if we are allowed to program part of the verification key vk. Namely, the verification key vk can be divided into two parts vk_0 and vk_1 where the size of vk_1 is compact (i.e., independent of the size of the circuit), and for a fixed vk_0 and $\mathbf{k}$, one can sample a signature $\boldsymbol{\sigma}$ and efficiently compute the remaining part of the verification key vk_1 without knowledge of the secret signing key sk so that $\boldsymbol{\sigma}$ is a valid signature on $\mathbf{k}$ with respect to the

[6] Note that any **NP** relation can be converted to an **NP** relation in $\mathbf{NC}^1$ by expanding the witness size as large as the circuit computing the original **NP** relation. Notably, a homomorphic signature scheme supporting the function class of $\mathbf{NC}^1$ circuits is sufficient for constructing DP-NIZK for all of **NP**.

entire verification key $\mathsf{vk} = (\mathsf{vk}_0, \mathsf{vk}_1)$. We then modify our above idea using this reverse direction of computation. Namely, we put the non-compact part of the verification key vk_0 in the common reference string. The prover first choose $\mathbf{k}$, $\boldsymbol{\sigma}$ on its own and then computes the remaining compact part of the verification key vk_1 from them so that $\boldsymbol{\sigma}$ is a valid signature on $\mathbf{k}$ with respect to the verification key vk. Notably, the prover no longer requires knowledge of the secret signing key sk, and thus, the prover can generate a proof publicly. The resulting proof is the same as in the case for the above naive construction except that we now only append vk_1 to the underlying DP-NIZK proof, rather than vk_0 *and* vk_1. The first problem of having a large proof size we encountered in our above attempt is now resolved since we moved the non-compact part of the verification key vk_0 to the common reference string and the proof now only contains the compact vk_1 and the compact proof of the underlying DP-NIZK. At first glance, the second problem of losing soundness seems to be resolved as well, as the prover is choosing the signature $\boldsymbol{\sigma}$ without knowledge of the underlying secret signing key sk. However, we encounter a new problem. Namely, once again, we cannot directly use the unforgeability of the homomorphic signature to prove soundness, since this time the part of the verification key vk_1 that the adversary appends to the underlying DP-NIZK proof may be maliciously chosen in a way that deviates from the security setting of the homomorphic signature. However, luckily, the proof for unforgeability provided by Katsumata et al. can be adapted without much change to the setting where vk_1 follows an arbitrary distribution since their proof does not depend on the specific distribution which vk_1 is chosen from. In this work, to capture this special security requirement as well as the syntactic structure that we require for the homomorphic signature, we introduce a new primitive that we call *homomorphic equivocal commitment* (HEC) and instantiate it by mimicking the homomorphic signature scheme of Katsumata et al. [53]. Roughly speaking, in our formulation, we regard vk_1 as a commitment of a message $\mathbf{k}$ with respect to a randomness $\boldsymbol{\sigma}$.

While the above explanation conveys our main idea, we need some more modification to obtain our final construction. In the above construction, an honest prover outputs a "commitment" vk_1 of a secret key $\mathbf{k}$. However, a malicious prover may choose the commitment that does not correspond to any secret key. In this case, we can no longer argue soundness. To avoid the problem, we rely on a non-compact NIZK to prove the well-formedness of the commitment. Since the size of the circuit for checking the well-formedness is independent of the size of the circuit for computing the relation to be proven, this does not harm the compactness of the proof. We finally remark that the construction we explained so far is still slightly different from the one we give in Sec. 3.2. There, we change the scheme so that the prover provides the proof of knowledge of σ instead of sending σ as part of the proof in the clear. While our scheme is secure without this change, this makes it easier to extend our construction to the UC-secure setting.

The proof size of the resulting CRS-NIZK is $|C| + \mathsf{poly}(\kappa)$ since our HEC only supports $\mathbf{NC}^1$ and thus we have to expand the witness to the concatenation

of all values corresponding to each wire of the circuit verifying the relation to make the verification of the relation be done in $\mathbf{NC}^1$. On the other hand, if the relation can be verified in $\mathbf{NC}^1$ from the beginning, then the expansion is not needed and the proof size is as small as $|w| + \mathsf{poly}(\kappa)$.

Interestingly, our CRS-NIZK can also be seen as a variant of the UC-NIZK recently proposed by Cohen, shelat, and Wichs [23]. The differences from their scheme are (1) an HTDF is replaced with an HEC, (2) a witness is encrypted by SKE of which key is committed by a HEC instead of the witness itself, and (3) one-time signatures are omitted. If we are to construct a UC-NIZK in the adaptive non-erasure setting as is done in [23], the modifications (2) and (3) are no longer applicable, but (1) is still applicable. Based on this observation, we obtain a UC-NIZK for $\mathbf{NC}^1$ in the adaptive non-erasure setting with a similar proof size to that of [23] based on a HEC instead of a HTDF. A caveat of our construction is that the scheme only supports NP languages verifiable in $\mathbf{NC}^1$ whereas their scheme supports all of NP (verifiable by a polynomial-size circuit). On the other hand, our abstraction as HEC instead of HTDF enables us to instantiate the scheme based on a pairing assumption instead of lattices. In particular, it seems difficult to construct HTDF based on a pairing assumption.

Compact DV-NIZKs based on Pairing-Free Groups. Here, we explain our constructions of compact DV-NIZKs. Actually, we give a generic compiler to convert any non-compact DV-NIZK to a compact one additionally assuming the existence of PKE and $\mathbf{NC}^1$-decryptable SKE with additive ciphertext overhead. In this overview, we discuss a specific instantiation based on the CDH assumption in pairing-free groups.

The starting point of our constructions is the recent construction of compact NIZKs in the preprocessing model (PP-NIZKs) by Katsumata et al. [53] based on inner-product functional encryptions (IPFE) [1].[7] PP-NIZK is a relaxation of (CRS, DV, DP)-NIZK where both the prover and the verifier are given proving and verification keys, respectively, which should be hidden from each other. Katsumata et al. first constructed a context-hiding homomorphic MAC for arithmetic circuits by adding the context-hiding property to the non-context-hiding homomorphic MAC of Catalano and Fiore [19] by using an IPFE. They then plugged the context-hiding homomorphic MAC into the generic conversion by Kim and Wu [57] to obtain PP-NIZKs.[8] Recall that in the PP-NIZK construction of Kim and Wu, a prover key consists of an SKE key $\mathbf{k}$ and a signature $\boldsymbol{\sigma}$ on $\mathbf{k}$, and a verification key consists of a verification key vk of a homomorphic MAC scheme. The reason why their scheme is PP-NIZK and not DV-NIZK is that a prover

[7] Actually, their construction is based on a variant of IPFE called IPFE on exponent (expIPFE). We note that their construction works with standard IPFE. They used the notion of expIPFE instead of IPFE for making it possible to instantiate the scheme based on the DDH-based scheme by Agrawal, Libert, and Stehlé [4].

[8] Kim and Wu [57] showed that if one uses their generic conversion on homomorphic MACs instead of homomorphic signatures, it would result in PP-NIZKs instead of DP-NIZKs.

has to obtain a signature $\boldsymbol{\sigma}$ on $\mathbf{k}$ which should be generated by a trusted third party who has the corresponding signing key sk.[9] Similarly to the case of our CRS-NIZK explained in the previous section, we observe the following fact. If one can choose $\boldsymbol{\sigma}$ and vk in the reverse order, that is, if one can first choose the signature $\boldsymbol{\sigma}$, and then define vk so that $\boldsymbol{\sigma}$ is a valid signature on $\mathbf{k}$, then we could modify the scheme to be a DV-NIZK by letting the prover choose $\mathbf{k}$ and $\boldsymbol{\sigma}$ on its own. Below, we observe that the homomorphic MAC of Katsumata et al. [53] indeed has this property. To explain this, we first recall the structure of their homomorphic MAC.

In their homomorphic MAC scheme, a verification key vk (which is also a signing key) consists of $s \xleftarrow{\$} \mathbb{Z}_p^*$, $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^\ell$ and a decryption key of an IPFE corresponding to the vector $(s, ..., s^D) \in \mathbb{Z}_p^D$ where p is a sufficiently large prime, ℓ is the message length, and D is the degree of the arithmetic circuits supported by the homomorphic MAC scheme.[10] A signature on $\mathbf{k}$ is defined to be $\boldsymbol{\sigma} := (\mathbf{r} - \mathbf{k}) \cdot s^{-1} \mod p$. From the form of $\boldsymbol{\sigma}$, we can see that for any fixed $\mathbf{k}$ and s, one can set $\boldsymbol{\sigma}$ and $\mathbf{r}$ in the reverse order, that is, one can first pick $\boldsymbol{\sigma}$ and then set $\mathbf{r} := \mathbf{k} + \boldsymbol{\sigma} \cdot s \mod p$.

Going back to the construction of NIZK, this structure enables us to get close to DV-NIZK. Namely, a prover can now choose $\mathbf{k}$ and $\boldsymbol{\sigma}$ by itself, and it no longer needs any proving key generated by a trusted third party. However, there is an important problem still remaining on how the verifier gets to know $\mathbf{r} = \mathbf{k} + \boldsymbol{\sigma} \cdot s \mod p$, which is required for verification. Recall that $\mathbf{r}$ was part of the private verification key of the PP-NIZK of Kim and Wu. If s is given to a prover, then we cannot rely on unforgeability of the homomorphic MAC to prove soundness, and if the prover sends $\mathbf{k}$ and $\boldsymbol{\sigma}$ in the clear, then we cannot rely on the security of SKE to prove zero-knowledge. Therefore the prover has to transmit $\mathbf{r} = \mathbf{k} + \boldsymbol{\sigma} \cdot s \mod p$ to the verifier without knowing s nor revealing $\mathbf{k}$ and $\boldsymbol{\sigma}$ to the verifier. We observe that this task can be done by using IPFE. Namely, we give a secret key corresponding to the vector $(1, s)$ of IPFE to the verifier as a part of his verification key, and a prover encrypts vectors (k_i, σ_i) for each $i \in [\ell]$ where k_i and σ_i are the i-th entry of $\mathbf{k}$ and $\boldsymbol{\sigma}$, respectively, and sends the ciphertexts as a part of the proof. Then a verifier can obtain $\mathbf{r} = \mathbf{k} + \boldsymbol{\sigma} \cdot s \mod p$ by simply decrypting the IPFE ciphertexts with his decryption key.

Though the above idea seems to work at first glance, there is a problem that was also addressed in [53]. Namely, since a standard security notion of IPFE does not consider a malicious encryptor, an adversary may generate a malformed ciphertext whose decryption result is perfectly under his control, which breaks soundness. To prevent such an attack, Katsumata et al. [53] required a property called an *extractability* for an IPFE, which means that one can extract a corresponding message from any possibly malformed ciphertext if it does not decrypt to $\perp$. They then showed that the DDH-based IPFE scheme of Agrawal, Libert, and Stehlé [4] can be used as an extractable IPFE. However, unfortunately,

[9] In a homomorphic MAC, we can let $\mathsf{sk} := \mathsf{vk}$ since both are kept private.

[10] We remark that we cannot include the master secret key of IPFE in vk since the context-hiding property should hold even against the verifier who sees vk.

we will not be able to simply plug in the extractable IPFE of Agrawal et al. into our DV-NIZK. This is because the IPFE of Agrawal et al. embeds the message into the exponent of a group element, and forces one to compute the discrete logarithm to decrypt. Therefore, unless we can be sure that the exponent will be small, the IPFE of Agrawal et al. is difficult to use. Here, the reason why the PP-NIZK of Katsumata et al. [53] did not face any issue with this somewhat awkward decryption algorithm was because the verification algorithm only consisted of checking whether the decryption result is equal to a certain value, which could be tested in the exponent, using the verification key $(s, \mathbf{r})$. However, in our case, the verifier must first decrypt $\mathbf{r}$ using the IPFE secret key corresponding to the vector $(1, s)$ to recover $\mathbf{r}$, and only then it can run the internal verification algorithm of [53] using the pair $(s, \mathbf{r})$. Notably, the verifier would have to solve the discrete logarithm for a random value in $\mathbb{Z}_p$ to recover the piece $\mathbf{r}$ of the verification key used in the PP-NIZK of Katsumata et al. However, obviously, there is no way to compute this efficiently. Therefore, in this work, we must take a different approach. Concretely, instead of relying on the extractability of IPFE, we require a prover to provide a proof that he has honestly generated ciphertexts by using another (non-compact) DV-NIZK. Here, since the validity check of IPFE ciphertexts can be done with computational complexity independent of the size of the language the prover really wants to prove, we can use a non-compact DV-NIZK for this part while keeping the whole proof size compact. In summary, we can convert the PP-NIZK of [53] to a DV-NIZK by adding ℓ IPFE ciphertexts along with their validity proof whose sizes are $\mathsf{poly}(\kappa)$. Since the proof size of the PP-NIZK of [53] is $|C| + \mathsf{poly}(\kappa)$, the proof size of the resulting DV-NIZK is also $|C| + \mathsf{poly}(\kappa)$. Moreover, we note that single-key secure IPFE suffices for the above construction of DV-NIZK. Since single-key secure functional encryption for all polynomial-sized functions exist under the existence of PKE [41] and DV-NIZK for all of **NP** exists under the CDH assumption on a pairing-free group [24,53,75], we can instantiate the above DV-NIZK based on the CDH assumption on a pairing-free group.[11] Finally, we note that by using the idea of the compact homomorphic MAC based on the ℓ-CDHI assumption by Catalano and Fiore [19], we can further reduce the proof size to be $|w| + \mathsf{poly}(\kappa)$ in the case when the language to be proven is computable in $\mathbf{NC}^1$.

Generic Construction of Prover-Efficient NIZK from LFE. To achieve prover-efficient NIZKs, we use laconic function evaluation (LFE) recently introduced by Quach, Wee, and Wichs [76]. LFE schemes are defined for a class of circuits $\mathcal{C}$. We can generate a short digest of circuit $C \in \mathcal{C}$ from a CRS and the circuit C. Anybody can then generate a ciphertext ct of a message m from the CRS, the digest, and m. Finally, anybody can decrypt the ciphertext to $C(m)$ using the ciphertext ct and the circuit C. Here, the security of LFE imposes that

[11] One may wonder why we only need CDH though [53] assumed DDH. Recall that the DDH in their construction comes from the necessity of an extractable expIPFE. We show that this can be replaced with any IPFE and DV-NIZK both of which exist under the CDH assumption based on the same idea as explained above.

the ciphertext ct leaks no additional information other than the value $C(m)$. The attractive feature of LFE is that the size of the CRS, digest, ciphertext ct, and the running time of the encryption algorithm are all strictly smaller than the size of the circuits in $\mathcal{C}$.

Our design idea is to impose the computation of the circuit C computing the NP-relation on the verifier by using LFE. Specifically, we put a digest of C (and a CRS of LFE) in the CRS of our NIZK. The prover then computes an LFE ciphertext of message (x, w) where x is a statement and w is its witness using the digest of C. A verifier can check the validity of the statement by decrypting the ciphertext with C. By the security of LFE, the verifier obtains nothing beyond $C(x, w)$, hence, zero-knowledge of our NIZK follows naturally. Furthermore, by the efficiency property of LFE, the running time of the prover is smaller than the size of C. However, this basic idea is not yet sufficient. This is because a cheating prover may not honestly compute an LFE ciphertext of the message (x, w) and may possibly break soundness of our NIZK. To overcome this issue, a prover must generate not only an LFE ciphertext of (x, w) but also a NIZK proof to prove that the prover honestly generated the LFE ciphertext of (x, w) with the CRS of LFE and the digest of C. We point out that this additional NIZK proof does not harm prover efficiency since the additional statement which the prover must prove is independent of the size of the circuit C owing to the feature of LFE. In particular, we can check the validity of the ciphertext by computing the encryption circuit of LFE whose size is independent of the size of C.

Using any non-prover-efficient NIZK for **NP** as building block and instantiating the LFE scheme by the sub-exponential security of LWE assumption with sub-exponential modulus-to-noise ratio, we obtain a prover-efficient CRS-NIZK for **NP** whose prover running time is $\mathsf{poly}(\kappa, |x|, |w|, d)$, where d is the depth of the circuit C computing the **NP** relation. In particular, the prover running time is independent of $|C|$. In fact, we can further reduce the prover running time to be as small as $\tilde{O}(|x| + |w|) \cdot \mathsf{poly}(\kappa, d)$ where the dependence of the statement x and witness w size is only quasi-linear if we further use the following two assumptions (1) the prover running time of the underlying NIZK is linear in the size of the circuit that computes the **NP** relation, that is, $|C| \cdot \mathsf{poly}(\kappa)$ (2) a natural variant of the above LWE assumption introduced by Quach et al. [76], called the *adaptive* LWE assumption. Note that the assumption we make on the underlying NIZK is not that strong, and in particular, we can use the NIZK of Groth, Ostrovsky, and Sahai [48].

1.4 Related Works

Other than CRS and DV-NIZKs, which have been the main interest of this paper, there are other variants of NIZKs. One is PP-NIZK and the other is DP-NIZK as we briefly mentioned in Sec. 1.3. Similarly to DV-NIZKs, due to the extra secret information shared by the prover and/or verifier, the soundness (resp. zero-knowledge) property of (PP, DP)-NIZKs may be compromised after verifying (resp. proving) multiple statements. In fact most of the PP or DP-NIZKs [29,55,59,26,25,50] are known only to be secure for bounded statements. The

first multi-theorem PP and DP-NIZKs (that are not a trivial downgrade of CRS-NIZKs) where given by Kim and Wu [57] who proposed a generic construction of them via homomorphic MACs and homomorphic signatures, respectively. Since homomorphic signatures were implied by lattice-based assumptions [42], this implied the first DP-NIZKs based on lattices. Subsequently, Katsumata et al. [53] constructed a homomorphic signature based on the CDHER assumption and a homomorphic MAC based on the DDH assumption over pairing-free groups, and thus constructed DP and PP-NIZKs relative to those assumptions. One attractive feature of the NIZKs of Kim and Wu [57] and Katsumata et al. [53] is that the proof size are compact: the DP-NIZK of [57] has proof size $|w| + \mathsf{poly}(\kappa, d)$ and the (PP, DP)-NIZK of [53] have proof size $|C| + \mathsf{poly}(\kappa)$, where d is the depth of the circuit C computing the **NP** relation.

2 Homomorphic Equivocal Commitment

2.1 Definition

We introduce a new primitive which we call homomorphic equivocal commitment (HEC), which can be seen as a relaxed variant of HTDF defined by Gorbunov et al. [42]. A HEC scheme with message space $\mathcal{X}$, randomness space $\mathcal{R}$, and a randomness distribution $\mathcal{D}_{\mathcal{R}}$ over $\mathcal{R}$ for a circuit class $\mathcal{C} = \{C : \mathcal{X} \to \mathcal{Z}\}$ consists of PPT algorithms $(\mathsf{HEC.Setup}, \mathsf{HEC.Commit}, \mathsf{HEC.Open}, \mathsf{HEC.Eval}^{in}, \mathsf{HEC.Eval}^{out}, \mathsf{HEC.Verify})$.

$\mathsf{HEC.Setup}(1^\kappa)$: The setup algorithm takes as input the security parameter 1^κ and outputs a public parameter pp, an evaluation key ek, and a master secret key msk.

$\mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}; R)$: The commit algorithm takes as input a public parameter pp and a message $\mathbf{x} \in \mathcal{X}$ along with a randomness $R \in \mathcal{R}$, and outputs a commitment com. When we omit R to denote $\mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x})$, we mean that R is chosen according to the distribution $\mathcal{D}_{\mathcal{R}}$.

$\mathsf{HEC.Open}(\mathsf{msk}, (\mathbf{x}, R), \mathbf{x}')$: The open algorithm takes as input a master secret key msk, a message $\mathbf{x} \in \mathcal{X}$, a randomness $R \in \mathcal{R}$, and a fake message $\mathbf{x}' \in \mathcal{X}$, and outputs a fake randomness $R' \in \mathcal{R}$.

$\mathsf{HEC.Eval}^{in}(\mathsf{ek}, C, \mathbf{x}, R)$: The inner evaluation algorithm takes as input an evaluation key ek, a circuit $C \in \mathcal{C}$, a message $\mathbf{x} \in \mathcal{X}$, and a randomness $R \in \mathcal{R}$, and outputs a proof π.

$\mathsf{HEC.Eval}^{out}(\mathsf{ek}, C, \mathsf{com})$: The outer evaluation algorithm is a deterministic algorithm that takes as input an evaluation key ek, a circuit $C \in \mathcal{C}$, and a commitment com, and outputs an evaluated commitment $\mathsf{com}_{\mathsf{eval}}$.

$\mathsf{HEC.Verify}(\mathsf{pp}, \mathsf{com}_{\mathsf{eval}}, z, \pi)$: The verification algorithm takes as input a public parameter pp, an evaluated commitment $\mathsf{com}_{\mathsf{eval}}$, a message $z \in \mathcal{Z}$, and a proof π, and outputs $\top$ if the proof is valid and $\bot$ otherwise.

Evaluation Correctness. For all $\kappa \in \mathbb{Z}$, $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}) \xleftarrow{\$} \mathsf{HEC.Setup}(1^\kappa)$, $\mathbf{x} \in \mathcal{X}$, $R \in \mathcal{R}$, $\mathsf{com} := \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}; R)$, $C \in \mathcal{C}$, $\pi \xleftarrow{\$} \mathsf{HEC.Eval}^{in}(\mathsf{msk}, C, \mathbf{x}, R)$, and $\mathsf{com}_{\mathsf{eval}} := \mathsf{HEC.Eval}^{out}(\mathsf{ek}, C, \mathsf{com})$, we have

$$\Pr[\mathsf{HEC.Verify}(\mathsf{pp}, \mathsf{com}_{\mathsf{eval}}, C(\mathbf{x}), \pi) = \top] = 1.$$

Distributional Equivalence of Open. We have

$$\{(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}, \mathbf{x}, R, \mathsf{com})\} \overset{\text{stat}}{\approx} \{(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}, \mathbf{x}, R', \mathsf{com}')\}$$

where $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}) \overset{\$}{\leftarrow} \mathsf{HEC.Setup}(1^\kappa)$, $(\mathbf{x}, \overline{\mathbf{x}}) \in \mathcal{X}^2$ are arbitrary random variables that may depend on $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk})$, $R \overset{\$}{\leftarrow} \mathcal{D}_\mathcal{R}$, $\mathsf{com} := \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}; R)$, $\overline{R} \overset{\$}{\leftarrow} \mathcal{D}_\mathcal{R}$, $\mathsf{com}' := \mathsf{HEC.Commit}(\mathsf{pp}, \overline{\mathbf{x}}; \overline{R})$, and $R' \overset{\$}{\leftarrow} \mathsf{HEC.Open}(\mathsf{msk}, (\overline{\mathbf{x}}, \overline{R}), \mathbf{x})$.

Computational Binding for Evaluated Commitment. For all PPT adversary $\mathcal{A}$,

$$\Pr\left[\begin{array}{r|r} \mathsf{HEC.Verify}(\mathsf{pp}, \mathsf{com}_{\mathsf{eval}}, z^*, \pi^*) = \top & (\mathsf{pp}, \mathsf{ek}, \mathsf{msk}) \overset{\$}{\leftarrow} \mathsf{HEC.Setup}(1^\kappa), \\ z^* \neq C(\mathbf{x}) & (\mathbf{x}, R, C, z^*, \pi^*) \overset{\$}{\leftarrow} \mathcal{A}(\mathsf{pp}, \mathsf{ek}), \\ & \mathsf{com} := \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}; R) \\ & \mathsf{com}_{\mathsf{eval}} := \mathsf{HEC.Eval}^{out}(\mathsf{ek}, C, \mathsf{com}) \end{array}\right] \leq \mathsf{negl}(\kappa).$$

Efficient Committing. There exists a polynomial poly such that for all $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}) \overset{\$}{\leftarrow} \mathsf{HEC.Setup}(1^\kappa)$, $\mathbf{x} \in \mathcal{X}$, $R \in \mathcal{R}$, the running time of $\mathsf{com} := \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}; R)$ is bounded by $|\mathbf{x}| \cdot \mathsf{poly}(\kappa)$.

Efficient Verification (optional). There exists a polynomial poly such that for all $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}) \overset{\$}{\leftarrow} \mathsf{HEC.Setup}(1^\kappa)$, $\mathbf{x} \in \mathcal{X}$, $R \in \mathcal{R}$, $\mathsf{com} := \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}; R)$, $C \in \mathcal{C}$, $\pi \overset{\$}{\leftarrow} \mathsf{HEC.Eval}^{in}(\mathsf{ek}, C, \mathbf{x}, R)$, $\mathsf{com}_{\mathsf{eval}} := \mathsf{HEC.Eval}^{out}(\mathsf{ek}, C, \mathsf{com})$, and $z \in \mathcal{Z}$, we have $|\pi| \leq \mathsf{poly}(\kappa)$ and $|\mathsf{com}_{\mathsf{eval}}| \leq \mathsf{poly}(\kappa)$ and the running time of $\mathsf{HEC.Verify}(\mathsf{pp}, \mathsf{com}_{\mathsf{eval}}, z, \pi)$ is at most $\mathsf{poly}(\kappa)$. We remark that poly does not depend on C.

Context-Hiding (optional). There exists a PPT simulator $\mathsf{HEC.ProofSim}$ such that for all $\kappa \in \mathbb{N}$, $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}) \overset{\$}{\leftarrow} \mathsf{HEC.Setup}(1^\kappa)$, $\mathbf{x} \in \mathcal{X}$, $C \in \mathcal{C}$, $R \in \mathcal{R}$, and $\mathsf{com} := \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}; R)$, we have

$$\{\pi \overset{\$}{\leftarrow} \mathsf{HEC.Eval}^{in}(\mathsf{ek}, C, \mathbf{x}, R))\} \overset{\text{stat}}{\approx} \{\pi' \overset{\$}{\leftarrow} \mathsf{HEC.ProofSim}(\mathsf{msk}, \mathsf{com}, C, C(\mathbf{x})))\}$$

where the probability is only over the randomness used by the algorithms $\mathsf{HEC.Eval}^{in}$ and $\mathsf{HEC.ProofSim}$.

Remark 2.1. We can generically convert any HEC scheme to a context-hiding one by using any statistical CRS-NIZK scheme. Namely, instead of directly using π as an output of the inner evaluation algorithm, it outputs a NIZK proof for the statement that there exists π that passes the verification.

Remark 2.2. The following properties immediately follow from the distributional equivalence of open.

Equivocality. We have

$$\Pr[\mathsf{HEC.Commit}(\mathsf{pp}, \overline{\mathbf{x}}; \overline{R}) \neq \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}; R)] = \mathsf{negl}(\kappa)$$

where $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}) \xleftarrow{\$} \mathsf{HEC.Setup}(1^\kappa)$, $(\mathbf{x}, \overline{\mathbf{x}}) \in \mathcal{X}^2$ are arbitrary random variables that may depend on $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk})$, $\overline{R} \xleftarrow{\$} \mathcal{D}_{\mathcal{R}}$, and $R \xleftarrow{\$} \mathsf{HEC.Open}(\mathsf{msk}, (\overline{\mathbf{x}}, \overline{R}), \mathbf{x})$.

Hiding. We have

$$\{\mathsf{pp}, \mathsf{ek}, \mathsf{com} \xleftarrow{\$} \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x})\} \overset{\text{stat}}{\approx} \{\mathsf{pp}, \mathsf{ek}, \mathsf{com}' \xleftarrow{\$} \mathsf{HEC.Commit}(\mathsf{pp}, \mathbf{x}')\},$$

where $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}) \xleftarrow{\$} \mathsf{HEC.Setup}(1^\kappa)$ and $(\mathbf{x}, \mathbf{x}') \in \mathcal{X}^2$ are arbitrary random variables that may depend on $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk})$. We say that a scheme is computationally hiding if the above two distributions are computationally indistinguishable.

Remark 2.3. If we require neither efficient verification nor context-hiding, then there is a trivial construction of HEC based on any equivocal commitment. Namely, we can just set $\mathsf{com}_{\mathsf{eval}} := C \| \mathsf{com}$ and $\pi := (\mathbf{x}, R)$. The verification algorithm can verify them by checking if com is a commitment of $\mathbf{x}$ with randomness R and $z = C(\mathbf{x})$ holds. On the other hand, if we require either of efficient verification or context hiding, then there does not seem to be such a trivial solution.[12] This is reminiscent of the similar situation for fully homomorphic encryption where a scheme without compactness nor function privacy is trivial to construct but a scheme with either of them is non-trivial [34].

2.2 Constructions of HEC

Here, we show that we can construct an HEC scheme based on a non-static falsifiable pairing assumption called the (n, m)-computational Diffie-Hellman exponent ratio (CDHER) assumption [53].

(n, m)-Computational Diffie-Hellman Exponent and Ratio Assumption. Let BGGen be a PPT algorithm that on input 1^κ returns a description $\mathcal{G} = (\mathbb{G}, \mathbb{G}_T, p, g, e(\cdot, \cdot))$ of symmetric pairing groups where $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of prime order p, g is the generator of $\mathbb{G}$, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an efficiently computable (non-degenerate) bilinear map.

Definition 2.1 ((n, m)-Computational Diffie-Hellman Exponent and Ratio Assumption). *[53] Let BGGen be a group generator and $n := n(\kappa) = \mathsf{poly}(\kappa)$, $m := m(\kappa) = \mathsf{poly}(\kappa)$. We say that the* (n, m)-decisional Diffie-Hellman exponent and ratio (CDHER) assumption *holds with respect to BGGen, if for all PPT adversaries $\mathcal{A}$, we have*

$$\Pr\left[\mathcal{A}(\mathcal{G}, \Psi) \to e(g, g)^{sa^{m+1}}\right] = \mathsf{negl}(\kappa)$$

[12] As remarked in Remark 2.1, we can convert the trivial construction to a context-hiding one additionally assuming a statistical CRS-NIZK for all of **NP**. Though this is less interesting than schemes with efficient verification, we do not consider it a "trivial solution" since the existence of a statistical CRS-NIZK is an additional assumption to an equivocal commitment.

where $\mathcal{G} = (\mathbb{G}, \mathbb{G}_T, p, g, e(\cdot,\cdot)) \xleftarrow{\$} \mathsf{BGGen}(1^\lambda)$, $s, a, b_1, \ldots, b_n, c_1, \ldots c_n \xleftarrow{\$} \mathbb{Z}_p^*$, *and*

$$\Phi := \begin{pmatrix} \left\{g^{a^j}\right\}_{j\in[m]}, & \{g^{c_i}\}_{i\in[n]}, & \left\{g^{a^j/b_i}\right\}_{\substack{i\in[n],j\in[2m]\\ j\neq m+1}}, & \left\{g^{a^{m+1}c_{i'}/b_ic_i}\right\}_{i,i'\in[n],i\neq i'}, \\ & \{g^{ac_i}\}_{i\in[n]}, & \left\{g^{a^j/b_ic_i}\right\}_{i\in[n],j\in[2m+1]}, & \left\{g^{a^jc_{i'}/b_i}\right\}_{i,i'\in[n],j\in[m]}, \\ g^s, & \{g^{sb_i}\}_{i\in[n]}, & \left\{g^{sa^{m+1}b_i/b_{i'}c_{i'}}\right\}_{i,i'\in[n],i\neq i'}, & \left\{g^{sa^jb_i/b_{i'}}\right\}_{\substack{i,i'\in[n],j\in[m]\\ i\neq i'}} \end{pmatrix}.$$

Katsumata et al. showed that the CDHER assumption holds on the generic group model introduced by Shoup [83].

Construction of HEC based on CDHER Assumption. We show the following theorem.

Theorem 2.1. *If the* (n, m)*-CDHER assumption holds on a pairing group for all* $n = \mathsf{poly}(\kappa)$ *and* $m = \mathsf{poly}(\kappa)$*, then there exists an HEC scheme that supports* $\boldsymbol{NC}^1$ *that satisfies evaluation correctness, distributional equivalence of open, computational binding for evaluated commitments, efficient committing, efficient verification, and context-hiding.*

The construction is obtained by a tweak to the homomorphic signature scheme by Katsumata et al. [53] as explained in Sec. 1.3. The full description of the construction and its security proof can be found in the full version.

In the full version, we also show that we can construct a context-hiding HEC scheme *without efficient verification* based on the weaker CDH assumption on a pairing group. Though this is not useful for constructing compact NIZKs as is done in Sec. 3, this can be used for constructing (non-compact) context-hiding homomorphic signature scheme as shown in the full version.

3 Compact CRS-NIZK from HEC

Here, we give a construction of a compact CRS-NIZK scheme based on any non-compact CRS-NIZK scheme and HEC with efficient verification. If we instantiate the construction with the HEC given in Sec. 2.2, then the proof size of the resulting CRS-NIZK scheme is $|C| + \mathsf{poly}(\kappa)$. Moreover, if the relation supported by the scheme is verifiable in $\mathbf{NC}^1$, then the proof size is $|w| + \mathsf{poly}(\kappa)$.

3.1 Extractable CRS-NIZK

First, we define extractability for CRS-NIZK, which is needed for our construction of compact CRS-NIZK scheme. We note that the extractability defined here is a mild property, and we can convert any CRS-NIZK scheme to the one with extractability if we additionally assume the existence of PKE as shown in Lemma 3.1.

An extractable CRS-NIZK is a CRS-NIZK with an additional deterministic algorithm $\mathsf{Extract}$ which takes as input a randomness r_{Setup} used in Setup and a

proof π, and outputs a witness w that satisfies the following.

Extractability. For all PPT adversary $\mathcal{A}$, we have

$$\Pr\left[\begin{array}{r|l} \mathsf{Verify}(\mathsf{crs}, x, \pi) = \top & \mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\kappa) \\ (x, w) \notin \mathcal{R} & (x, \pi) \xleftarrow{\$} \mathcal{A}(\mathsf{crs}) \\ & w \xleftarrow{\$} \mathsf{Extract}(r_{\mathsf{Setup}}, \pi) \end{array}\right] \leq \mathsf{negl}(\kappa).$$

where r_{Setup} is the randomness used in Setup to generate crs.

The following lemma is easy to prove. The proof can be found in the full version.

Lemma 3.1. *If there exist CRS-NIZK for all of* ***NP*** *and a CPA-secure PKE scheme, then there exists CRS-NIZK for all of* ***NP*** *with extractability.*

3.2 Construction of Compact CRS-NIZK

Before describing the construction, we prepare some building blocks and notations.

- Let $\mathcal{L}$ be an **NP** language defined by a relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$. Let $n(\kappa)$ and $m(\kappa)$ be any fixed polynomials. Let C be a circuit that computes the relation $\mathcal{R}$ on $\{0,1\}^n \times \{0,1\}^m$, i.e., for $(x, w) \in \{0,1\}^n \times \{0,1\}^m$, we have $C(x, w) = 1$ if and only if $(x, w) \in \mathcal{R}$.
- Let $\Pi_{\mathsf{SKE}} = (\mathsf{SKE.KeyGen}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ be a symmetric key encryption (SKE) scheme with ciphertext space $\mathcal{CT}$ and key space $\{0,1\}^\ell$.

In the following, for $x \in \{0,1\}^n$ and $\mathsf{ct} \in \mathcal{CT}$, we define the function

$$f_{x,\mathsf{ct}}(K) := C(x, \mathsf{SKE.Dec}(K, \mathsf{ct})).$$

- Let $\Pi_{\mathsf{HEC}} = (\mathsf{HEC.Setup}, \mathsf{HEC.Commit}, \mathsf{HEC.Open}, \mathsf{HEC.Eval}^{in}, \mathsf{HEC.Eval}^{out}, \mathsf{HEC.Verify})$ be a HEC scheme with the message space that contains $\{0,1\}^\ell$ and randomness space $\mathcal{R}$ on which a distribution $\mathcal{D}_{\mathcal{R}}$ is defined. We need the HEC scheme to support a function class containing $\{f_{x,\mathsf{ct}}\}_{x \in \{0,1\}^n, \mathsf{ct} \in \mathcal{CT}}$.
- Let $\Pi_{\mathsf{CRSNIZK}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ be an extractable CRS-NIZK for the language corresponding to the relation $\widetilde{\mathcal{R}}$ defined below:
  $((\mathsf{pp}, \mathsf{com}, \mathsf{com}_{\mathsf{eval}}), (K, R, \pi_{\mathsf{HEC}})) \in \widetilde{\mathcal{R}}$ if and only if the followings are satisfied:
  1. $K \in \{0,1\}^\ell$,
  2. $\mathsf{HEC.Commit}(\mathsf{pp}, K; R) = \mathsf{com}$,
  3. $\mathsf{HEC.Verify}(\mathsf{pp}, \mathsf{com}_{\mathsf{eval}}, 1, \pi_{\mathsf{HEC}}) = \top$.

  We note that extractable CRS-NIZK for all of **NP** exists assuming (non-extractable) CRS-NIZK for all of **NP** and CPA secure PKE as shown in Lemma 3.1.

The CRS-NIZK $\Pi'_{\mathsf{CRSNIZK}} = (\mathsf{Setup}', \mathsf{Prove}', \mathsf{Verify}')$ for $\mathcal{L}$ is described as follows.

$\mathsf{Setup}'(1^\kappa)$**:** This algorithm generates $\mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\kappa)$ and $(\mathsf{pp}, \mathsf{ek}, \mathsf{msk}), \xleftarrow{\$} \mathsf{HEC.Setup}(1^\kappa)$. It outputs a common reference string $\mathsf{crs}' = (\mathsf{crs}, \mathsf{pp}, \mathsf{ek})$.

$\mathsf{Prove}'(\mathsf{crs}', x, w)$**:** This algorithm aborts if $\mathcal{R}(x, w) = 0$. Otherwise it parses $(\mathsf{crs}, \mathsf{pp}, \mathsf{ek}) \leftarrow \mathsf{crs}'$, picks $K \xleftarrow{\$} \mathsf{SKE.KeyGen}(1^\kappa)$ and $R \xleftarrow{\$} \mathcal{D}_{\mathcal{R}}$, computes $\mathsf{ct} \xleftarrow{\$} \mathsf{SKE.Enc}(K, w)$, generates $\mathsf{com} := \mathsf{HEC.Commit}(\mathsf{pp}, K; R)$, $\pi_{\mathsf{HEC}} \xleftarrow{\$} \mathsf{HEC.Eval}^{in}(\mathsf{ek}, f_{x,\mathsf{ct}}, K, R)$, $\mathsf{com}_{\mathsf{eval}} := \mathsf{HEC.Eval}^{out}(\mathsf{ek}, f_{x,\mathsf{ct}}, \mathsf{com})$, and $\pi_{\mathsf{NIZK}} \xleftarrow{\$} \mathsf{Prove}(\mathsf{crs}, (\mathsf{pp}, \mathsf{com}, \mathsf{com}_{\mathsf{eval}}), (K, R, \pi_{\mathsf{HEC}}))$, and outputs a proof $\pi' := (\mathsf{ct}, \mathsf{com}, \pi_{\mathsf{NIZK}})$.

$\mathsf{Verify}'(\mathsf{crs}', x, \pi')$**:** This algorithm parses $(\mathsf{crs}, \mathsf{pp}, \mathsf{ek}) \leftarrow \mathsf{crs}'$ and $(\mathsf{ct}, \mathsf{com}, \pi_{\mathsf{NIZK}}) \leftarrow \pi'$, computes $\mathsf{com}_{\mathsf{eval}} := \mathsf{HEC.Eval}^{out}(\mathsf{ek}, f_{x,\mathsf{ct}}, \mathsf{com})$, and outputs $\top$ if $\mathsf{Verify}(\mathsf{crs}, (\mathsf{pp}, \mathsf{com}, \mathsf{com}_{\mathsf{eval}}), \pi_{\mathsf{NIZK}}) = \top$, and outputs $\bot$ otherwise.

Correctness. Suppose that $(\mathsf{ct}, \mathsf{com}, \pi_{\mathsf{NIZK}})$ is an honestly generated proof on $(x, w) \in \mathcal{R}$. Then we have $\mathsf{ct} \xleftarrow{\$} \mathsf{SKE.Enc}(K, w)$ and $\mathsf{com} = \mathsf{HEC.Commit}(\mathsf{pp}, K; R)$ with some K and R. By the correctness of Π_{SKE}, we have $f_{x,\mathsf{ct}}(K) = 1$, and by the correctness of Π_{HEC}, we have $\mathsf{HEC.Verify}(\mathsf{pp}, \mathsf{com}_{\mathsf{eval}}, 1, \pi_{\mathsf{HEC}}) = \top$ where we generate $\mathsf{com}_{\mathsf{eval}} := \mathsf{HEC.Eval}^{out}(\mathsf{ek}, f_{x,\mathsf{ct}}, \mathsf{com})$ and $\pi_{\mathsf{HEC}} \xleftarrow{\$} \mathsf{HEC.Eval}^{in}(\mathsf{ek}, f_{x,\mathsf{ct}}, K, R)$. Since we have $((\mathsf{pp}, \mathsf{com}, \mathsf{com}_{\mathsf{eval}}), (K, R, \pi_{\mathsf{HEC}})) \in \widetilde{\mathcal{R}}$, if we generate $\pi_{\mathsf{NIZK}} \xleftarrow{\$} \mathsf{Prove}(\mathsf{crs}, (\mathsf{pp}, \mathsf{com}, \mathsf{com}_{\mathsf{eval}}), (K, R, \pi_{\mathsf{HEC}}))$, then we have $\mathsf{Verify}(\mathsf{crs}, (\mathsf{pp}, \mathsf{com}, \mathsf{com}_{\mathsf{eval}}), \pi_{\mathsf{NIZK}}) = \top$ by the correctness of Π_{CRSNIZK}.

Security. The security of NIZK' is stated as follows. The proofs can be found in the full version.

Theorem 3.1 (Soundness.). *If Π_{CRSNIZK} satisfies extractability and* HEC *satisfies computational binding for evaluated commitment, then Π'_{CRSNIZK} satisfies computational soundness.*

Theorem 3.2 (Zero-knowledge.). *If Π_{CRSNIZK} satisfies zero-knowledge,* HEC *is computationally hiding,*[13] *and* SKE *is CPA secure, then Π'_{CRSNIZK} satisfies zero-knowledge.*

3.3 Instantiations

Here, we discuss that by appropriately instantiating Π_{CRSNIZK}, we can achieve compact proof size. In particular, we consider instantiating the HEC scheme with our construction in Sec. 2.2. Since our HEC scheme only supports $\mathbf{NC}^1$ circuits, we have to ensure that $f_{x,\mathsf{ct}}$ is computable in $\mathbf{NC}^1$. For ensuring this, we use the fact that any efficiently verifiable relation can be verified in $\mathbf{NC}^1$ at the cost of making the witness size as large as the size of a circuit that verifies the relation (e.g., [32]). This is done by considering all values corresponding to all gates when computing the circuit on input (x, w) to be the new witness. In addition, we use an SKE scheme whose decryption circuit is in $\mathbf{NC}^1$ with additive ciphertext overhead (i.e., the ciphertext length is the message length plus $\mathsf{poly}(\kappa)$) and the key size $\ell = \kappa$, which exists under the CDH assumption [53]. Then $f_{x,\mathsf{ct}}$ is computable in $\mathbf{NC}^1$ for every x and ct. In this case, we have that $|\mathsf{ct}| \leq |C| + \mathsf{poly}(\kappa)$. In order to bound the length of the proof π', we also bound

[13] Recall that the computational hiding (or even statistical hiding) follows from the distributional equivalence of open.

$|\mathsf{com}|$ and $|\pi_{\mathsf{NIZK}}|$. By the efficient committing property of HEC, $|\mathsf{com}|$ and the size of the circuit computing $\mathsf{HEC.Commit}$ is bounded by $|K| \cdot \mathsf{poly}(\kappa) \leq \mathsf{poly}(\kappa)$. Furthermore, by the efficient verification property of HEC, the size of the circuit computing $\mathsf{HEC.Verify}$ is bounded by $\mathsf{poly}(\kappa)$. Therefore, the size of the circuit computing $\widetilde{\mathcal{R}}$ is bounded by $\mathsf{poly}(\kappa)$, which implies that $|\pi_{\mathsf{NIZK}}|$ is bounded by $\mathsf{poly}(\kappa)$ as well (even if Π_{CRSNIZK} is non-compact). To sum up, we have that the proof size of Π_{CRSNIZK} is $|C| + \mathsf{poly}(\kappa)$. Moreover, if we only consider a relation computable in $\mathbf{NC}^1$ in the first place, then we need not expand the witness, and the proof size can be further reduced to be $|w| + \mathsf{poly}(\kappa)$. Finally, we remark that (non-compact) CRS-NIZK for all of $\mathbf{NP}$ exists under the CDH assumption on a pairing group [17,3], which in particular holds under the CDHER assumption. In summary, we obtain the following corollary.

Corollary 3.1. *If the CDHER assumption holds on a pairing group, then there exists CRS-NIZK for all of* $\boldsymbol{NP}$ *with proof size* $|C| + \mathsf{poly}(\kappa)$*. Moreover, if the corresponding relation is computable in* $\boldsymbol{NC}^1$*, then the proof size is* $|w| + \mathsf{poly}(\kappa)$*.*

Variant with Sublinear Proof Size. Katsumata et al. [53] showed that their DP-NIZK achieves sublinear proof size i.e., $|w| + |C|/\log\kappa + \mathsf{poly}(\kappa)$ if C is a leveled circuit [12] whose gates are divided into L levels, and all incoming wires to a gate of level $i+1$ come from gates of level i. Exactly the same idea can be applied to our CRS-NIZK to achieve sublinear proof size. More detailed explanation can be found in the full version. Namely, we obtain the following corollary:

Corollary 3.2. *If the CDHER assumption holds on a pairing group, then there exists CRS-NIZK for all* $\boldsymbol{NP}$ *languages whose corresponding relation is computable by a leveled circuit with proof size* $|w| + |C|/\log\kappa + \mathsf{poly}(\kappa)$*.*

Variant with UC-Security. We can modify the above scheme to satisfy the UC security in the non-erasure adaptive setting. Namely, we can show the following theorem. The proof can be found in the full version.

Theorem 3.3. *If the DLIN assumption and the CDHER assumption hold in a bilinear group, then for any relation* $\mathcal{R}$ *that is computable in* $\boldsymbol{NC}^1$*, there exists a UC-secure NIZK scheme for* $\mathcal{R}$ *tolerating an adaptive, malicious adversary.*

4 Compact DV-NIZK

4.1 Preliminaries

Here, we recall some lemmas which are implicit or explicit in [53].

Lemma 4.1. *(Implicit in [53]) Let* C *be a boolean circuit that computes a relation* $\mathcal{R}$ *on* $\{0,1\}^n \times \{0,1\}^m$*, i.e., for* $(x,w) \in \{0,1\}^n \times \{0,1\}^m$*, we have* $C(x,w) = 1$ *if and only if* $(x,w) \in \mathcal{R}$*, and* p *be an integer larger than* $|C|$*. Then there exists a deterministic algorithm* $\mathsf{Exp}_{C,x}$ *and an arithmetic circuit* $\tilde{C}$ *on* $\mathbb{Z}_p$ *with degree at most* 3 *such that we have*

- $|\mathsf{Exp}_{C,x}(w)| = |C(x,\cdot)|$ *for all* $w \in \{0,1\}^m$.
- *If* $C(x,w) = 1$, *then we have* $\tilde{C}(x, \mathsf{Exp}_{C,x}(w)) = 1 \mod p$.
- *For any* $x \in \{0,1\}^n$, *if there does not exist* $w \in \{0,1\}^m$ *such that* $C(x,w) = 1$, *then there does not exist* $w' \in \{0,1\}^{|C(x,\cdot)|}$ *such that* $\tilde{C}(x,w') = 1 \mod p$

Lemma 4.2. *([53]) There exists a deterministic polynomial-time algorithm* $\mathsf{Coefficient}$ *that satisfies the following: for any* $p \in \mathbb{N}$, *arithmetic circuit* f *over* $\mathbb{Z}_p$ *of degree* D, $\mathbf{x} = (x_1, ..., x_\ell) \in \mathbb{Z}_p^\ell$ *and* $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_\ell) \in \mathbb{Z}_p^\ell$, $\mathsf{Coefficient}(1^D, p, f, \mathbf{x}, \boldsymbol{\sigma})$ *outputs* $(c_1, ..., c_D) \in \mathbb{Z}_p^D$ *such that*

$$f(\sigma_1 Z + x_1, ..., \sigma_\ell Z + x_\ell) = f(x_1, ..., x_\ell) + \sum_{j=1}^{D} c_j Z^j \mod p. \tag{1}$$

where Z *is an indeterminate.*

4.2 Construction

Here, we give a generic construction of compact DV-NIZK. Namely, we construct DV-NIZK with the proof size $|C| + \mathsf{poly}(\kappa)$ from any (non-compact) DV-NIZK, SKE scheme whose decryption circuit is in $\mathbf{NC}^1$ with additive ciphertext overhead, and PKE scheme. First, we prepare notations and the building blocks.

- Let $\mathcal{L}$ be an $\mathbf{NP}$ language defined by a relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$. Let $n(\kappa)$ and $m(\kappa)$ be any fixed polynomials. Let C be a circuit that computes the relation $\mathcal{R}$ on $\{0,1\}^n \times \{0,1\}^m$, i.e., for $(x,w) \in \{0,1\}^n \times \{0,1\}^m$, we have $C(x,w) = 1$ if and only if $(x,w) \in \mathcal{R}$. Let $\mathsf{Exp}_{C,x}$ and $\tilde{C}$ be as defined in Lemma 4.1.
- Let $\Pi_{\mathsf{IPFE}} = (\mathsf{IPFE.Setup}, \mathsf{IPFE.KeyGen}, \mathsf{IPFE.Enc}, \mathsf{IPFE.Dec})$ be an adaptively single-key secure IPFE scheme with a prime modulus $p > |C|$. Such an IPFE scheme can be constructed from any PKE scheme [41].
- Let $\Pi_{\mathsf{SKE}} = (\mathsf{SKE.KeyGen}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ be a CPA-secure symmetric key encryption scheme over a ciphertext space $\mathcal{CT}$ and a key space $\{0,1\}^\ell$ with additive ciphertext overhead (i.e., the ciphertext size is the message size plus $\mathsf{poly}(\kappa)$) whose decryption algorithm is computed in $\mathbf{NC}^1$. Especially, the decryption circuit can be expressed by an arithmetic circuit over $\mathbb{Z}_p$ of degree $\mathsf{poly}(\kappa)$. We note that such an SKE scheme exists under the CDH assumption [53].
- For $x \in \{0,1\}^n$ and $\mathsf{ct} \in \mathcal{CT}$, we define the function $f_{x,\mathsf{ct}}(K) := \tilde{C}(x, \mathsf{SKE.Dec}(K, \mathsf{ct}))$. Let D be the maximal degree of $f_{x,\mathsf{ct}}$ (as a multivariate polynomial). Since $\tilde{C}$'s degree is at most 3 and $\mathsf{SKE.Dec}(\cdot, \mathsf{ct})$'s degree is $\mathsf{poly}(\kappa)$, we have $D = \mathsf{poly}(\kappa)$ (which especially does not depend on $|C|$).
- Let $\Pi_{\mathsf{DVNIZK}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ be DV-NIZK for the language corresponding to the relation $\tilde{\mathcal{R}}$ defined below:

  $$\left((\mathsf{pp}_{\mathsf{IPFE}}, \{\mathsf{ct}^i_{\mathsf{IPFE}}\}_{i\in[\ell]}, \mathsf{pp}'_{\mathsf{IPFE}}, \mathsf{ct}'_{\mathsf{IPFE}}), \ \left(\{(K_i, \sigma_i, R_i)\}_{i\in[\ell]}, (c_1, ..., c_D, R')\right) \right) \in \tilde{\mathcal{R}}$$

  if and only if the following conditions are satisfied:

1. For all $i \in [\ell]$, $K_i \in \{0,1\}$,
2. For all $i \in [\ell]$, $\mathsf{IPFE.Enc}(\mathsf{pp}_{\mathsf{IPFE}}, (K_i, \sigma_i); R_i) = \mathsf{ct}^i_{\mathsf{IPFE}}$,
3. $\mathsf{IPFE.Enc}(\mathsf{pp}'_{\mathsf{IPFE}}, (c_1, ..., c_D); R') = \mathsf{ct}'_{\mathsf{IPFE}}$.

The DV-NIZK $\Pi'_{\mathsf{DVNIZK}} = (\mathsf{Setup}', \mathsf{Prove}', \mathsf{Verify}')$ for $\mathcal{L}$ is described as follows.

$\mathsf{Setup}'(1^\kappa)$**:** This algorithm picks $s \xleftarrow{\$} \mathbb{Z}_p^*$ and generates $(\mathsf{crs}, k_{\mathsf{V}}) \xleftarrow{\$} \mathsf{Setup}(1^\kappa)$, $(\mathsf{pp}_{\mathsf{IPFE}}, \mathsf{msk}_{\mathsf{IPFE}}) \xleftarrow{\$} \mathsf{IPFE.Setup}(1^\kappa, 1^2)$, $(\mathsf{pp}'_{\mathsf{IPFE}}, \mathsf{msk}'_{\mathsf{IPFE}}) \xleftarrow{\$} \mathsf{IPFE.Setup}(1^\kappa, 1^D)$, $\mathsf{sk}_{\mathsf{IPFE}} \xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{msk}_{\mathsf{IPFE}}, (1, s))$, and $\mathsf{sk}'_{\mathsf{IPFE}} \xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{msk}'_{\mathsf{IPFE}}, (s, ..., s^D))$. It outputs a common reference string $\mathsf{crs}' := (\mathsf{crs}, \mathsf{pp}_{\mathsf{IPFE}}, \mathsf{pp}'_{\mathsf{IPFE}})$ and a verifier key $k'_{\mathsf{V}} := (k_{\mathsf{V}}, s, \mathsf{sk}_{\mathsf{IPFE}}, \mathsf{sk}'_{\mathsf{IPFE}})$.

$\mathsf{Prove}'(\mathsf{crs}', x, w)$**:** This algorithm aborts if $(x, w) \notin \mathcal{R}$. Otherwise it parses $(\mathsf{crs}, \mathsf{pp}_{\mathsf{IPFE}}, \mathsf{pp}'_{\mathsf{IPFE}}) \leftarrow \mathsf{crs}'$, picks $K \xleftarrow{\$} \mathsf{SKE.KeyGen}(1^\kappa)$ and $\sigma_i \xleftarrow{\$} \mathbb{Z}_p$ for $i \in [\ell]$, and generates $\mathsf{ct}_{\mathsf{SKE}} \xleftarrow{\$} \mathsf{SKE.Enc}(K, \mathsf{Exp}_{C,x}(w))$ and $(c_1, ..., c_D) \leftarrow \mathsf{Coefficient}(1^D, p, f_{x,\mathsf{ct}_{\mathsf{SKE}}}, K = (K_1, ..., K_\ell), (\sigma_1, ..., \sigma_\ell))$. Then it generates $\mathsf{ct}^i_{\mathsf{IPFE}} := \mathsf{IPFE.Enc}(\mathsf{pp}_{\mathsf{IPFE}}, (K_i, \sigma_i); R_i)$ for $i \in [\ell]$ (where R_i is the randomness used by the encryption algorithm), $\mathsf{ct}'_{\mathsf{IPFE}} := \mathsf{IPFE.Enc}(\mathsf{pp}'_{\mathsf{IPFE}}, (c_1, ..., c_D); R')$ (where R' is the randomness used by the encryption algorithm), and $\pi \xleftarrow{\$} \mathsf{Prove}(\mathsf{crs}, (\mathsf{pp}_{\mathsf{IPFE}}, \{\mathsf{ct}^i_{\mathsf{IPFE}}\}_{i\in[\ell]}, \mathsf{pp}'_{\mathsf{IPFE}}, \mathsf{ct}'_{\mathsf{IPFE}}), (\{(K_i, \sigma_i, R_i)\}_{i\in[\ell]}, (c_1, ..., c_D, R')))$ and outputs a proof $\pi' := (\pi, \mathsf{ct}_{\mathsf{SKE}}, \{\mathsf{ct}^i_{\mathsf{IPFE}}\}_{i\in[\ell]}, \mathsf{ct}'_{\mathsf{IPFE}})$.

$\mathsf{Verify}'(\mathsf{crs}', k'_{\mathsf{V}}, x, \pi')$**:** This algorithm parses $(\mathsf{crs}, \mathsf{pp}_{\mathsf{IPFE}}, \mathsf{pp}'_{\mathsf{IPFE}}) \leftarrow \mathsf{crs}'$, $(k_{\mathsf{V}}, s, \mathsf{sk}_{\mathsf{IPFE}}, \mathsf{sk}'_{\mathsf{IPFE}}) \leftarrow k'_{\mathsf{V}}$, and $(\pi, \mathsf{ct}_{\mathsf{SKE}}, \{\mathsf{ct}^i_{\mathsf{IPFE}}\}_{i\in[\ell]}, \mathsf{ct}'_{\mathsf{IPFE}}) \leftarrow \pi'$, computes $r_i \xleftarrow{\$} \mathsf{IPFE.Dec}(\mathsf{pp}_{\mathsf{IPFE}}, \mathsf{ct}^i_{\mathsf{IPFE}}, \mathsf{sk}_{\mathsf{IPFE}})$ for $i \in [\ell]$ and $t \xleftarrow{\$} \mathsf{IPFE.Dec}(\mathsf{pp}'_{\mathsf{IPFE}}, \mathsf{ct}'_{\mathsf{IPFE}}, \mathsf{sk}'_{\mathsf{IPFE}})$, and outputs $\top$ if we have $\mathsf{Verify}(\mathsf{crs}, (\mathsf{pp}_{\mathsf{IPFE}}, \{\mathsf{ct}^i_{\mathsf{IPFE}}\}_{i\in[\ell]}, \mathsf{pp}'_{\mathsf{IPFE}}, \mathsf{ct}'_{\mathsf{IPFE}}), \pi) = \top$ and

$$f_{x,\mathsf{ct}_{\mathsf{SKE}}}(r_1, ..., r_\ell) = 1 + t \mod p,$$

and outputs $\perp$ otherwise.

Correctness. Suppose that $(\pi, \mathsf{ct}_{\mathsf{SKE}}, \{\mathsf{ct}^i_{\mathsf{IPFE}}\}_{i\in[\ell]}, \mathsf{ct}'_{\mathsf{IPFE}})$ is an honestly generated proof on $(x, w) \in \mathcal{R}$. Then it is clear that we have $\mathsf{Verify}(\mathsf{crs}, (\mathsf{pp}_{\mathsf{IPFE}}, \{\mathsf{ct}^i_{\mathsf{IPFE}}\}_{i\in[\ell]}, \mathsf{pp}'_{\mathsf{IPFE}}, \mathsf{ct}'_{\mathsf{IPFE}}), \pi) = \top$ by the way of generating the proof and the correctness of Π_{DVNIZK}. By the way of generating $(\{\mathsf{ct}^i_{\mathsf{IPFE}}\}_{i\in[\ell]}, \mathsf{ct}'_{\mathsf{IPFE}})$ and correctness of Π_{IPFE}, we have $r_i = K_i + \sigma_i s \mod p$ for $i \in [\ell]$ and $t = \sum_{j\in[D]} c_j s^j$ where r_i and t are generated as in the verification. Since we have $f_{x,\mathsf{ct}_{\mathsf{SKE}}}(K_1 + \sigma_1 Z, ..., K_\ell + \sigma_\ell Z) = 1 + \sum_{j\in[D]} c_j Z^j$ for an indeterminate Z by the correctness of Π_{SKE} and Lemma 4.2, we have $f_{x,\mathsf{ct}_{\mathsf{SKE}}}(r_1, ..., r_\ell) = 1 + t$ by substituting s for Z.

Proof Size. First, we remark that the relation $\tilde{\mathcal{R}}$ can be verified by a circuit whose size is a fixed polynomial in $(\kappa, \ell, \log p, D)$ that does not depend on $|C|$. Moreover, we have $|\mathsf{Exp}_{C,x}(w)| = |C(x, \cdot)| \leq |C|$ for all $w \in \{0,1\}^m$ by Lemma 4.1. Then we have $|\pi| = \mathsf{poly}(\kappa, \ell, \log p, D)$, $|\mathsf{ct}_{\mathsf{SKE}}| = |C(x, \cdot)| + \mathsf{poly}(\kappa)$, $|\mathsf{ct}^i_{\mathsf{IPFE}}| = \mathsf{poly}(\kappa, \log p)$, and $|\mathsf{ct}'_{\mathsf{IPFE}}| = \mathsf{poly}(\kappa, \log p, D)$. By setting $\ell = \kappa$ and $p = 2^{O(\kappa)}$ and remarking that $D = \mathsf{poly}(\kappa)$, we have $|\pi'| = |C(x, \cdot)| + \mathsf{poly}(\kappa) \leq |C| + \mathsf{poly}(\kappa)$.

Security. The security of our scheme Π'_{DVNIZK} is stated as follows. The proofs are similar to the security proof for PP-NIZK by Katsumata et al. [53], and thus given in the full version.

Theorem 4.1 (Soundness.). *If Π_{DVNIZK} satisfies statistical (resp. computational) soundness and $p = \kappa^{\omega(1)}$, then Π'_{DVNIZK} satisfies statistical (resp. computational) soundness.*

Theorem 4.2 (Zero-knowledge). *If SKE is CPA secure, Π_{IPFE} is adaptively single-key secure, and Π_{DVNIZK} satisfies zero-knowledge, then Π'_{DVNIZK} satisfies zero-knowledge.*

Instantiation. The above construction can be instantiated based on the CDH assumption on a pairing-free group since

- An adaptively single-key secure IPFE scheme exists under any PKE scheme [41], and there exists a PKE scheme based on the CDH assumption.
- An SKE scheme whose decryption circuit is in $\mathbf{NC}^1$ with additive ciphertext overhead exists under the CDH assumption [53].
- DV-NIZK for all of **NP** exists under the CDH assumption [24,53,75]

Therefore we obtain the following corollary.

Corollary 4.1. *If the CDH assumption holds on a pairing-free group, then there exists DV-NIZK for all of **NP** with proof size $|C| + \mathsf{poly}(\kappa)$.*

Variant with Sublinear Proof Size. Similarly to the case of CRS-NIZK as discussed in Sec. 3.3, we can make the proof size of the above DV-NIZK sublinear in $|C|$ if C is a leveled circuit. More detailed explanation can be found in the full version. Namely, we obtain the following corollary:

Corollary 4.2. *If the CDH assumption holds on a pairing-free group, then there exists DV-NIZK for all **NP** languages whose corresponding relation is computable by a leveled circuit with proof size $|w| + |C|/\log\kappa + \mathsf{poly}(\kappa)$.*

Variant with Shorter Proof Size for $\mathbf{NC}^1$ Relations. We can further reduce the proof size to $|w| + \mathsf{poly}(\kappa)$ if the relation to prove is computable in $\mathbf{NC}^1$ and we additionally assume ℓ-computational Diffie-Hellman inversion (CDHI) assumption [66,19].

Theorem 4.3. *If the ℓ-CDHI assumption holds for all $\ell = \mathsf{poly}(\kappa)$, then there exists DV-NIZK for all relations for all **NP** languages whose corresponding relation is computable in $\mathbf{NC}^1$ with proof size $|w| + \mathsf{poly}(\kappa)$.*

The construction and security proofs can be found in the full version.

5 CRS-NIZK with Efficient Prover From Laconic Function Evaluation

In this section, we present a NIZK proof system where a prover is efficient, that is, the running time of a prover is smaller than the size of circuit that computes the relation. We use laconic function evaluation to achieve our NIZK proof system.

Before describing the construction, we prepare some building blocks and notations.

– Let $\mathcal{L}$ be an **NP** language defined by a relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$. Let $n(\kappa)$ and $m(\kappa)$ be any fixed polynomials. Let C be a circuit that computes the relation $\mathcal{R}$ on $\{0,1\}^n \times \{0,1\}^m$, i.e., for $(x,w) \in \{0,1\}^n \times \{0,1\}^m$, we have $C(x,w) = 1$ if and only if $(x,w) \in \mathcal{R}$
– Let $\mathsf{LFE} = (\mathsf{LFE.crsGen}, \mathsf{LFE.Compress}, \mathsf{LFE.Enc}, \mathsf{LFE.Dec})$ be a LFE scheme whose function class $\mathcal{C}$ is the class of all circuits with $\mathsf{params} = (1^k, 1^d)$ consisting of the input size k and the depth d of the circuits and contains $\{C\}$ that computes the relation $\mathcal{R}$ for NP-complete language.
– Let $\Pi_{\mathsf{CRSNIZK}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ be a CRS-NIZK for the language corresponding to the relation $\tilde{\mathcal{R}}$ defined below:

$$((x, \mathsf{lfe.crs}, \mathsf{digest}_C, \mathsf{lfe.ct}), (w,r)) \in \tilde{\mathcal{R}} \iff \mathsf{LFE.Enc}(\mathsf{lfe.crs}, \mathsf{digest}_C, (x,w); r) = \mathsf{lfe.ct}\,.$$

The CRS-NIZK $\Pi'_{\mathsf{CRSNIZK}} = (\mathsf{Setup}', \mathsf{Prove}', \mathsf{Verify}')$ for $\mathcal{L}$ is described as follows.

$\mathsf{Setup}'(1^\kappa)$**:** This algorithm generates $\mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\kappa)$ and $\mathsf{lfe.crs} \xleftarrow{\$} \mathsf{LFE.crsGen}(1^\kappa, \mathsf{params})$. It generates $\mathsf{digest}_C := \mathsf{LFE.Compress}(\mathsf{lfe.crs}, C)$. It outputs a common reference string $\mathsf{crs}' = (\mathsf{crs}, \mathsf{lfe.crs}, \mathsf{digest}_C)$.

$\mathsf{Prove}'(\mathsf{crs}', x, w)$**:** This algorithm aborts if $\mathcal{R}(x,w) = 0$. Otherwise it parses $(\mathsf{crs}, \mathsf{lfe.crs}, \mathsf{digest}_C) \leftarrow \mathsf{crs}'$, generates $\mathsf{lfe.ct} := \mathsf{LFE.Enc}(\mathsf{lfe.crs}, \mathsf{digest}_C, (x,w); r)$ where r is the randomness for $\mathsf{LFE.Enc}$ and $\pi_{\mathsf{NIZK}} \xleftarrow{\$} \mathsf{Prove}(\mathsf{crs}, (x, \mathsf{lfe.crs}, \mathsf{digest}_C, \mathsf{lfe.ct}), (w,r))$. It outputs a proof $\pi' := (\mathsf{lfe.ct}, \pi_{\mathsf{NIZK}})$.

$\mathsf{Verify}'(\mathsf{crs}', x, \pi')$**:** This algorithm parses $(\mathsf{crs}, \mathsf{lfe.crs}, \mathsf{digest}_C) \leftarrow \mathsf{crs}'$, $(\mathsf{lfe.ct}, \pi_{\mathsf{NIZK}}) \leftarrow \pi'$, and computes $t := \mathsf{Verify}(\mathsf{crs}, (x, \mathsf{lfe.crs}, \mathsf{digest}_C, \mathsf{lfe.ct}), \pi_{\mathsf{NIZK}})$. If $t = \bot$ or $0 \xleftarrow{\$} \mathsf{LFE.Dec}(\mathsf{lfe.crs}, C, \mathsf{lfe.ct})$, then outputs $\bot$. Otherwise, outputs $\top$.

Completeness. By the completeness of Π_{CRSNIZK}, the proof π_{NIZK} in an honestly generated proof π' passes the verification of Π_{CRSNIZK}. That is, it holds that $\mathsf{Verify}(\mathsf{crs}, (x, \mathsf{lfe.crs}, \mathsf{digest}_C, \mathsf{lfe.ct}), \pi_{\mathsf{NIZK}}) = \top$. By the correctness of LFE, it holds that $1 = C(x,w) \xleftarrow{\$} \mathsf{LFE.Dec}(\mathsf{lfe.crs}, C, \mathsf{lfe.ct})$ with probability 1. Thus, the completeness follows.

Prover Efficiency. First, we remark that the relation $\tilde{\mathcal{R}}$ can be verified by a circuit whose size is $|\mathsf{LFE.Enc}|$ since the relation is about the validity of LFE ciphertexts. The running time of Prove' is the sum of those of $\mathsf{LFE.Enc}$ and Prove. We defer concrete efficiency analysis until Sec. 5.1 since the running time depends on instantiations of $\mathsf{LFE.Enc}$ and Prove.

Security. The security of the scheme is stated as follows. See the full version for the proofs.

Theorem 5.1 (Soundness). *Π'_{CRSNIZK} is computationally/statistically sound if Π_{CRSNIZK} is computationally/statistically sound, respectively.*

Theorem 5.2 (Zero-Knowledge). *Π'_{CRSNIZK} is computational zero-knowledge if Π_{CRSNIZK} is zero-knowledge and LFE is adaptively secure.*

5.1 Instantiations

We can consider two cases since there are two instantiations of adaptively secure LFE.

1. (Under sub-exponential security of the LWE assumption with sub-exponential modulus-to-noise ratio): By the result of [76], it holds that $|\mathsf{lfe.crs}| = \mathsf{poly}(\kappa, |x|, |w|, d)$, $|\mathsf{digest}_C| = \mathsf{poly}(\kappa)$, $|\mathsf{lfe.ct}| = \mathsf{poly}(\kappa, |x|, |w|, d)$, and the running time of $\mathsf{LFE.Enc}$ is $\mathsf{poly}(\kappa, |x|, |w|, d)$ where d is the depth of C since the input length of C is $|x| + |w|$. In this case, we use a NIZK whose prover running time is $\mathsf{poly}(\tilde{C}, \kappa)$ where $\tilde{C}$ is a circuit that computes the relation $\tilde{\mathcal{R}}$, which holds for any NIZK. In this case, $\tilde{C}$ just runs $\mathsf{LFE.Enc}$, so it takes $|\mathsf{LFE.Enc}| + \mathsf{poly}(|\mathsf{LFE.Enc}|, \kappa)$ time to generate π_{NIZK}. Thus, the running time of the prover is $\mathsf{poly}(\kappa, |x|, |w|, d)$.
2. (Under the *adaptive* LWE assumption with sub-exponential modulus-to-noise ratio): By the result of [76], it holds that $|\mathsf{lfe.crs}| = (|x| + |w|) \cdot \mathsf{poly}(\kappa, d)$, $|\mathsf{digest}_C| = \mathsf{poly}(\kappa)$, $|\mathsf{lfe.ct}| = \tilde{O}(|x| + |w|) \cdot \mathsf{poly}(\kappa, d)$, and the running time of $\mathsf{LFE.Enc}$ is $\tilde{O}(|x| + |w|) \cdot \mathsf{poly}(\kappa, d)$ where d is the depth of C since the input length of C is $|x| + |w|$. In this case, we use a NIZK whose prover running time is $|\tilde{C}| \cdot \mathsf{poly}(\kappa)$. An example of such a NIZK is the NIZK by Groth et al. [48]. By using the efficiency of Groth et al. NIZK, it takes $|\mathsf{LFE.Enc}| + |\mathsf{LFE.Enc}| \cdot \mathsf{poly}(\kappa)$ time to generate π_{NIZK}. Thus, the running time of the prover is $\tilde{O}(|x| + |w|) \cdot \mathsf{poly}(\kappa, d) \cdot \mathsf{poly}(\kappa) = \tilde{O}(|x| + |w|) \cdot \mathsf{poly}(\kappa, d)$.

Therefore, we obtain the following two corollaries.

Corollary 5.1. *If a CRS-NIZK scheme for all of* $\boldsymbol{NP}$ *exists and the sub-exponentially secure LWE assumption with sub-exponential modulus-to-noise ratio holds, then there exists a CRS-NIZK scheme for all of* $\boldsymbol{NP}$ *whose prover running time is* $\mathsf{poly}(\kappa, |x|, |w|, d)$.

Corollary 5.2. *If the DLIN assumption in a bilinear group and the adaptive LWE assumption with sub-exponential modulus-to-noise ratio hold, then there exists a CRS-NIZK scheme for all of* $\boldsymbol{NP}$ *whose prover running time is* $\tilde{O}(|x| + |w|)\mathsf{poly}(\kappa, d)$.

Acknowledgement. We thank anonymous reviewers of Crypto 2019 for their helpful comments. The first and the third authors were supported by JST CREST Grant Number JPMJCR19F6. The third author was supported by JSPS KAKENHI Grant Number 16K16068.

References

1. Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. *PKC 2015*, pages 733–751. 2015.
2. Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. *Journal of Cryptology*, 29(2):363–421, 2016.

3. Hamza Abusalah. Generic instantiations of the hidden bits model for non-interactive zero-knowledge proofs for NP, 2013. Master's thesis, RWTH-Aachen University.
4. Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. *CRYPTO 2016, Part III*, pages 333–362. 2016.
5. Paul W Beame, Stephen A Cook, and H James Hoover. Log depth circuits for division and related problems. *SIAM Journal on Computing*, 15(4):994–1003, 1986.
6. Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. *EUROCRYPT 2003*, pages 614–629. 2003.
7. Mihir Bellare and Moti Yung. Certifying permutations: Noninteractive zero-knowledge based on any trapdoor permutation. *Journal of Cryptology*, 9(3):149–166.
8. Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. *ITCS 2012*, pages 326–349. 2012.
9. Nir Bitansky and Omer Paneth. ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. *TCC 2015, Part II*, pages 401–427.
10. Nir Bitansky, Omer Paneth, and Daniel Wichs. Perfect structure on the edge of chaos - trapdoor permutations from indistinguishability obfuscation. *TCC 2016-A, Part I*, pages 474–502. 2016.
11. Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). *20th ACM STOC*, pages 103–112. 1988.
12. Elette Boyle, Niv Gilboa, and Yuval Ishai. Breaking the circuit size barrier for secure computation under DDH. *CRYPTO 2016, Part I*, pages 509–539. 2016.
13. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, 2000.
14. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. *42nd FOCS*, pages 136–145. 2001.
15. Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-Shamir: from practice to theory. *STOC 2019 (to appear)*, 2019.
16. Ran Canetti, Yilei Chen, Leonid Reyzin, and Ron D. Rothblum. Fiat-Shamir and correlation intractability from strong KDM-secure encryption. *EUROCRYPT 2018, Part I*, pages 91–122. 2018.
17. Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. *Journal of Cryptology*, 20(3):265–294, 2007.
18. Ran Canetti and Amit Lichtenberg. Certifying trapdoor permutations, revisited. *TCC 2018, Part I*, pages 476–506. 2018.
19. Dario Catalano and Dario Fiore. Practical homomorphic message authenticators for arithmetic circuits. *Journal of Cryptology*, 31(1):23–59, 2018.
20. Pyrros Chaidos and Geoffroy Couteau. Efficient designated-verifier non-interactive zero-knowledge proofs of knowledge. *EUROCRYPT 2018, Part III*, pages 193–221.
21. Pyrros Chaidos and Jens Groth. Making sigma-protocols non-interactive without random oracles. *PKC 2015*, pages 650–670. 2015.
22. David Chaum and Eugène van Heyst. Group signatures. *EUROCRYPT'91*, pages 257–265. 1991.
23. Ran Cohen, abhi shelat, and Daniel Wichs. Adaptively secure mpc with sublinear communication complexity. *CRYPTO 2019*. 2019.
24. Geoffroy Couteau and Dennis Hofheinz. Designated-verifier pseudorandom generators, and their applications. *EUROCRYPT 2019, Part II*, pages 562–592. 2019.

25. Ronald Cramer and Ivan Damgård. Secret-key zero-knowlegde and non-interactive verifiable exponentiation. *TCC 2004*, pages 223–237. 2004.
26. Ivan Damgård. Non-interactive circuit based proofs and non-interactive perfect zero-knowledge with proprocessing. *EUROCRYPT'92*, pages 341–355. 1993.
27. Ivan Damgård, Nelly Fazio, and Antonio Nicolosi. Non-interactive zero-knowledge from homomorphic encryption. *TCC 2006*, pages 41–59. 2006.
28. George Danezis, Cédric Fournet, Jens Groth, and Markulf Kohlweiss. Square span programs with applications to succinct NIZK arguments. *ASIACRYPT 2014, Part I*, pages 532–550. 2014.
29. Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Non-interactive zero-knowledge with preprocessing. *CRYPTO'88*, pages 269–282. 1990.
30. Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM J. Comput.*, 30(2):391–437, 2000.
31. Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM J. Comput.*, 29(1):1–28, 1999.
32. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM J. Comput.*, 45(3):882–929, 2016.
33. Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. *EUROCRYPT 2013*, pages 626–645. 2013.
34. Craig Gentry. A fully homomorphic encryption scheme, 2009. Ph.D. thesis, Stanford University.
35. Craig Gentry, Jens Groth, Yuval Ishai, Chris Peikert, Amit Sahai, and Adam D. Smith. Using fully homomorphic hybrid encryption to minimize non-interative zero-knowledge proofs. *Journal of Cryptology*, 28(4):820–843, October 2015.
36. Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. *43rd ACM STOC*, pages 99–108. June 2011.
37. Oded Goldreich. Foundations of cryptography: Volume 2, basic applications. 2004.
38. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. *19th ACM STOC*, pages 218–229. 1987.
39. Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, December 1994.
40. Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.
41. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. *CRYPTO 2012*, pages 162–179. 2012.
42. Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. *STOC 2015*, pages 469–477. 2015.
43. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. *IACR Cryptology ePrint Archive*, 2006:309, 2006. Version 20061007:061901.
44. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. *CCS 2006*, pages 89–98.
45. Jens Groth. Short non-interactive zero-knowledge proofs. *ASIACRYPT 2010*, pages 341–358. 2010.
46. Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. *ASIACRYPT 2010*, pages 321–340. 2010.

47. Jens Groth. On the size of pairing-based non-interactive arguments. *EUROCRYPT 2016, Part II*, pages 305–326. 2016.
48. Jens Groth, Rafail Ostrovsky, and Amit Sahai. New techniques for noninteractive zero-knowledge. *J. ACM*, 59(3):11:1–11:35, 2012.
49. Jens Groth and Amit Sahai. Efficient noninteractive proof systems for bilinear groups. *SIAM J. Comput.*, 41(5):1193–1232, 2012.
50. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM J. Comput.*, 39(3):1121–1152.
51. Yael Tauman Kalai, Guy N. Rothblum, and Ron D. Rothblum. From obfuscation to the security of Fiat-Shamir for proofs. *CRYPTO 2017, Part II*, pages 224–251.
52. Shuichi Katsumata. On the untapped potential of encoding predicates by arithmetic circuits and their applications. *ASIACRYPT 2017, Part III*, pages 95–125. 2017.
53. Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Designated verifier/prover and preprocessing NIZKs from Diffie-Hellman assumptions. *EUROCRYPT 2019, Part II*, pages 622–651. 2019.
54. Joe Kilian. On the complexity of bounded-interaction and noninteractive zero-knowledge proofs. *35th FOCS*, pages 466–477. 1994.
55. Joe Kilian, Silvio Micali, and Rafail Ostrovsky. Minimum resource zero-knowledge proofs (extended abstract). *CRYPTO'89*, pages 545–546. 1990.
56. Joe Kilian and Erez Petrank. An efficient noninteractive zero-knowledge proof system for NP with general assumptions. *Journal of Cryptology*, 11(1):1–27, 1998.
57. Sam Kim and David J. Wu. Multi-theorem preprocessing NIZKs from lattices. *CRYPTO 2018, Part II*, pages 733–765. 2018.
58. Sam Kim and David J. Wu. Multi-theorem preprocessing NIZKs from lattices. Cryptology ePrint Archive, Report 2018/272, 2018.
59. Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. *CRYPTO'90*, pages 353–365. 1991.
60. Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. *EUROCRYPT 2011*, pages 568–588. 2011.
61. Yehuda Lindell. How to simulate it - A tutorial on the simulation proof technique. Cryptology ePrint Archive, Report 2016/046, 2016.
62. Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. *TCC 2012*, pages 169–189. 2012.
63. Helger Lipmaa. Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes. *ASIACRYPT 2013, Part I*, pages 41–60.
64. Helger Lipmaa. Optimally sound sigma protocols under DCRA. *FC 2017*, pages 182–203. 2017.
65. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. *EUROCRYPT 2012*, pages 700–718. 2012.
66. Shigeo Mitsunari, Ryuichi Saka, and Masao Kasahara. A new traitor tracing. *IEICE Transactions*, E85-A(2):481–484, 2002.
67. David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. *ACM CCS 98*, pages 59–66. 1998.
68. Moni Naor. On cryptographic assumptions and challenges (invited talk). *CRYPTO 2003*, pages 96–109. 2003.
69. Moni Naor, Benny Pinkas, and Omer Reingold. Distributed pseudo-random functions and KDCs. *EUROCRYPT'99*, pages 327–346. 1999.
70. M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *J. ACM 51(2)*, pages 231-262. 2004.

71. Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. *22nd ACM STOC*, pages 427–437. 1990.
72. Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: nearly practical verifiable computation. *Commun. ACM*, 59(2):103–112, 2016.
73. Rafael Pass, abhi shelat, and Vinod Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. *CRYPTO 2006*, pages 271–289. 2006.
74. Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. *CRYPTO'91*, pages 129–140. 1992.
75. Willy Quach, Ron Rothblum, and Daniel Wichs. Reusable designated-verifier NIZKs for all NP from CDH. *EUROCRYPT 2019, Part II*, pages 593–621. 2019.
76. Willy Quach, Hoeteck Wee, and Daniel Wichs. Laconic function evaluation and applications. *59th FOCS*, pages 859–870. 2018.
77. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, 2009.
78. Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. *ASIACRYPT 2001*, pages 552–565. 2001.
79. Yannis Rouselakis and Brent Waters. New constructions and proof methods for large universe attribute-based encryption. *IACR Cryptology ePrint Archive*, 2012:583, 2012. Version 20140828:060226.
80. Yannis Rouselakis and Brent Waters. Practical constructions and new proof methods for large universe attribute-based encryption. *ACM CCS 13*, pages 463–474. 2013.
81. Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. *40th FOCS*, pages 543–553. 1999.
82. Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. *46th ACM STOC*, pages 475–484. 2014.
83. Victor Shoup. Lower bounds for discrete logarithms and related problems. *EUROCRYPT'97*, pages 256–266. 1997.
84. Carmine Ventre and Ivan Visconti. Co-sound zero-knowledge with public keys. *AFRICACRYPT 09*, pages 287–304. 2009.