SNARGs for NP via Fiat–Shamir in the Plain Model

Ziyi Guan
ziyi.guan@epfl.ch
EPFL

Eylon Yogev
eylon.yogev@biu.ac.il
Bar-Ilan University

Abstract

We study constructions of succinct non-interactive arguments (SNARGs) for NP in the standard model. Specifically, we revisit the seminal Micali transformation (applying Fiat–Shamir to Kilian's protocol), which has traditionally only been analyzed in the random oracle model.

We show that the Micali framework can be successfully instantiated in the standard model by leveraging a new interaction between two primitives: a PCP satisfying a property we term *shadow soundness*, and a vector commitment scheme satisfying *function statistical binding*.

We prove that any language in NP admits a shadow PCP algorithm in the class NC. Combining such a PCP with a vector commitment with function-statistical-binding for NC yields a secure SNARG, under the assumption of sub-exponential hardness of LWE.

Our results refute "universal" attacks on the Micali framework by exhibiting a large family of concrete instantiations of the underlying components for which the transformation is sound.

Keywords: SNARGs, Fiat–Shamir security, vector commitments, PCP

Contents

1 Introduction

Succinct non-interactive arguments (SNARGs) for NP [Mic00] constitute a fundamental primitive in modern cryptography. They allow a prover to convince a verifier of the validity of a statement with a proof whose size is significantly smaller than the time required to verify the statement directly, and ideally, much smaller than the witness itself. This "non-interactive" feature is particularly crucial for real-world applications, enabling scenarios such as delegating computation to untrusted servers, scaling blockchains via rollups, and constructing privacy-preserving cryptocurrencies.

Building on the seminal work of Kilian [Kil92], Micali [Mic00] provided the first construction of a SNARG for NP in the random oracle model (ROM). Since then, there has been tremendous progress in the design of efficient SNARGs. However, the vast majority of efficient constructions are proven secure only within idealized models. Some of these works rely solely on the random oracle model or the generic group model (GGM) to establish security (e.g., [Gro16; BCS16; CY21a; CY21b; ACY23; AHIV17; BBHR18; ZCF24; ACFY24; ACFY25]). Others rely on the random oracle model in addition to other specific cryptographic assumptions (e.g., [BBB+18; AGL+23; AFLN24; BC25]). While these models provide a heuristic rationale for security, they do not guarantee security in the standard model, where the hash function or group is instantiated with concrete algorithms.

A parallel line of work has constructed SNARGs for NP in the common reference string (CRS) model based on various *non-falsifiable* assumptions (e.g., knowledge of exponent assumptions) [Gro10; BCCT12; DFH12; BCCT13; Lip13; GGPR13; BCI+13; BCP13; BISW17; BCC+14; BISW18; ACL+23; CLM23]. In the CRS model, the prover and verifier rely on a trusted reference string (or random string) generated during a setup phase.

SNARGs from falsifiable assumptions. Despite their practical significance and widespread deployment, the theoretical foundations of SNARGs remain somewhat unsatisfactory. For a long time, the existence of SNARGs based solely on falsifiable assumptions remained an open question. The first construction of a SNARG for NP from (sub-exponentially) falsifiable assumptions was provided by Sahai and Waters [SW14], who gave a (non-adaptive) construction based on indistinguishability obfuscation (iO) and one-way functions.[1]

To get around this challenge, an alternative approach is to relax the target setting—either by restricting the class of languages or by weakening the notion being achieved. Along these lines, several works construct SNARGs from standard falsifiable assumptions for P [CJJ22; KVZ21; HJKS22; CGJ+23], P/poly [GZ21; WW23; WW15; BCFL23; WW24b; Wee25a; Wee25b], batch NP [CJJ22; CJJ21; GSWW22; WW22; DGKV22; PP22; KLVW23; CGJ+23; DWW24], monotone policy batch NP [BBK+23; NWW24; NWW25], and NP languages with propositional proofs of non-membership [JKLM25].

Building adaptively-sound SNARGs for NP from falsifiable assumptions is a challenging problem, as any such result must circumvent known black-box separations [GW11; CGKS23]. While recent works have improved the Sahai-Waters construction to achieve adaptive soundness [WW24a; WZ24; WW25], these results still necessitate the full power of iO. Although iO is a powerful primitive, it is considered a heavy assumption, both in terms of the complexity of its construction and the concrete efficiency of its instantiations. Consequently, constructing SNARGs from standard, well-studied assumptions (such as the hardness of lattice problems) without relying on iO or idealized models

[1] Recall that iO alone does not imply one-way functions and must be supplemented with a cryptographic assumption (e.g., one-way functions). However, this additional assumption can sometimes be replaced by a weaker worst-case assumption [KMN+22].

remains a major open problem in the field. Recently, Devadas et. al [DHK+26] constructed a SNARG for NP based on LWE (or other standard assumptions), and a mathematical conjecture about multivariate polynomials over the reals (low-norm Nullstellensatz hypothesis).

The challenge of Fiat–Shamir for arguments. If we allow interactions, the situation is much better. In his seminal work, Kilian [Kil92] showed how to construct succinct *interactive* arguments for NP assuming only the existence of collision-resistant hash functions (CRH); moreover, it was later shown that even multi-collision resistant hash functions suffice [KNY18]. Kilian's protocol works by having the prover construct a PCP for the statement and commit to it using a vector commitment scheme (typically instantiated via a Merkle tree). The verifier then queries specific locations of the PCP, for which the prover provides openings to the commitment.

To remove interaction, Micali [Mic00] proposed applying the Fiat–Shamir transformation to Kilian's protocol. The idea is to replace the verifier's random queries with the output of a cryptographic hash function applied to the prover's commitment. Micali proved that this construction yields a secure SNARG in the ROM. Consequently, a natural approach to obtain a SNARG from falsifiable assumptions is to instantiate this blueprint using a concrete hash function in the standard model.

In recent years, there has been significant progress in instantiating the Fiat–Shamir heuristic in the standard model. A line of work has successfully established hash functions that can securely instantiate Fiat–Shamir for specific classes of interactive protocols based on standard assumptions (e.g., [CCR16; KRR17; CCRR18; HL18; CCH+19; PS19; BKM20; JJ21; HLR21; CJJ22; HJKS22; KLV23]). However, a crucial limitation of these works is that they primarily apply to interactive *proofs* (i.e., protocols with statistical soundness). Kilian's protocol, however, is an interactive *argument* (i.e., it has only computational soundness) due to its reliance on a succinct commitment scheme.

Standard techniques for instantiating Fiat–Shamir generally fail when compiling argument systems into non-interactive counterparts, rendering the instantiation of Micali's construction in the standard model a formidable challenge. Indeed, there is a line of work that highlight the risks of replacing the random oracle with a concrete hash function in an argument scheme, regardless of how the hash function is implemented [CGH04; Bar01; GK03; KRS25; AY25].

Furthermore, Bartusek et al. [BBH+19] identified strong barriers specifically targeting Micali's blueprint. They constructed a specific CRH for which the Micali transformation results in an unsound protocol for *any* concrete instantiation of the Fiat–Shamir hash function. This implies that any successful instantiation of Micali's construction cannot treat the CRH as a generic component. Rather, it must leverage some specific structure of the underlying CRH.

1.1 Our results

We show that the Micali transformation can be successfully instantiated in the standard model under standard assumptions. To achieve this, we rely on a new property of probabilistically checkable proofs which we term ***shadow PCP***. We show that for any language L admitting such a PCP, the Micali blueprint yields a secure argument system.

Central to this concept is the *shadow algorithm* associated with the PCP. Our construction uses a vector commitment scheme that satisfies *function-statistical-binding* for a specific family of functions $\mathcal{F}$. The construction is secure provided that $\mathcal{F}$ is sufficiently expressive to compute the shadow algorithm.

Theorem 1. *Let $\mathcal{F}$ be a function family such that* 3SAT *has a shadow PCP with a shadow algorithm in $\mathcal{F}$, and there is a vector commitment scheme satisfying function-statistical-binding for $\mathcal{F}$. Then, under the sub-exponential hardness of LWE, there exists a non-adaptive SNARG for* 3SAT*.*

We further show that every language in NP has a shadow PCP with a shadow algorithm in NC. This allows us to instantiate the theorem above with a vector commitment schemes that satisfies function-statistical-binding for NC. Thus, we get that, assuming LWE, vector commitment schemes for NC imply SNARGs for NP.

Theorem 2. *Every language in* NP *has a shadow PCP with a shadow algorithm in* NC*.*

As a consequence, vector commitment schemes satisfying function-statistical-binding for NC *and the sub-exponential hardness of LWE imply the existence non-adaptive SNARGs for* 3SAT*.*

Finally, we observe that strong assumptions such as indistinguishability obfuscation (iO) allow us to construct vector commitments with function-statistical-binding for all P/poly. This yields a concrete instantiation of the Micali blueprint in the standard model.

Crucially, however, our construction only requires vector commitments for the class NC. This relaxation suggests that the heavy machinery of iO may be an overkill: constructing vector commitments for NC from simpler primitives would immediately yield SNARGs from those weaker assumptions.

1.2 Refuting universal attacks on the Micali framework

As discussed, a line of work has established "universal" impossibility results for the Fiat–Shamir transformation, showing that for certain protocols, the compiled argument is insecure regardless of the concrete hash function used [BBH+19]. However, such impossibility results generally do not apply to interactive *proofs*. This is because the existence of correlation intractable hash (CIH) functions guarantees that, for a broad class of proofs, a secure instantiation of Fiat–Shamir *does* exist. The mere existence of such a hash function, even if inefficient, serves as a counterexample that rules out any universal attack.

Our work provides a parallel guarantee for the Kilian-Micali *argument* system. By constructing a concrete, secure instantiation of the Micali transformation (via a specific PCP, a vector commitment, and an LWE-based hash), we provide an existential proof that refutes the possibility of universal attacks against this paradigm. Consequently, the Micali framework is not inherently unsound; any successful attack on the transformation must necessarily rely on specific, contrived choices of the underlying components rather than the structure itself.

2 Techniques

We give a high-level overview of our techniques. As mentioned, the general blueprint of the SNARG construction is the work of [Mic00] (see also [CY24]). That is, we use a PCP for NP in which the prover first computes a PCP proof and then commits to it using a specialized vector commitment scheme. The verifier's PCP randomness is derived via a specialized Fiat–Shamir hash. Finally, the prover provides all the answers to the PCP along with a proof of consistency. We will dive into each of these components separately and describe how they interplay. For clarity of exposition, we focus on achieving a sublinear proof size (i.e., $o(n)$), without optimizing for the precise bound stated in our main theorem, and at end of the overview we give more details on how to achieve the precise bound.

To motivate our specific design, we first recall how the Fiat–Shamir transformation is typically instantiated in the standard model for statistically sound protocols. For 3-message interactive *proofs* (protocols with statistical soundness), secure instantiation is achieved using a correlation intractable hash (CIH) function [CCR16; KRR17; CCRR18; HL18; CCH+19; PS19; BKM20; JJ21; HLR21; CJJ22; HJKS22; KLV23]. This works because such protocols induce an underlying ***sparse relation***: for any false statement, the set of "bad" verifier challenges (those that lead to acceptance) is negligibly small. A CIH hash function is simply defined to avoid outputting values within this sparse set.

However, this standard paradigm fails for Kilian's protocol. Because Kilian's protocol is an interactive *argument*, relying on the computational binding of the vector commitment, there is no underlying sparse relation. Since the committed string is only computationally binding and hidden from the verifier, one cannot define a fixed sparse set of bad challenges in the standard information-theoretic sense. Thus, one cannot directly apply the same CIH approach, and some new ideas are necessary. In particular, we need to have some statistical property hiding under the computational commitment.

In this overview, we describe how we "couple" PCPs with our vector commitment scheme to induce a sparse relation, thereby enabling the use of a correlation-intractable hash function to instantiate the Fiat–Shamir transformation for Kilian's protocol. Our approach is inspired by previous works that construct SNARGs for subsets of NP using somewhere statistically binding hash [BBH+19; CJJ21; CJJ22; PP22; DGKV22; KLVW23].

We present the PCP we need in Section 2.1, our VC scheme in Section 2.2, and the final SNARG construction and analyses in Section 2.3.

2.1 PCP shadows

Consider the $\mathsf{3SAT}$ language over n variables. The first component is an information-theoretic PCP construction that satisfies a strong soundness notion we call ***shadow soundness***. A shadow is a digest or projection of a PCP string Π (either valid or malicious) into a short state (of size $o(n)$), denoted by z. This compression is performed by a *randomized* algorithm denoted by F.

The shadow z is designed to predict the value of the verifier's output $\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}, \rho)$. This is achieved via a decider algorithm D that takes the shadow and the PCP randomness ρ as input and outputs a decision bit. The *correctness* property asserts that for any fixed ρ, with high probability (over the randomness of F), it holds that $\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}, \rho) = \mathsf{D}(\mathbb{x}, z, \rho)$. Note that the shadow loses the local properties that the proof Π had. While correctness is the primary property, we additionally require two auxiliary properties: *sparsity*, which implies that D accepts a sparse set of randomness for

instances not in the language; and *consistency*, which ensures that proof strings sharing a local view produce consistent shadows. Formally, we provide the following definition.

Definition 1 (Shadow soundness)**.** *A* $\mathsf{PCP} = (\mathbf{P}, \mathbf{V})$ *has shadow soundness if there exists a function* F *with randomness complexity* $\mathsf{r_F}$ *and a deterministic algorithm* D *(a decider) such that for every instance* $\mathbb{x} \notin L(R)$*, the following holds:*

1. ***Correctness.*** *For any* $\widetilde{\Pi}$*, and verifier randomness* $\rho \in \{0,1\}^{\mathsf{r}}$*:*

$$\Pr\left[\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}, \rho) \neq \mathsf{D}(\mathbb{x}, z, \rho) \;\middle|\; \begin{array}{l} \gamma \leftarrow \{0,1\}^{\mathsf{r_F}} \\ z := \mathsf{F}(\mathbb{x}, \widetilde{\Pi}; \gamma) \end{array}\right] \leq \epsilon_{\mathsf{c}}(\mathbb{x})\,.$$

2. ***Sparsity.*** *There exists an efficiently computable set* Z *where* $\mathsf{Im}(\mathsf{F}(\mathbb{x}, \cdot)) \subseteq Z$ *such that for any shadow state* $z \in Z$*:*

$$\Pr\left[\mathsf{D}(\mathbb{x}, z, \rho) = 1 \mid \rho \leftarrow \{0,1\}^{\mathsf{r}}\right] \leq \epsilon_{\mathsf{s}}(\mathbb{x})\,.$$

3. ***Consistency.*** *For every shadow randomness* $\gamma \in \{0,1\}^{\mathsf{r_F}}$*, verifier randomness* $\rho \in \{0,1\}^{\mathsf{r}}$*,* $\widetilde{\Pi}$ *and* $\widetilde{\Pi}'$ *such that* $\widetilde{\Pi}[\mathsf{Q}] = \widetilde{\Pi}'[\mathsf{Q}]$*, where* Q *is the query set of the PCP verifier* $\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho)$*:*

$$\mathsf{D}(\mathbb{x}, z, \rho) = \mathsf{D}(\mathbb{x}, z', \rho)\,,$$

where $z := \mathsf{F}(\mathbb{x}, \widetilde{\Pi}; \gamma)$ *and* $z' := \mathsf{F}(\mathbb{x}, \widetilde{\Pi}'; \gamma)$*.*

Moreover, we say that PCP *has shadow state size* $\mathsf{s_{PCP}}$ *if* F *outputs at most* $\mathsf{s_{PCP}}$ *bits.*

Remark: shadow soundness and ETH. The exponential time hypothesis (ETH) posits that solving 3SAT on n variables requires time exponential in n. This implies that no PCP for 3SAT can have a proof of size $o(n)$, as this would allow solving the problem in $2^{o(n)}$ time by iterating over all proofs. Since our construction yields a shadow state of size $o(n)$, it might appear to contradict ETH (or its randomized variant) by using the shadow as a succinct witness.

However, this approach fails for two main reasons. First, the shadow computation F is randomized. Even if the verifier supplies the randomness, the PCP definition provides no mechanism to ensure the prover generates the shadow using *that* specific randomness rather than a maliciously chosen one. Second, even if F were deterministic, we still cannot verify that the state we are iterating over is a valid image of $\mathsf{F}(\mathbb{x}; \cdot)$. Indeed, the image of F depends on whether the instance is in the language. Thus, if the instance is not in the language, we risk enumerating a state that falsely appears to be a valid witness.

How to use shadow PCPs. The main idea for leveraging shadow PCPs is to require the prover to commit to the PCP string in a way that is statistically binding with respect to the *shadow* of the PCP. To achieve this, we introduce a vector commitment (VC) scheme that satisfies a property we call *function statistical binding* with respect to a family of functions $\mathcal{F}$. While the commitment scheme is succinct, and thus cannot be statistically binding on the message itself, the function statistically binding property ensures that the commitment is statistically binding *relative to* a specific function $f \in \mathcal{F}$ (provided f has a small output length). The function f will compute the shadow from the PCP.

A crucial requirement for this approach is that the shadow randomness γ must remain hidden from the prover: if the prover knows γ, it could tailor its proof to the specific shadow check. We

achieve this by sampling γ as part of the CRS and requiring the VC scheme to satisfy *function hiding*. Concretely, function hiding guarantees that the public parameters (which depend on γ) do not leak any information about this randomness. As a result, the prover has to commit to the shadow without knowing the underlying randomness, thereby preserving the probabilistic guarantees of the shadow soundness. We elaborate more on our VC below in Section 2.2.

Constructing shadow PCPs. We provide a PCP construction whose shadow can be computed by polynomial-size low-depth circuit. Let $\mathsf{PCP} = (\mathbf{P}, \mathbf{V})$ be a PCP for $\mathsf{3SAT}$ with randomness complexity r. We design a compiler that converts PCP to a shadow PCP $\mathsf{PCP}' = (\mathbf{P}', \mathbf{V}')$ when r is small. Specifically, $\mathbf{P}'$ behaves the same as $\mathbf{P}$; $\mathbf{V}'$ breaks the set of all original verifier randomness $\rho \in \{0,1\}^{\mathsf{r}}$ into consecutive blocks of size $2^{\mathsf{r}/2}$, samples a random block index b, and accepts if and only if the original verifier $\mathbf{V}$ accepts all randomness in block b. Hence, the shadow circuit F can deterministically compute the new verifier's decision easily for every randomness and record them in a list of size $2^{\mathsf{r}/2}$. We emphasize that this construction gives *deterministic* shadow function, which is stronger than what shadow PCP requires. To provide full generality, we keep the rest of the discussion with randomized shadow function. Moreover, we emphasize that the circuit for the new PCP verifier $\mathbf{V}'$, given the instance $\mathbb{x}$, can be implemented with low-depth circuits, specifically circuits in NC. As a result, the PCP shadow circuit F is also in NC.

2.2 Vector commitments with function binding

We construct a vector commitment (VC) scheme, called a *function VC*, that satisfies a specialized *function statistically binding* property for a broad family of functions. We believe this construction is of independent interest.

Our function VC can be viewed as a programmable somewhere statistically binding (SSB) hash ([HW15; OPWW15]) where the "binding index" is replaced by a "binding function" (the tree algorithm). Recall that an SSB hash is initialized with a secret index i hidden from the adversary; while the hash is compressing (and thus only computationally binding globally), it guarantees that one can *extract* the i-th bit of the message, i.e., it is statistical binding specifically for the i-th bit. Formally, a hash family $\mathcal{H} = \{h_{\mathsf{k}}\}_{k \leftarrow \mathcal{H}.\mathsf{Gen}}$ is an SSB hash if there exists an efficient extractor $\mathcal{H}.\mathsf{Extract}$ such that for any (unbounded) adversary $\mathcal{A}$ and index i,

$$\Pr\left[\begin{array}{l|l} \mathcal{H}.\mathsf{Check}(\mathsf{k}, v, i, \sigma, \mathsf{pf}) = 1 & (\mathsf{k}, \mathsf{td}) \leftarrow \mathcal{H}.\mathsf{Gen}(1^\lambda, i) \\ \wedge\, \sigma \neq \widetilde{\sigma} & (v, \sigma, \mathsf{pf}) \leftarrow \mathcal{A}(\mathsf{k}) \\ & \widetilde{\sigma} := \mathcal{H}.\mathsf{Extract}(\mathsf{td}, v) \end{array}\right] \leq \mathrm{negl}(\lambda)\,,$$

where v is the hash output chosen by $\mathcal{A}$, σ is the alleged value at the i-th position of the preimage of v, and pf is an opening proof.

In our case, we initialize the VC with a hidden function f. Although the commitment is succinct, it is statistically binding with respect to the evaluation of f. We formalize this via a property called *function statistically binding*. Specifically, a function VC is a tuple of efficient algorithms

$$\mathsf{VC} = (\mathsf{VC.Gen}, \mathsf{VC.Commit}, \mathsf{VC.Open}, \mathsf{VC.Check}, \mathsf{VC.Extract})$$

where $(\mathsf{VC.Gen}, \mathsf{VC.Commit}, \mathsf{VC.Open}, \mathsf{VC.Check})$ works similarly as a standard VC, and function statistically binding is defined via $\mathsf{VC.Extract}$:

Definition 2 (Informal)**.** VC *is* **function statistically binding** *if for every function f and (unbounded) adversary A_{vc},*

$$\Pr\left[\begin{array}{l|l} \mathsf{VC.Check}(\mathsf{pp},\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})=1 & (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,f) \\ \wedge\,\forall\,\widetilde{m}:(f(\widetilde{m})\neq y\ \vee\ \widetilde{m}[\mathsf{Q}]\neq\mathsf{a}) & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})\coloneqq A_{\mathsf{vc}}(\mathsf{pp}) \\ & y\coloneqq\mathsf{VC.Extract}(\mathsf{td},\mathsf{cm}) \end{array}\right]\leq\mathrm{negl}(\lambda)\,.$$

Moreover, we require a function VC to satisfy *function hiding*, which ensures that given $f \neq f'$, no efficient adversary can distinguish the public parameters pp and pp' computed using f and f', respectively.

2.3 SNARG construction

Warm-up: designated verifier SNARG. To build intuition, we first describe how to construct a *designated verifier* SNARG (dv-SNARG) using a shadow PCP and a functional vector commitment. In this setting, the verifier holds a secret key—specifically, the trapdoor td of the vector commitment. We then demonstrate how to modify the construction to be *publicly verifiable*.

The designated verifier construction is given as follows.

- $\mathcal{P}$:
  1. Compute the PCP proof Π.
  2. Commit to Π using the function VC and obtain cm.
  3. Sample a verifier randomness ρ and provide the local openings $(\mathsf{Q},\mathsf{a},\mathsf{pf})$ corresponding to ρ.
  4. Send $(\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})$.
- $\mathcal{V}$:
  1. Use $\mathsf{VC.Decode}$ to extract the PCP shadow z underneath the commitment cm.
  2. Sample verifier randomness ρ (not necessarily the same as the one sampled by the prover).
  3. Check the following:
     (a) $\mathsf{VC.Check}(\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})=1$;
     (b) $\mathsf{D}(\mathbb{x},z,\rho)=1$.

By the the function statistically binding property of the VC, we know that if $\mathsf{VC.Check}(\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})=1$, there exists a PCP proof Π such that $\mathsf{F}(\Pi)=z$. Then the correctness of PCP shadow ensures that the second check using $\mathsf{D}(z,\rho)$ is consistent with the PCP verifier check $\mathbf{V}^{\Pi}(\mathbb{x};\rho)$.

Consequently, our main challenge is how to convert this dv-SNARG to a publicly verifiable one. The key idea is to use a correlation intractable hash family to "hide" the secret key of the verifier into a sparse relation.

Correlation-intractable hash instead of secret key. We now describe how to modify the construction to be publicly-verifier. The scheme follows the blueprint of Micali's construction [Kil92; Mic00]: On input an instance $\mathbb{x}$ and a witness $\mathbb{w}$, the SNARG prover $\mathcal{P}$ computes the PCP proof Π and commits to it using the VC scheme, obtaining a short commitment cm. It then derives the PCP verifier's randomness ρ via Fiat–Shamir, namely by computing $\rho\coloneqq h(x,\mathsf{cm})$ (where h is the Fiat–Shamir hash function), and uses ρ to determine the PCP query locations. The prover outputs cm together with the VC openings to the queried positions of Π. The verifier $\mathcal{V}$ recomputes ρ, checks that all openings are valid with respect to cm, and then runs the PCP verifier on the opened symbols.

To prove Theorem 1, we instantiate the above construction using a PCP with shadow soundness, our function VC scheme, and a correlation-intractable hash family. We outline the soundness analysis.

Consider $\mathbb{x} \notin L(R)$. As mentioned at the beginning of this section, one challenge is that interactive arguments only have computational soundness and do not induce sparse relations that allow us to use CIH on. However, since our function VC satisfies function statistically binding, we are able to leverage the statistical soundness of the underlying PCP. More formally, let F be the tree-computable function that computes the PCP shadow and D be the decider. Assume that the SNARG adversary $\widetilde{\mathcal{P}}$ successfully fools the SNARG verifier $\mathcal{V}$, which means that $\widetilde{\mathcal{P}}$ outputs $(\mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf})$ such that

$$\rho = h_{\mathsf{CI}}(\mathsf{cm}) \text{ and } \mathsf{VC.Check}(\mathsf{pp}, \mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf}) = 1 \text{ and } \mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x}; \rho) = 1\,,$$

where h_{CI} is the CI hash function and $\mathbf{V}$ is the PCP verifier. By the function statistically binding of the function VC, since $\mathsf{VC.Check}(\mathsf{pp}, \mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf}) = 1$, we can find $\widetilde{\Pi}$ such that

$$\mathsf{F}(\mathbb{x}, \widetilde{\Pi}; \gamma) = \mathsf{VC.Extract}(\mathsf{td}, \mathsf{cm}) \text{ and } \widetilde{\Pi}[\mathsf{Q}] = \mathsf{a}\,,$$

which implies that

$$\rho = h_{\mathsf{CI}}(\mathsf{cm}) \text{ and } \mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho) = 1\,.$$

This enables us to use PCP shadows to bound the SNARG soundness error.

Recall that in the definition of PCP shadows in Section 2.1, it is important that the shadow randomness γ remains hidden from the PCP adversary, as otherwise it's easy to choose some PCP string $\widetilde{\Pi}$ that fools the decider always. In the SNARG construction, the shadow randomness γ is sampled at random and then encrypted with the FHE, and the SNARG adversary only has access to its encryption. Intuitively, if the SNARG adversary were to guess γ, it has to break the FHE. On the other hand, if the SNARG adversary does not know γ, the sparsity property of PCP shadows induces a sparse relation that enables us to use CIH.

To formalize the intuition above, we set $z := \mathsf{F}(\mathbb{x}, \widetilde{\Pi}; \gamma)$ and consider two cases: $\mathsf{D}(\mathbb{x}, z, \rho) = 0$ and $\mathsf{D}(\mathbb{x}, z, \rho) = 1$. Specifically, we argue that neither of the following events can happen with high probability:

$$\begin{bmatrix} \rho = h_{\mathsf{CI}}(\mathsf{cm}) \\ \wedge\, \mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho) = 1 \\ \wedge\, \mathsf{D}(\mathbb{x}, z, \rho) = 0 \end{bmatrix} \text{ and } \begin{bmatrix} \rho = h_{\mathsf{CI}}(\mathsf{cm}) \\ \wedge\, \mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho) = 1 \\ \wedge\, \mathsf{D}(\mathbb{x}, z, \rho) = 1 \end{bmatrix}.$$

The probability of the event on the left-hand side can be bounded by the correctness error of PCP shadows (the first property in Definition 1), which ensures that for any PCP verifier randomness ρ, the PCP verifier $\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho)$ and the shadow decider $\mathsf{D}(\mathbb{x}, z, \rho)$ agree except with small probability over shadow randomness γ. In other words, in order to find $\widetilde{\Pi}$ and ρ such that $\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho)$ and $\mathsf{D}(\mathbb{x}, z, \rho)$ disagree with high probability, one has to know the secret randomness γ of the PCP shadow function F. Therefore, if $\widetilde{\mathcal{P}}$ succeeds with high probability, we would be able to correctly guess γ using $\widetilde{\mathcal{P}}$ with high probability, which violates the security of FHE. We note that in the formal analysis in Section 6.2, we additionally rely on the consistency of PCP shadows (the third property in Definition 1).

The other event can be analyzed using the CIH. By the sparsity of PCP shadows (the second property in Definition 1), the following relation is sparse:

$$R_{\mathbb{x},\mathsf{td}} := \{(\mathsf{cm}, \rho) : \mathsf{D}(\mathbb{x}, z, \rho) = 1 \text{ where } z := \mathsf{VC.Extract}(\mathsf{td}, \mathsf{cm})\}\,.$$

Therefore, $\rho = h_{\mathsf{CI}}(\mathsf{cm})$ and $\mathsf{D}(\mathbb{x}, z, \rho) = 1$ can happen with probability at most the CI hash error. CIH for all sparse relations is not known to follow from LWE alone: existing LWE-based constructions either require the relation to be computable by a bounded-size circuit [PS19], or they rely on relations with an inherent product structure [HLR21]. In our construction, we adopt the latter approach of [HLR21], since the PCP shadow soundness guarantee can be amplified via parallel repetition, which naturally induces a sparse product relation (see details in Sections 4 and 6).

Plugging in the shadow PCP constructed in Section 2.1 and given a function VC for NC, we can prove Theorem 2.

3 Preliminaries

Definition 3.1. *For every function $f\colon X \to Y$, the image of f, denoted as $\mathsf{Im}(f)$, is*

$$\mathsf{Im}(f) \coloneqq \{f(x) \in Y : x \in X\}\,.$$

Definition 3.2. *A* **relation** *R is a set of pairs $(\mathbb{x}, \mathbb{w})$ where $\mathbb{x}$ is an instance and $\mathbb{w}$ a witness. The corresponding* **language** *$L(R)$ is the set of instances $\mathbb{x}$ for which there exists a witness $\mathbb{w}$ such that $(\mathbb{x}, \mathbb{w}) \in R$.*

3.1 Correlation intractability

Definition 3.3. *For every relation ensemble $\mathbf{R} = \{R_\phi \subseteq \mathcal{X}_\phi \times \mathcal{W}_\phi\}$, we say that $\mathbf{R}$ is $\tau(\cdot)$-***sparse** *if for every ϕ and $x \in \mathcal{X}_\phi$,*

$$\Pr\left[(x, y) \in R_\phi \mid y \leftarrow \mathcal{W}_\phi\right] \leq \tau(\phi)\,.$$

Definition 3.4. *A hash family $\mathcal{H} = \{\mathcal{H}_\lambda\}_\lambda$ is* **correlation intractable wrt sparse relations** *if for every security parameter λ, ϕ, τ-sparse relation R_ϕ, $t_{\mathsf{CI}} \in \mathbb{N}$ and t_{CI}-size adversary A,*

$$\Pr\left[(x, h(x)) \in R_\phi \;\middle|\; \begin{array}{l} \mathsf{aux} \leftarrow \mathsf{A} \\ h \leftarrow \mathcal{H}_\lambda \\ x \leftarrow \mathsf{A}(\mathsf{aux}, h) \end{array}\right] \leq \epsilon_{\mathsf{CI}}(\phi, \tau, t_{\mathsf{CI}})\,.$$

Definition 3.5 (Product relation)**.** *A relation $R \subseteq \mathcal{X} \times \mathcal{W}^t$ is a product relation if for every $\mathbb{x}$, the set $R_\mathbb{x} \coloneqq \{\mathbb{w} : (\mathbb{x}, \mathbb{w} \in R)\}$ is the Cartesian product of sets $S_{\mathbb{x},1}, \ldots, S_{\mathbb{x},t}$:*

$$R_\mathbb{x} = S_{\mathbb{x},1} \times \cdots \times S_{\mathbb{x},t}\,.$$

Definition 3.6 (Efficient product verifiability)**.** *A product relation R is T-time product verifiable if there exists a size-T circuit C such that for every $\mathbb{x}$ and its corresponding $S_{\mathbb{x},1}, \ldots, S_{\mathbb{x},t}$, $\mathbb{w}_i \in S_{\mathbb{x},i}$ if and only if $C(\mathbb{x}, \mathbb{w}_i, i) = 1$.*

Definition 3.7 (Product sparsity)**.** *A product relation $R \subseteq \mathcal{X} \times \mathcal{W}^t$ has sparsity τ if for every $\mathbb{x}$ and its corresponding $S_{\mathbb{x},1}, \ldots, S_{\mathbb{x},t}$, $|S_{\mathbb{x},i}| \leq \tau\,|\mathcal{W}|$.*

Theorem 3.8 ([HLR21])**.** *Let $\mathbf{R}$ be the ensemble of all T-time verifiable product relation $R_\lambda \subseteq \mathcal{X} \times \mathcal{W}^{t_\lambda}$ with product sparsity τ, where $t_\lambda > \lambda/(1-\tau)$. There exists a hash family $\mathcal{H}_{\mathsf{CI}} = \{\mathcal{H}_{\mathsf{CI}}^{(\lambda)}\colon \mathcal{X} \to \mathcal{W}^{t_\lambda}\}_{\lambda \in \mathbb{N}}$ that is correlation intractable for $\mathbf{R}$ under the hardness of LWE. Moreover, $\mathcal{H}_{\mathsf{CI}}$ depends only on $\mathcal{X}, \mathcal{W}, T, t_\lambda, \tau$ and can be evaluated in time $\mathsf{poly}(\log|\mathcal{X}|, t_\lambda, T)$.*

3.2 Non-interactive arguments

A *non-interactive argument* for a relation R (in the common reference string model) is a tuple of algorithms $\mathsf{SNARG} = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ satisfying the following properties.

Definition 3.9. *$\mathsf{SNARG} = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ has* **perfect completeness** *if for every security parameter $\lambda \in \mathbb{N}$, instance size bound $n \in \mathbb{N}$, public parameter $\mathsf{pp} \in \mathcal{G}(1^\lambda, n)$, and instance-witness pair $(\mathbb{x}, \mathbb{w}) \in R$,*

$$\Pr\left[\mathcal{V}(\mathsf{pp}, \mathbb{x}, \pi) = 1 \mid \pi \leftarrow \mathcal{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w})\right] = 1\,.$$

Definition 3.10. $\mathsf{SNARG} = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ *has* **non-adaptive soundness error** $\epsilon_{\mathsf{SNARG}}$ *if for every security parameter* $\lambda \in \mathbb{N}$*, instance* $\mathbb{x} \notin L(R)$*, circuit size bound* $t_{\mathsf{SNARG}} \in \mathbb{N}$*, and* t_{SNARG}*-size circuit* $\widetilde{\mathcal{P}}$*,*

$$\Pr\left[\mathcal{V}(\mathsf{pp}, \mathbb{x}, \pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda, n) \\ \pi \leftarrow \widetilde{\mathcal{P}}(\mathsf{pp}) \end{array}\right] \leq \epsilon_{\mathsf{SNARG}}(\lambda, \mathbb{x}, t_{\mathsf{ARG}})\,.$$

Definition 3.11. $\mathsf{SNARG} = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ *has* **adaptive soundness error** $\epsilon_{\mathsf{SNARG}}$ *if for every security parameter* $\lambda \in \mathbb{N}$*, instance size bound* $n \in \mathbb{N}$*, circuit size bound* $t_{\mathsf{SNARG}} \in \mathbb{N}$*, and* t_{SNARG}*-size circuit* $\widetilde{\mathcal{P}}$*,*

$$\Pr\left[\begin{array}{l} |\mathbb{x}| \leq n \\ \wedge\, \mathbb{x} \notin L(R) \\ \wedge\, \mathcal{V}(\mathsf{pp}, \mathbb{x}, \pi) = 1 \end{array} \;\middle|\; \begin{array}{l} \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda, n) \\ \pi \leftarrow \widetilde{\mathcal{P}}(\mathsf{pp}) \end{array}\right] \leq \epsilon_{\mathsf{SNARG}}(\lambda, \mathbb{x}, t_{\mathsf{ARG}})\,.$$

Theorem 3.12 ([WW25])**.** *Assume the existence of sub-exponential indistinguishability obfuscation (iO) for Boolean circuits and sub-exponential one-way functions (OWF), there exists a SNARG for any* NP *relation* R *that satisfies the following:*

- *proof size* $\mathsf{poly}(\lambda)$*;*
- *public parameter size* $\mathsf{poly}(\lambda, |R|)$*, where* $|R|$ *is the size of the Boolean circuit computing* R*;*
- *perfect completeness;*
- *adaptive soundness error* $\mathrm{negl}(\lambda)$*.*

3.3 Probabilistically checkable proofs

A *probabilistically checkable proof* (PCP) is an information-theoretic proof system where a probabilistic verifier has oracle access to a proof string.

Definition 3.13 (Completeness)**.** $\mathsf{PCP} = (\mathbf{P}, \mathbf{V})$ *for a relation* R *has* **perfect completeness** *if, for every instance-witness pair* $(\mathbb{x}, \mathbb{w}) \in R$*,*

$$\Pr\left[\mathbf{V}^{\Pi}(\mathbb{x}) = 1 \;\middle|\; \Pi \leftarrow \mathbf{P}(\mathbb{x}, \mathbb{w})\right] = 1\,.$$

Definition 3.14 (Soundness)**.** $\mathsf{PCP} = (\mathbf{P}, \mathbf{V})$ *for a relation* R *has* **soundness error** ϵ_{PCP} *if, for every instance* $\mathbb{x} \notin L(R)$ *and (unbounded) circuit* $\widetilde{\mathbf{P}}_{\mathsf{PCP}}$*,*

$$\Pr\left[\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}) = 1 \;\middle|\; \widetilde{\Pi} \leftarrow \widetilde{\mathbf{P}}_{\mathsf{PCP}}\right] \leq \epsilon_{\mathsf{PCP}}(\mathbb{x})\,.$$

We consider several efficiency measures for a PCP:

- the *proof alphabet* Σ is the alphabet over which a PCP string is written;
- the *proof length* ℓ is the number of alphabet symbols in the PCP string;
- the *query complexity* $\mathsf{q} \in [\ell]$ is the number of queries that the PCP verifier makes to the PCP string (each query is an index in $[\ell]$ and is answered by the corresponding symbol in Σ in the PCP string);
- the *randomness complexity* r is the number of random bits used by the PCP verifier.

An efficiency measure may be a function of the instance $\mathbb{x}$ (e.g., of its size $|\mathbb{x}|$).

Theorem 3.15 ([BS05; Din07; BMVY25])**.** *There is a* PCP *for* 3SAT *that satisfies the following:*

- *perfect completeness;*
- *soundness error* $\epsilon_{\mathsf{PCP}} = 1/2$*;*
- *non-adaptive verifier;*
- *query complexity* $\mathsf{q} = O(1)$*;*
- *randomness complexity* $\mathsf{r} = \log n + O(\log \log n)$*;*
- *proof length* $\ell = n \cdot \mathsf{polylog} n$*;*
- *alphabet* $\{0,1\}$*.*

3.4 Merkle tree

We recall the construction of the Merkle vector commitment scheme, adapted from [CY24].

Definition 3.16. *For* $\ell \in \mathbb{N}$*,* $T_\ell = (V_\ell, E_\ell)$ *is the (perfect)* **binary tree graph** *of depth*

$$d \coloneqq \log_2 \ell$$

where the vertex set V_ℓ *and edge set* E_ℓ *are defined as follows:*

$$V_\ell \coloneqq \left\{(j,i) : j \in \{0,1,\ldots,d\}, i \in [2^j]\right\} ,$$
$$E_\ell \coloneqq \left\{\big((j-1,i),(j,2i-b)\big) : j \in \{1,\ldots,d\}, i \in [2^{j-1}], b \in \{1,0\}\right\} .$$

Definition 3.17. *We make the following notational definitions:*
- *The* **root vertex** *of* T_ℓ *is the vertex* $(0,1)$*.*
- *The* **leaf vertices** *of* T_ℓ *are the vertices* $\{(d,i)\}_{i\in[\ell]}$*.*
- *A vertex* (j,i) *is* **odd** *if* i *is odd and it is* **even** *if* i *is even.*
- *The* **sibling** *of a non-root vertex* (j,i) *(i.e., with* $j > 0$*) is* $(j,i+1)$ *if* (j,i) *is odd and is* $(j,i-1)$ *if* (j,i) *is even.*
- *The* **path** *from the* i*-th leaf vertex* (d,i) *to the root vertex is denoted by* $\mathsf{path}(i)$*; the vertex in* $\mathsf{path}(i)$ *that is in layer* j *is denoted* $\mathsf{p}(i,j) \in \{j\} \times [2^j]$*.*
- *The* **copath** *from the* i*-th leaf vertex* (d,i) *to the root vertex is denoted by* $\mathsf{copath}(i)$*, and is the list of siblings of each vertex in* $\mathsf{path}(i)$*, except the root vertex (which has no siblings); the vertex in* $\mathsf{copath}(i)$ *that is in layer* j *is denoted* $\bar{\mathsf{p}}(i,j) \in \{j\} \times [2^j]$ *(and is the sibling of* $\mathsf{p}(i,j)$*).*
- *For* $I \subseteq [\ell]$*,* $\mathsf{path}(I) \coloneqq \cup_{i\in I}\mathsf{path}(i)$ *and* $\mathsf{copath}(I) \coloneqq \cup_{i\in I}\mathsf{copath}(i)$ *(viewing lists as sets).*

Using this notation we can write:

$$\mathsf{path}(i) = \Big(\mathsf{p}(i,j)\Big)_{j\in\{0,1,\ldots,d\}} \text{ and } \mathsf{copath}(i) = \Big(\bar{\mathsf{p}}(i,j)\Big)_{j\in\{1,\ldots,d\}} .$$

For every $i \in [\ell]$, $\mathsf{p}(i,0) = (0,1)$ is the root vertex and $\mathsf{p}(i,d) = (d,i)$ is the i-th leaf.

Construction 3.18. Given $d+1$ hash families $\mathcal{H}_0, \ldots, \mathcal{H}_d$, we construct the Merkle commitment scheme as follows:

- $\mathsf{MT.Gen}(1^\lambda, \ell, \mathsf{seed}_0, \ldots, \mathsf{seed}_d)$:
  1. For $i \in \{0, \cdots, d\}$: Sample $(h_i, \mathsf{td}_i) \leftarrow \mathcal{H}_i(\mathsf{seed}_i)$.
  2. Set $\mathsf{pp} \coloneqq (h_i)_{i\in\{0,\ldots,d\}}$.
  3. Set $\mathsf{td} \coloneqq (\mathsf{td}_i)_{i\in\{0,\ldots,d\}}$.
  4. Output $(\mathsf{pp}, \mathsf{td})$.

- $\mathsf{MT.Commit}(\mathsf{pp}, m)$:
  1. Parse pp as $(h_i)_{i\in\{0,\ldots,d\}}$.
  2. For each message location $i = 1, \ldots, \ell$: Compute $c_{(d,i)} \coloneqq h_d(i, m[i])$. Note that in the leaf level, we hash both the value at location i and the location itself.
  3. For each layer $j = d-1, \ldots, 0$ and for each $i = 1, \ldots, 2^j$: Compute $c_{(j,i)} \coloneqq h_j(c_{(j+1,2i-1)}, c_{(j+1,2i)})$.
  4. Set the Merkle commitment $\mathsf{cm} \coloneqq c_{(0,1)}$.
  5. Set the auxiliary information $\mathsf{aux} \coloneqq ((c_{(j,i)})_{i\in[2^j]})_{j=0}^{d}$, which we can visualize in the following way:
$$\mathsf{aux} = \begin{bmatrix} c_{(0,1)} & & & & & \\ c_{(1,1)} & c_{(1,2)} & & & & \\ c_{(2,1)} & c_{(2,2)} & c_{(2,3)} & c_{(2,4)} & & \\ \vdots & \vdots & \vdots & \vdots & \ddots & \\ c_{(d,1)} & \cdots & \cdots & \cdots & \cdots & c_{(d,2^d)} \end{bmatrix}.$$
  6. Output $(\mathsf{cm}, \mathsf{aux})$.

- $\mathsf{MT.Open}(\mathsf{pp}, \mathsf{aux}, \mathcal{Q})$:
  1. For every $i \in \mathcal{Q}$, set the authentication path for location i:
$$\mathsf{auth}_i \coloneqq (c_{\bar{\mathsf{p}}(i,j)})_{j\in\{1,\ldots,d\}}.$$
  2. Output the opening proof $\mathsf{pf} \coloneqq (\mathsf{auth}_i)_{i\in\mathcal{Q}}$.

- $\mathsf{MT.Check}(\mathsf{pp}, \mathsf{cm}, \mathcal{Q}, \mathsf{ans}, \mathsf{pf})$:
  1. Parse pf as $(\mathsf{auth}_i)_{i\in\mathcal{Q}}$.
  2. For every $i \in \mathcal{Q}$, check that auth_i authenticates the value $\mathsf{ans}[i]$ as the i-th opening relative to the Merkle commitment cm by computing $(c_{\mathsf{p}(i,j)})_{j\in\{0,1,\ldots,d\}}$:
     (a) Compute the commitment $c_{(d,i)} \coloneqq h_j(\mathsf{ans}[i])$;
     (b) Compute the commitments $(c_{\mathsf{p}(i,j)})_{j\in\{0,1,\ldots,d-1\}}$ as follows:
     For each layer $j = d-1, \ldots, 1, 0$:
     – If $\mathsf{p}(i, j+1)$ is an odd vertex then set $c_{\mathsf{L}} \coloneqq c_{\mathsf{p}(i,j+1)}$ and $c_{\mathsf{R}} \coloneqq c_{\bar{\mathsf{p}}(i,j+1)}$;
     – If $\mathsf{p}(i, j+1)$ is an even vertex then set $c_{\mathsf{L}} \coloneqq c_{\bar{\mathsf{p}}(i,j+1)}$ and $c_{\mathsf{R}} \coloneqq c_{\mathsf{p}(i,j+1)}$;
     – Compute the commitment $c_{\mathsf{p}(i,j)} \coloneqq h_j(c_{\mathsf{L}}, c_{\mathsf{R}})$.
     (c) Check that $\mathsf{cm} = c_{(0,1)}$.

In the construction above, each query $i \in \mathcal{Q}$ has a separate authentication path auth_i. We introduce the operation Merge that takes in a list of authentication paths $(\mathsf{auth}_i)_{i\in\mathcal{Q}}$ as input and works as follows:

$\mathsf{Merge}((\mathsf{auth}_k)_{k\in\mathcal{Q}})$:
1. For every vertex (j, i) in the tree:
   (a) If there is no $i \in \mathcal{Q}$ such that (j, i) is included, set $c_{(j,i)} \coloneqq \perp$.
   (b) If (j, i) is included in some paths, and there is no inconsistency across paths, set $c_{(j,i)}$ to be the corresponding value in the path.
   (c) If (j, i) has inconsistency values across different paths, output $\perp$.
2. Output $(c_{(j,i)})_{j\in d, i\in[2^j]}$.

Definition 3.19 (Collision resistant hash function)**.** *A hash family $\mathcal{H} = \{\mathcal{H}_\lambda\}_\lambda$ is* **collision resistant** *if for every security parameter $\lambda \in \mathbb{N}$, $t \in \mathbb{N}$ and t-size adversary A,*

$$\Pr\left[\begin{array}{l|l} x \neq x' & h \leftarrow \mathcal{H}(\lambda) \\ h(x) = h(x') & (x, x') \leftarrow A(h) \end{array}\right] \leq \epsilon_{\mathsf{CRHF}}(\lambda, t)\,.$$

Lemma 3.20. *Let $\mathcal{H}$ be a collision resistant hash family with error ϵ_{CRHF}. For every $\lambda \in \mathbb{N}$, $t \in \mathbb{N}$, and t-size adversary A,*

$$\Pr\left[\begin{array}{l|l} \mathsf{MT.Check}(\mathsf{pp}, \mathsf{cm}, \mathcal{Q}, \mathsf{ans}, \mathsf{pf}) = 1 & \textit{For } i \in \{0, \ldots, d\}: \\ \wedge\, \mathsf{Merge}((\mathsf{auth}_i)_{i \in \mathcal{Q}}) = \perp & \quad h_i \leftarrow \mathcal{H}(\lambda) \\ & \mathsf{pp} := (h_i)_{i \in \{0,\ldots,d\}} \\ & (\mathsf{cm}, \mathcal{Q}, \mathsf{ans}, \mathsf{pf} := (\mathsf{auth}_i)_{i \in \mathcal{Q}}) \leftarrow A(\mathsf{pp}) \end{array}\right] \leq \epsilon_{\mathsf{CRHF}}(\lambda, t)\,.$$

Proof. It is easy to see that if A is able to find inconsistent authentication paths that are accepted by $\mathsf{MT.Check}$, it can find collisions in h_{CRHF}. □

3.5 Learning with errors

Definition 3.21 (LWE distribution)**.** *For any positive integers $n, q \in \mathbb{N}$, vector $\mathbf{s} \in \mathbb{Z}^n$, and error distribution χ over $\mathbb{Z}$, the learning with error (LWE) distribution $A_{\mathbf{s},\chi}$ is defined by uniformly sampling a vector $\mathbf{a} \in \mathbb{Z}^n$, sampling error e from χ, and outputting $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + \mathsf{e} \bmod q) \in \mathbb{Z}_q^{\mathbf{s}} \times \mathbb{Z}_q$. χ is usually the discrete Gaussian of parameter $\alpha \cdot q$. The modulus-to-noise ratio is $\frac{q}{\alpha \cdot q} = 1/\alpha$.*

Definition 3.22 (Sub-exponential hardness of LWE)**.** *Fix the security parameter $\lambda \in \mathbb{N}$. Consider n, q and χ that depend on λ. In particular, let $n = \Theta(\lambda)$ be the LWE secret vector size. Let $q = \mathsf{poly}(\lambda)$ be the LWE modulus. The sub-exponential LWE assumption $\mathsf{LWE}_{n,q,\chi}$ states that for every $c > 1$, there exists c' such that no non-uniform probabilistic $2^{\lambda^{1/c}}$-size adversary can distinguish, with probability more than $2^{-\lambda^{1/c'}}$, between (i) the distribution $A_{\mathbf{s},\chi}$ for a random $\mathbf{s} \leftarrow \mathbb{Z}_q^n$; and (ii) the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$.*

3.6 Fully homomorphic encryption

A ***fully homomorphic encryption scheme*** with alphabet Σ, message size ℓ and cipher text size n is a tuple of efficient algorithms $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Dec}, \mathsf{FHE.Eval})$ such that:

- $\mathsf{FHE.Gen}(1^\lambda)$ is a randomized algorithm that takes as input the security parameter and outputs a public key pk and a secret key sk.
- $\mathsf{FHE.Enc}(\mathsf{pk}, m)$ is a randomized algorithm that takes as input the public key pk and a message m and outputs a cipher text ct.
- $\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{ct})$ is a deterministic algorithm that takes as input the secret key sk and a cipher text ct and outputs a message m'.
- $\mathsf{FHE.Eval}(\mathsf{pk}, f, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)$ is a deterministic algorithm that takes as input the public key pk, a circuit f that takes in k inputs and output a string in Σ^ℓ, and a vector of cipher texts $(\mathsf{ct}_1, \ldots, \mathsf{ct}_k)$ and outputs another cipher text ct'.

We require the following properties for an FHE scheme:

- *Correctness:* For every $\lambda \in \mathbb{N}$ and message m,

$$\Pr\left[\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{FHE.Enc}(\mathsf{pk}, m)) = m \;\middle|\; (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)\right] = 1\,.$$

- *Semantic security:* For every $\lambda \in \mathbb{N}$, adversary time bound $t_{\mathsf{FHE}} \in \mathbb{N}$, and t_{FHE}-time adversary $\mathcal{A}$,

$$\left|\Pr\left[b = 1 \;\middle|\; \begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)\\ (m_0, m_1, \mathsf{aux}) \leftarrow \mathcal{A}(\mathsf{pk})\\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_0)\\ b \leftarrow \mathcal{A}(\mathsf{aux}, \mathsf{pk}, \mathsf{ct})\end{array}\right] - \Pr\left[b = 1 \;\middle|\; \begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)\\ (m_0, m_1, \mathsf{aux}) \leftarrow \mathcal{A}(\mathsf{pk})\\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_1)\\ b \leftarrow \mathcal{A}(\mathsf{aux}, \mathsf{pk}, \mathsf{ct})\end{array}\right]\right| \leq \epsilon_{\mathsf{FHE}}(\lambda, t_{\mathsf{FHE}})\,.$$

- e_{FHE}*-expansion:* For every $\lambda \in \mathbb{N}$, message m, and $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Supp}(\mathsf{FHE.Gen}(1^\lambda))$,

$$|\mathsf{FHE.Enc}(\mathsf{pk}, m)| \leq e_{\mathsf{FHE}}(\lambda, |m|)\,.$$

- c_{FHE}*-compactness:* For every $\lambda \in \mathbb{N}$, circuit f that takes in k inputs and output a string in Σ^ℓ, inputs $(m_1, \ldots, m_k)$, $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Supp}(\mathsf{FHE.Gen}(1^\lambda))$, $\mathsf{ct}_i \in \mathsf{Supp}(\mathsf{FHE.Enc}(\mathsf{pk}, m_i))$ for every $i \in [k]$,

$$|\mathsf{FHE.Eval}(\mathsf{pk}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_k))| \leq c_{\mathsf{FHE}}(\lambda, |f(m_1, \ldots, m_k)|)\,.$$

- *Perfect homomorphism:* For every $\lambda \in \mathbb{N}$, $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Supp}(\mathsf{FHE.Gen}(1^\lambda))$, circuit f, and inputs $(m_1, \ldots, m_k)$, consider $n_i \in \mathsf{Supp}(\mathsf{FHE.Enc}(\mathsf{pk}, m_i))$ for every $i \in [k]$, it holds that

$$\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{FHE.Eval}(\mathsf{pk}, f, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)) = f(m_1, \ldots, m_k)\,.$$

There is a long line of work that constructs FHE using LWE. The theorem below states the existence of FHE with parameters and security that we need in this paper.

Theorem 3.23 ([BGV12; Bra12; GSW13; BDGM19])**.** *Assume the sub-exponential hardness of LWE holds. Let $\lambda \in \mathbb{N}$ be the security parameter, there exists* FHE *for all circuits of depth* $\mathsf{poly}(\lambda)$ *that satisfies the following:*

- *size of the secret key* $\mathsf{poly}(\lambda)$*;*
- *size of public key* $\mathsf{poly}(\lambda)$*;*
- *expansion* $e_{\mathsf{FHE}}(\lambda, \ell) = \ell \cdot \mathsf{poly}(\lambda)$*;*
- *compactness* $c_{\mathsf{FHE}}(\lambda, \ell) = \ell \cdot \mathsf{poly}(\lambda)$*;*
- *perfect correctness;*
- *sub-exponential semantic security: for every $c > 1$, there exists c' such that if $t_{\mathsf{FHE}} \leq 2^{\lambda^{1/c}}$, then $\epsilon_{\mathsf{FHE}}(\lambda, t_{\mathsf{FHE}}) \leq 2^{-\lambda^{1/c'}}$;*
- *perfect homomorphism.*

The following lemma generalizes semantic security to hold for a list of n messages.

Lemma 3.24 (Multi-message semantic security)**.** *For every $\lambda \in \mathbb{N}$, $n \in \mathbb{N}$, adversary time bound $t_{\mathsf{FHE}} \in \mathbb{N}$, and t_{FHE}-time adversary $\mathcal{A}$,*

$$\left|\Pr\left[b = 0 \;\middle|\; \begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)\\ ((m_{0,i})_{i\in[n]}, (m_{1,i})_{i\in[n]}) \leftarrow \mathcal{A}(\mathsf{pk})\\ \textit{For } i \in [n]:\\ \quad \mathsf{ct}_i \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_{0,i})\\ b \leftarrow \mathcal{A}(\mathsf{pk}, (\mathsf{ct}_i)_{i\in[n]})\end{array}\right] - \Pr\left[b = 0 \;\middle|\; \begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)\\ ((m_{0,i})_{i\in[n]}, (m_{1,i})_{i\in[n]}) \leftarrow \mathcal{A}(\mathsf{pk})\\ \textit{For } i \in [n]:\\ \quad \mathsf{ct}_i \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_{1,i})\\ b \leftarrow \mathcal{A}(\mathsf{pk}, (\mathsf{ct}_i)_{i\in[n]})\end{array}\right]\right| \leq n \cdot \epsilon_{\mathsf{FHE}}(\lambda, t_{\mathsf{FHE}})\,.$$

Proof. Let $\mathcal{A}$ be an adversary described in the lemma statement. We construct the following adversary $\mathcal{B}$ for the semantic security of FHE:

- $\mathcal{B}(\mathsf{pk})$:
  1. Compute $((m_{0,i})_{i\in[n]}, (m_{1,i})_{i\in[n]}) \leftarrow \mathcal{A}(\mathsf{pk})$.
  2. Sample $i \leftarrow [n]$.
  3. Output $(\mathsf{aux} := (i, ((m_{0,j})_{j\in[n]}, (m_{1,j})_{j\in[n]})), m_{0,i}, m_{1,i})$.
- $\mathcal{B}(\mathsf{aux}, \mathsf{pk}, \mathsf{ct})$:
  1. Parse aux as $(i, (i, ((m_{0,j})_{j\in[n]}, (m_{1,j})_{j\in[n]})))$.
  2. For $j < i$: Compute $\mathsf{ct}_j \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_{0,j})$.
  3. For $j > i$: Compute $\mathsf{ct}_j \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_{1,j})$.
  4. Set $\mathsf{ct}_i := \mathsf{ct}$.
  5. Compute $b \leftarrow \mathcal{A}(\mathsf{pk}, (\mathsf{ct}_j)_{j\in[n]})$.

  We define p_i as follows:

$$p_i := \Pr\left[b = 1 \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ ((m_{0,j})_{j\in[n]}, (m_{1,j})_{j\in[n]}) \leftarrow \mathcal{A}(\mathsf{pk}) \\ \text{For } j \le i : \mathsf{ct}_j \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_{0,j}) \\ \text{For } j > i : \mathsf{ct}_j \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_{1,j}) \\ b \leftarrow \mathcal{A}(\mathsf{pk}, (\mathsf{ct}_j)_{j\in[n]}) \end{array} \right].$$

Therefore, we remain to bound $|p_n - p_1|$. Notice that

$$\Pr\left[b = 1 \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ (m_0, m_1) \leftarrow \mathcal{B}(\mathsf{pk}) \\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_0) \\ b \leftarrow \mathcal{B}(\mathsf{pk}, (\mathsf{ct}_j)_{j\in[n]}) \end{array} \right] = \frac{1}{n} \cdot \sum_{i=1}^{n} p_i \,.$$

Moreover,

$$\Pr\left[b = 1 \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ (m_0, m_1) \leftarrow \mathcal{B}(\mathsf{pk}) \\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_1) \\ b \leftarrow \mathcal{B}(\mathsf{pk}, (\mathsf{ct}_j)_{j\in[n]}) \end{array} \right] = \frac{1}{n} \cdot \sum_{i=1}^{n} p_{i-1} \,.$$

Hence,

$$\epsilon_{\mathsf{FHE}}(\lambda, t_{\mathsf{FHE}}) \ge \left| \frac{1}{n} \cdot \sum_{i=1}^{n} p_i - \frac{1}{n} \cdot \sum_{i=1}^{n} p_{i-1} \right| = \frac{1}{n} \left| p_n - p_1 \right| .$$

□

The lemma below reformulates the semantic security to hide the randomness of a function under encryption. In particular, if we encrypt a randomized function with a uniformly sampled randomness, no efficient adversary would be able to guess the randomness with high probability.

Lemma 3.25 (Randomness hiding)**.** *Fix* $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Dec})$*. For every* $\lambda \in \mathbb{N}$*, randomized function* f*, adversary running time* $t \in \mathbb{N}$*, and* t*-time adversary* A*, the following holds:*

$$\Pr\left[\rho' = \rho \;\middle|\; \begin{array}{l} \rho \leftarrow \{0,1\}^\ell \\ (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, f_\rho) \\ \rho' \leftarrow A(\mathsf{pk}, \mathsf{ct}) \end{array}\right] \leq \frac{1}{2^\ell} + \epsilon_{\mathsf{FHE}}(\lambda, t)\,.$$

Proof. Assume for contradiction that there exists an adversary A such that the probability in the lemma statement is greater than $\frac{1}{2^\ell} + \epsilon_{\mathsf{FHE}}(\lambda, t)$. We construct an adversary A' that breaks the semantic security of FHE. Let ρ be a randomly sampled element from $\{0,1\}^\ell$. Let $m_0 \coloneqq \mathbf{0}$ and $m_1 \coloneqq f_\rho$.

$A'_{m_0,m_1}(\mathsf{pk}, \mathsf{ct})$:
1. Run $\rho' \leftarrow A(\mathsf{pk}, \mathsf{ct})$.
2. If ρ' is such that $f_\rho = f_{\rho'}$, output 1.
3. Otherwise, output 0.

$$\begin{aligned}
&\left|\Pr\left[b = 1 \;\middle|\; \begin{array}{l} \rho \leftarrow \{0,1\}^\ell \\ m_0 \coloneqq \mathbf{0} \\ m_1 \coloneqq f_\rho \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_1) \\ b \leftarrow A'_{m_0,m_1}(\mathsf{pk}, \mathsf{ct}) \end{array}\right] - \Pr\left[b = 1 \;\middle|\; \begin{array}{l} \rho \leftarrow \{0,1\}^\ell \\ m_0 \coloneqq \mathbf{0} \\ m_1 \coloneqq f_\rho \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_0) \\ b \leftarrow A'_{m_0,m_1}(\mathsf{pk}, \mathsf{ct}) \end{array}\right]\right| \\
&= \left|\Pr\left[b = 1 \;\middle|\; \begin{array}{l} \rho \leftarrow \{0,1\}^\ell \\ m_0 \coloneqq \mathbf{0} \\ m_1 \coloneqq f_\rho \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, m_1) \\ b \leftarrow A'_{m_0,m_1}(\mathsf{pk}, \mathsf{ct}) \end{array}\right] - \frac{1}{2^\ell}\right| \\
&> \frac{1}{2^\ell} + \epsilon_{\mathsf{FHE}}(\lambda, t) - \frac{1}{2^\ell} \\
&= \epsilon_{\mathsf{FHE}}(\lambda, t)\,,
\end{aligned}$$

where the equality holds since when ct is an encryption of m_0 it contains no information about ρ, and thus the output ρ' of A is independent from sampling ρ and thus $\rho = \rho'$ with probability exactly $2^{-\ell}$, and the inequality holds because of our assumption. □

3.7 Non-interactive batch arguments

Definition 3.26 (Batched index relation)**.** *A* **batched index relation** R *is a set of tuples* $(\mathtt{i}, (\mathtt{x}_1, \ldots, \mathtt{x}_k), (\mathtt{w}_1, \ldots, \mathtt{w}_k))$*. The corresponding* **language** $L(R)$ *is the set of tuples* $(\mathtt{i}, (\mathtt{x}_1, \ldots, \mathtt{x}_k))$ *for which there exist witnesses* $(\mathtt{w}_1, \ldots, \mathtt{w}_k)$ *such that* $(\mathtt{i}, (\mathtt{x}_1, \ldots, \mathtt{x}_k), (\mathtt{w}_1, \ldots, \mathtt{w}_k)) \in R$*.*

A non-interactive batch argument (BARG) is a tuple of algorithms $\mathsf{BARG} = (\mathsf{Setup}_{\mathsf{BG}}, \mathbf{P}_{\mathsf{BG}}, \mathbf{V}_{\mathsf{BG}})$ with the following interface:

- $\mathsf{Setup}_{\mathsf{BG}}(1^\lambda, k, 1^m, i) \to \mathsf{pp}_{\mathsf{BG}}$: On input the security parameter $\lambda \in \mathbb{N}$, the number of instances $k \in \mathbb{N}$, the witness size $m \in \mathbb{N}$, and an index $i \in [k]$, $\mathsf{Setup}_{\mathsf{BG}}$ outputs a trapdoor public parameter $\mathsf{pp}_{\mathsf{BG}}$.
- $\mathbf{P}_{\mathsf{BG}}(\mathsf{pp}_{\mathsf{BG}}, \mathbb{i}, (\mathbb{x}_1, \ldots, \mathbb{x}_k), (\mathbb{w}_1, \ldots, \mathbb{w}_k)) \to \pi_{\mathsf{BG}}$: On input the public parameter $\mathsf{pp}_{\mathsf{BG}}$, an index $\mathbb{i}$, and k instance-witness pairs $(\mathbb{x}_1, \ldots, \mathbb{x}_k), (\mathbb{w}_1, \ldots, \mathbb{w}_k)$, $\mathbf{P}_{\mathsf{BG}}$ outputs a proof π_{BG}.
- $\mathbf{V}_{\mathsf{BG}}(\mathsf{pp}_{\mathsf{BG}}, \mathbb{i}, (\mathbb{x}_1, \ldots, \mathbb{x}_k), \pi_{\mathsf{BG}}) \to b$: On input the public parameter $\mathsf{pp}_{\mathsf{BG}}$, an index $\mathbb{i}$, k instances $(\mathbb{x}_1, \ldots, \mathbb{x}_k)$, and a proof π_{BG}, $\mathbf{V}_{\mathsf{BG}}$ outputs a decision bit $b \in \{0, 1\}$.

Definition 3.27 (Completeness)**.** $\mathsf{BARG} = (\mathsf{Setup}_{\mathsf{BG}}, \mathbf{P}_{\mathsf{BG}}, \mathbf{V}_{\mathsf{BG}})$ *is perfectly complete if for every* $\lambda \in \mathbb{N}$*,* $k \in \mathbb{N}$*,* $i \in [k]$*, and* $(\mathbb{i}, (\mathbb{x}_1, \ldots, \mathbb{x}_k), (\mathbb{w}_1, \ldots, \mathbb{w}_k)) \in R$*,*

$$\Pr\left[\mathbf{V}_{\mathsf{BG}}(\mathsf{pp}_{\mathsf{BG}}, \mathbb{i}, (\mathbb{x}_1, \ldots, \mathbb{x}_k), \pi_{\mathsf{BG}}) = 1 \;\middle|\; \begin{array}{l} \mathsf{pp}_{\mathsf{BG}} \leftarrow \mathsf{Setup}_{\mathsf{BG}}(1^\lambda, k, 1^m, i) \\ \pi_{\mathsf{BG}} \leftarrow \mathbf{P}_{\mathsf{BG}}(\mathsf{pp}_{\mathsf{BG}}, \mathbb{i}, (\mathbb{x}_1, \ldots, \mathbb{x}_k), (\mathbb{w}_1, \ldots, \mathbb{w}_k)) \end{array}\right] = 1\,.$$

Definition 3.28 (Index hiding)**.** $\mathsf{BARG} = (\mathsf{Setup}_{\mathsf{BG}}, \mathbf{P}_{\mathsf{BG}}, \mathbf{V}_{\mathsf{BG}})$ *has index hiding error* ζ_{BG} *if for every* $\lambda \in \mathbb{N}$*,* $k \in \mathbb{N}$*,* $m \in \mathbb{N}$*, indices* $i_0 \neq i_1 \in [k]$*, adversary size* $t_{\mathsf{BG}} \in \mathbb{N}$*, and* t_{BG}*-time adversary* A*,*

$$\left|\begin{array}{l} \Pr\left[A(\mathsf{pp}_{\mathsf{BG}}) = 1 \;\middle|\; \mathsf{pp}_{\mathsf{BG}} \leftarrow \mathsf{Setup}_{\mathsf{BG}}(1^\lambda, k, 1^m, i_0)\right] \\ -\Pr\left[A(\mathsf{pp}_{\mathsf{BG}}) = 1 \;\middle|\; \mathsf{pp}_{\mathsf{BG}} \leftarrow \mathsf{Setup}_{\mathsf{BG}}(1^\lambda, k, 1^m, i_1)\right] \end{array}\right| \leq \zeta_{\mathsf{BG}}(\lambda, k, t_{\mathsf{BG}})\,.$$

Definition 3.29 (Semi-adaptive somewhere soundness)**.** $\mathsf{BARG} = (\mathsf{Setup}_{\mathsf{BG}}, \mathbf{P}_{\mathsf{BG}}, \mathbf{V}_{\mathsf{BG}})$ *has semi-adaptive somewhere soundness error* ϵ_{BG} *if for every* $\lambda \in \mathbb{N}$*,* $k \in \mathbb{N}$*,* $m \in \mathbb{N}$*, index* $i \in [k]$*, adversary size* $t_{\mathsf{BG}} \in \mathbb{N}$*, and* t_{BG}*-time adversary* $\widetilde{\mathbf{P}}_{\mathsf{BG}}$*,*

$$\Pr\left[\begin{array}{l} \mathbf{V}_{\mathsf{BG}}(\mathsf{pp}_{\mathsf{BG}}, \mathbb{i}, (\mathbb{x}_1, \ldots, \mathbb{x}_k), \widetilde{\pi}_{\mathsf{BG}}) = 1 \\ \wedge\, (\mathbb{i}, \mathbb{x}_i) \notin L(R) \end{array} \;\middle|\; \begin{array}{l} \mathsf{pp}_{\mathsf{BG}} \leftarrow \mathsf{Setup}_{\mathsf{BG}}(1^\lambda, k, 1^m, i) \\ (\mathbb{i}, (\mathbb{x}_1, \ldots, \mathbb{x}_k), \widetilde{\pi}_{\mathsf{BG}}) \leftarrow \widetilde{\mathbf{P}}_{\mathsf{BG}}(\mathsf{pp}_{\mathsf{BG}}) \end{array}\right] \leq \epsilon_{\mathsf{BG}}(\lambda, k, t_{\mathsf{BG}})\,.$$

Theorem 3.30 ([PP22; DGKV22])**.** *Under the sub-exponential hardness of* LWE*, for every* $R \in \mathsf{NP}$ *with witness size* m*, there exists a semi-adaptive somewhere sound* BARG *for* k *instances that satisfies the following:*

- *perfect completeness;*
- *the public parameter* $\mathsf{pp}_{\mathsf{BG}}$ *is of size* $\mathsf{poly}(\lambda, \log k, m)$*;*
- *the proof* π_{BG} *is of size* $O(m + \lambda^{10})$*;*[2]
- *for every* $c > 1$ *and* $t_{\mathsf{BG}} \leq 2^{\lambda^{1/c}}$*, there exists* c' *such that the semi-adaptive somewhere soundness error* $\epsilon_{\mathsf{BG}}(\lambda, t_{\mathsf{BG}}) \leq 2^{-\lambda^{1/c'}}$*;*
- *for every* $c > 1$ *and* $t_{\mathsf{BG}} \leq 2^{\lambda^{1/c}}$*, there exists* c' *such that the index hiding error is at most* $\zeta_{\mathsf{BG}}(\lambda, t_{\mathsf{BG}}) \leq 2^{-\lambda^{1/c'}}$*.*

[2]Constructions of rate-1 BARGs (e.g.,[PP22; DGKV22]) usually write proof size to be $m + \mathsf{poly}(\lambda)$. We trace the parameters in [PP22] and an obvious upper bound on the proof size is $O(m + \lambda^{10})$.

4 PCP shadows

We define and construct PCPs with *shadow soundness*. Informally, this notion requires that, given an instance $\mathbb{x}$ and a (malicious) PCP proof $\widetilde{\Pi}$, there exists an efficiently computable *shadow* z that succinctly represents the PCP verifier's behavior with respect to its randomness: namely, there exists a decider such that, given the instance $\mathbb{x}$ and the shadow z, it outputs the verifier's decision for any verifier randomness ρ with high probability. We then show that there exists a PCP for the canonical NP-complete language $\mathsf{3SAT}$ that satisfies shadow soundness, and whose shadow can be computed by *polynomial-size circuits*.

Definition 4.1 (Shadow soundness)**.** *A* $\mathsf{PCP} = (\mathbf{P}, \mathbf{V})$ *has shadow soundness if there exists a function* F *with randomness complexity* r_{F} *and a deterministic algorithm* D *(a decider) such that for every instance* $\mathbb{x} \notin L(R)$*, the following holds:*

1. ***Correctness.*** *For any* $\widetilde{\Pi}$*, and verifier randomness* $\rho \in \{0,1\}^{\mathsf{r}}$*:*

$$\Pr\left[\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}, \rho) \neq \mathsf{D}(\mathbb{x}, z, \rho) \;\middle|\; \begin{array}{l} \gamma \leftarrow \{0,1\}^{\mathsf{r}_{\mathsf{F}}} \\ z := \mathsf{F}(\mathbb{x}, \widetilde{\Pi}; \gamma) \end{array}\right] \leq \epsilon_{\mathsf{c}}(\mathbb{x})\,.$$

2. ***Sparsity.*** *There exists an efficiently computable set* Z *where* $\mathsf{Im}(\mathsf{F}(\mathbb{x}, \cdot)) \subseteq Z$ *such that for any shadow state* $z \in Z$*:*

$$\Pr\left[\mathsf{D}(\mathbb{x}, z, \rho) = 1 \mid \rho \leftarrow \{0,1\}^{\mathsf{r}}\right] \leq \epsilon_{\mathsf{s}}(\mathbb{x})\,.$$

3. ***Consistency.*** *For every shadow randomness* $\gamma \in \{0,1\}^{\mathsf{r}_{\mathsf{F}}}$*, verifier randomness* $\rho \in \{0,1\}^{\mathsf{r}}$*,* $\widetilde{\Pi}$ *and* $\widetilde{\Pi}'$ *such that* $\widetilde{\Pi}[\mathsf{Q}] = \widetilde{\Pi}'[\mathsf{Q}]$*, where* Q *is the query set of the PCP verifier* $\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho)$*:*

$$\mathsf{D}(\mathbb{x}, z, \rho) = \mathsf{D}(\mathbb{x}, z', \rho)\,,$$

where $z := \mathsf{F}(\mathbb{x}, \widetilde{\Pi}; \gamma)$ *and* $z' := \mathsf{F}(\mathbb{x}, \widetilde{\Pi}'; \gamma)$*.*

Moreover, we say that PCP *has shadow state size* $\mathsf{s}_{\mathsf{PCP}}$ *if* F *outputs at most* $\mathsf{s}_{\mathsf{PCP}}$ *bits.* F *has size at most* t_{F} *and* D *has size at most* t_{D}*.*

In our construction of SNARG (Section 6), we actually rely on a stronger sparsity property, called the *product sparsity*.

Definition 4.2 (Product sparsity)**.** *We say* $\mathsf{PCP} = (\mathbf{P}, \mathbf{V})$ *has shadow soundness with product sparsity with product parameter* t *if there exists a deterministic algorithm* D' *such that for every* $\mathbb{x}$*,* z*, and* $\rho = (\rho_1, \ldots, \rho_t)$ *where* $\rho_i \in \{0,1\}^{\frac{\mathsf{r}}{t}}$*,*

$$\mathsf{D}(\mathbb{x}, z, \rho) = \bigwedge_{i \in [t]} \mathsf{D}'(\mathbb{x}, z, \rho_i)\,.$$

Moreover, there exists an efficiently computable set Z *where* $\mathsf{Im}(\mathsf{F}(\mathbb{x}, \cdot)) \subseteq Z$ *such that for any state* $z \in Z$ *and* $i \in [t]$*:*

$$\Pr\left[\mathsf{D}'(\mathbb{x}, z, \rho_i) = 1 \;\middle|\; (\rho_1, \ldots, \rho_t) \leftarrow \{0,1\}^{t \cdot \frac{\mathsf{r}}{t}}\right] \leq \epsilon_{\mathsf{s}}(\mathbb{x})\,.$$

Remark 4.3. PCP that satisfies product sparsity with error ϵ_{s} and product parameter t also satisfies sparsity with error ϵ_{s}^t. We use product sparsity because in the SNARG construction in Section 6, we use correlation intractable hash functions for sparse product relations (see Theorem 3.8). If there exists correlation intractable hash functions for all sparse relations, it suffices to use plain sparsity.

4.1 Shadow PCP for NP

The theorem below describe the PCP we build to construct the SNARG in Section 6.

Theorem 4.4. *There is a* PCP *for* $\mathsf{3SAT}$ *that satisfies the following:*

- *perfect completeness;*
- *soundness error* $\epsilon_{\mathsf{PCP}} = 2^{-(\log n)^2}$*;*
- *non-adaptive verifier;*
- *query complexity* $\mathsf{q} \leq \tilde{O}(n^{\frac{10}{11}})$*;*
- *randomness complexity* $\mathsf{r} \leq \tilde{O}((\log n)^3)$*;*
- *proof length* $\ell = \tilde{O}(n)$*;*
- *alphabet* $\{0,1\}^{O(1)}$*;*
- *shadow soundness with:*
  1. *shadow state size* $s_{\mathsf{T}} = O(\sqrt{n})$*;*
  2. *shadow circuit size* $t_{\mathsf{T}} = \mathsf{poly}(n)$ *(with depth* $\mathsf{polylog}(n)$*);*
  3. *shadow randomness complexity* $\mathsf{r}_{\mathsf{F}} = 0$*;*
  4. *product sparsity error* $\epsilon_{\mathsf{s}} = 1/2$ *and product parameter* $t = 1$*;*
  5. *correctness error* $\epsilon_{\mathsf{c}} = 0$*;*

Proof. We start by describing the PCP construction, then we present and analyze the circuit that computes its shadow.

PCP construction. Let $\mathsf{PCP} = (\mathbf{P}, \mathbf{V})$ be the PCP for $\mathsf{3SAT}$ in Theorem 3.15. Fix $B \in \mathbb{N}$. Set $t := 2^{\mathsf{r}}/B$. We break $\{0,1\}^{\mathsf{r}}$ into t consecutive blocks of size B: $R_1, \ldots, R_t$. To prove our theorem, we choose B to be $2^{\mathsf{r}/2}$. $\mathsf{PCP}' = (\mathbf{P}', \mathbf{V}')$ works as follows:

- *Generating the proof string:* $\mathbf{P}'$ runs $\mathbf{P}(\mathbb{x}, \mathbb{w})$ to get the PCP proof Π and sends Π to $\mathbf{V}$.
- *Verifier's checks:*
  1. Sample $b \leftarrow [t]$.
  2. For every $\rho \in R_b$: Check if $\mathbf{V}^{\Pi}(\mathbb{x}; \rho) = 1$.

We claim that the soundness error of PCP' is ϵ_{PCP}. Note that by soundness of PCP, there are at most $\epsilon_{\mathsf{PCP}} \cdot 2^{\mathsf{r}}$ many accepting verifier randomness. Therefore, there are at most $\frac{\epsilon_{\mathsf{PCP}} \cdot 2^{\mathsf{r}}}{B} = \epsilon_{\mathsf{PCP}} \cdot t$ many randomness blocks that contain only accepting verifier randomness. Therefore, $\mathbf{V}'$ accepts with probability at most ϵ_{PCP}. Hence, PCP' satisfies the following:

- non-adaptive verifier;
- perfect completeness;
- soundness error ϵ_{PCP};
- proof length ℓ;
- randomness complexity $\log t$;
- query complexity $\mathsf{q} \cdot B$;
- alphabet Σ.

Shadow circuit construction. The following circuit F gives shadow soundness for $\mathsf{3SAT}$:

$\mathsf{F}(\mathbb{x}, \widetilde{\Pi})$:

1. For every PCP verifier randomness $\rho \in \{0,1\}^{\mathsf{r}}$: let $b_\rho := \mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho)$.

2. Initialize $z \coloneqq 0^t$.
3. For every randomness block $b \in [t]$: let $z[b] \coloneqq \bigwedge_{\rho \in R_b} z_\rho$.
4. Output z.

It's easy to see that the circuit size of F is $\mathsf{poly}(n)$. Since the PCP verifier $\mathbf{V}$, given instance $\mathbb{x}$, can be implemented with circuits in NC, F can also be implemented with circuits in NC. Moreover, the shadow state size is $\sqrt{2^{\mathsf{r}}}$ if we choose $B \coloneqq \sqrt{2^{\mathsf{r}}}$.

Correctness. Fix $\widetilde{\Pi}$ and PCP verifier randomness b. F computes exactly the new PCP verifier $\mathbf{V}'$.

Sparsity. Since the soundness error of PCP' is $1/2$ and the correctness error for the shadow circuit is 0, the sparsity error is $1/2$.

Consistency. This follows directly from the structure of PCP'.

□

4.2 Amplification of shadow soundness

We prove a parallel repetition theorem for amplifying shadow soundness of PCP.

Theorem 4.5. *Consider* PCP *that satisfies the following:*

- *perfect completeness;*
- *soundness error* ϵ_{PCP}*;*
- *proof length* ℓ*;*
- *randomness complexity* r*;*
- *query complexity* q*;*
- *alphabet* Σ*;*
- *shadow soundness with:*
  1. *shadow state size* $\mathsf{s}_{\mathsf{PCP}}$*;*
  2. *shadow circuit size* t_{F}*;*
  3. *shadow randomness complexity* r_{F}*;*
  4. *sparsity error* ϵ_{s}*;*
  5. *correctness error* ϵ_{c}*.*

Let $\mathsf{PCP}_t \coloneqq \mathsf{PR}[\mathsf{PCP}, t]$ *be the* t*-wise parallel repetition of* PCP *specified in Construction 4.7.* PCP_t *satisfies the following:*

- *perfect completeness;*
- *soundness error* $\epsilon_{\mathsf{PCP}}^t$*;*
- *proof length* ℓ*;*
- *randomness complexity* $t \cdot \mathsf{r}$*;*
- *query complexity* $t \cdot \mathsf{q}$*;*
- *alphabet* Σ*;*
- *shadow soundness with:*
  1. *shadow state size* $\mathsf{s}_{\mathsf{PCP}}$*;*
  2. *shadow circuit size* t_{F}*;*
  3. *shadow randomness complexity* r_{F}*;*
  4. *product sparsity error* ϵ_{s} *and product parameter* t*;*
  5. *correctness error* $2t \cdot \epsilon_{\mathsf{c}}$*.*

Corollary 4.6. *There is a* PCP *for* 3SAT *that satisfies the following:*

- *perfect completeness;*
- *soundness error* $\epsilon_{\mathsf{PCP}} = 2^{-(\log n)^2}$*;*
- *non-adaptive verifier;*
- *query complexity* $\mathsf{q} \leq \tilde{O}(n^{\frac{10}{11}})$*;*
- *randomness complexity* $\mathsf{r} \leq \tilde{O}((\log n)^3)$*;*
- *proof length* $\ell = \tilde{O}(n)$*;*
- *alphabet* $\{0,1\}^{O(1)}$*;*
- *shadow soundness with:*
  1. *shadow state size* $s_{\mathsf{T}} = O(\sqrt{n})$*;*
  2. *shadow circuit size* $t_{\mathsf{T}} = \mathsf{poly}(n)$*;*
  3. *shadow randomness complexity* $\mathsf{r}_{\mathsf{F}} = 0$*;*
  4. *product sparsity error* $\epsilon_{\mathsf{s}} = 1/2$ *and product parameter* $t = (\log n)^2$*;*
  5. *correctness error* $\epsilon_{\mathsf{c}} = 0$*;*

Construction 4.7. Consider $\mathsf{PCP} = (\mathbf{P}, \mathbf{V})$. We first recall how the t-wise parallel repetition $\mathsf{PCP}_t = (\vec{\mathbf{P}}, \vec{\mathbf{V}})$ works:

- **Generating the PCP proof**: $\vec{\mathbf{P}}$ computes $\Pi \leftarrow \mathbf{P}(\mathbb{x}, \mathbb{w})$ and sends Π to $\vec{\mathbf{V}}$.
- **Verifier's checks**:
  1. $\vec{\mathbf{V}}$ samples $(\rho_1, \ldots, \rho_t) \leftarrow \{0,1\}^{t \cdot \mathsf{r}}$.
  2. $\vec{\mathbf{V}}$ accepts iff for every $i \in [t]$, $\mathbf{V}^{\Pi}(\mathbb{x}; \rho_i) = 1$.

Soundness. We show that the soundness error of PCP_t is $\epsilon_{\mathsf{PCP}}^t$. Let $\widetilde{\Pi}$ be a malicious PCP proof string. We bound the following probability:

$$\begin{aligned}
&\Pr\left[\vec{\mathbf{V}}^{\widetilde{\Pi}}(\mathbb{x}; \rho_1, \ldots, \rho_t) = 1 \;\middle|\; (\rho_1, \ldots, \rho_t) \leftarrow \{0,1\}^{t \cdot \mathsf{r}}\right] \\
&= \left(\Pr\left[\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho) = 1 \;\middle|\; \rho \leftarrow \{0,1\}^{\mathsf{r}}\right]\right)^t \\
&\leq \epsilon_{\mathsf{PCP}}^t \,.
\end{aligned}$$

Tree algorithm. Let T be the tree algorithm for PCP and D be the decider. We construct $\mathsf{T}_t, \mathsf{D}_t$ for PCP_t as follows:

- $\mathsf{T}_t(\mathbb{x}, \Pi; \gamma)$: Output $z \coloneqq \mathsf{T}(\mathbb{x}, \Pi; \gamma)$.
- $\mathsf{D}_t(\mathbb{x}, z, \rho_i)$: Output 1 iff $\mathsf{D}(\mathbb{x}, z, \rho_i) = 1$.

We first analyze the product sparsity. Fix $z \in \mathsf{Im}(\mathsf{T}_t)$. For every $i \in [t]$, we bound the following probability:

$$\Pr\left[\mathsf{D}_t(\mathbb{x}, z, \rho_i) = 1 \;\middle|\; (\rho_1, \ldots, \rho_t) \leftarrow \{0,1\}^{t \cdot \mathsf{r}}\right] \leq \epsilon_{\mathsf{s}} \,.$$

Now we analyze correctness. Fix $\widetilde{\Pi}$ and verifier randomness $(\rho_1, \ldots, \rho_t) \leftarrow \{0,1\}^{t \cdot \mathsf{r}}$. We bound the following probability:

$$\Pr\left[\vec{\mathbf{V}}^{\widetilde{\Pi}}(\mathbb{x}; \rho_1, \ldots, \rho_t) \neq \bigwedge_{i \in [t]} \mathsf{D}_t(\mathbb{x}, z, \rho_i) \;\middle|\; \begin{array}{l} \gamma \leftarrow \{0,1\}^{\mathsf{r}_{\mathsf{T}}} \\ z \coloneqq \mathsf{T}_t(\mathbb{x}, \widetilde{\Pi}; \gamma) \end{array}\right]$$

$$
\begin{aligned}
&= \Pr\left[\begin{array}{l|l} \forall i \in [t], \mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho_i) = 1 & \gamma \leftarrow \{0,1\}^{r_\mathsf{T}} \\ \wedge\, \exists i \in [t], \mathsf{D}(\mathbb{x}, z, \rho_i) = 0 & z \coloneqq \mathsf{T}(\mathbb{x}, \widetilde{\Pi}; \gamma) \end{array}\right] \\
&\quad + \Pr\left[\begin{array}{l|l} \exists i \in [t], \mathbf{V}^{\widetilde{\Pi}}(\mathbb{x}; \rho_i) = 0 & \gamma \leftarrow \{0,1\}^{r_\mathsf{T}} \\ \wedge\, \forall i \in [t], \mathsf{D}(\mathbb{x}, z, \rho_i) = 1 & z \coloneqq \mathsf{T}(\mathbb{x}, \widetilde{\Pi}; \gamma) \end{array}\right] \\
&\leq 2t \cdot \epsilon_\mathsf{c} \,.
\end{aligned}
$$

5 Function vector commitment scheme

We introduce a new notion of vector commitment schemes, called *function vector commitment (VC) scheme*, that allow one to commit to a vector m relative a function f. Specifically, *(statistical) binding* requires that for every $m \neq m'$ such that $f(m) \neq f(m')$, the corresponding commitments cm and cm' are distinct. In fact, we enforce a stronger property, called *function statistically binding*, which requires that, given a trapdoor, a commitment and a set of accepting local openings, an (inefficient) extractor can recover m that is consistent with both the commitment and the local openings.

We begin by formally defining function VC in Section 5.1. Then, in Section 5.2, we construct a function VC for polynomial-size circuits.

5.1 Definition

A *function vector commitment (VC) scheme* over alphabet Σ for a family of function $\mathcal{F}$ is a tuple of algorithms

$$\mathsf{VC} = (\mathsf{VC.Gen}, \mathsf{VC.Commit}, \mathsf{VC.Open}, \mathsf{VC.Check}, \mathsf{VC.Extract})$$

with the following syntax:

- $\mathsf{VC.Gen}(1^\lambda, f) \to (\mathsf{pp}, \mathsf{td})$: On input a security parameter $\lambda \in \mathbb{N}$ and description of a function f, $\mathsf{VC.Gen}$ samples public parameter pp and a corresponding trapdoor td.
- $\mathsf{VC.Commit}(\mathsf{pp}, m) \to (\mathsf{cm}, \mathsf{aux})$: On input a public parameter pp and a message m, $\mathsf{VC.Commit}$ produces a commitment cm and the corresponding auxiliary state aux.
- $\mathsf{VC.Open}(\mathsf{pp}, \mathsf{aux}, \mathsf{Q}) \to \mathsf{pf}$: On input a public parameter pp, an auxiliary state aux, and a query set $\mathsf{Q} \subseteq [\ell]$, $\mathsf{VC.Open}$ outputs an opening proof string pf attesting that $m[\mathsf{Q}]$ is a restriction of m to Q.
- $\mathsf{VC.Check}(\mathsf{pp}, \mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf}) \to \{0,1\}$: On input a public parameter pp, a commitment cm, a query set $\mathsf{Q} \subseteq [\ell]$, an answer string $\mathsf{a} \in \Sigma^{\mathsf{Q}}$, and an opening proof string pf, $\mathsf{VC.Check}$ determines if pf is a valid proof for $\mathsf{a} \in \Sigma^{\mathsf{Q}}$ being a restriction of the message committed in cm to Q.
- $\mathsf{VC.Extract}(\mathsf{td}, \mathsf{cm}) \to y$: On input the trapdoor td and a commitment cm, $\mathsf{VC.Extract}$ outputs deterministically a string y.

Moreover, VC has the following property.

Definition 5.1 (Completeness)**.** VC *has* **perfect completeness** *if for every security parameter $\lambda \in \mathbb{N}$, message $m \in \Sigma^\ell$, auxiliary function $f \in \mathcal{F}$, and query set $\mathcal{Q} \subseteq [\ell]$,*

$$\Pr\left[\mathsf{VC.Check}(\mathsf{pp}, \mathsf{cm}, \mathsf{Q}, m[\mathsf{Q}], \mathsf{pf}) = 1 \;\middle|\; \begin{array}{l} (\mathsf{pp}, \mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, f) \\ (\mathsf{cm}, \mathsf{aux}) \leftarrow \mathsf{VC.Commit}(\mathsf{pp}, m) \\ \mathsf{pf} \leftarrow \mathsf{VC.Open}(\mathsf{pp}, \mathsf{aux}, \mathsf{Q}) \end{array}\right] = 1\,.$$

Definition 5.2 (Function hiding)**.** VC *is* **functional-hiding** *if for every security parameter $\lambda \in \mathbb{N}$, auxiliary functions $f, f' \in \mathcal{F}$, and t_{vc}-size adversary A_{vc},*

$$\left|\Pr\left[A_{\mathsf{vc}}(\mathsf{pp}) = 1 \;\middle|\; (\mathsf{pp}, \mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, f)\right] - \Pr\left[A_{\mathsf{vc}}(\mathsf{pp}) = 1 \;\middle|\; (\mathsf{pp}, \mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, f')\right]\right| \leq z_{\mathsf{vc}}(\lambda, t_{\mathsf{vc}})\,.$$

Definition 5.3 (Function statistically binding)**.** VC *is* **function statistically binding** *if for every security parameter $\lambda \in \mathbb{N}$, auxiliary function $f \in \mathcal{F}$, adversary size $t_{\mathsf{VC}} \in \mathbb{N}$, and t_{VC}-size adversary A_{VC}, the following holds:*

$$\Pr\left[\begin{array}{l|l} \mathsf{VC.Check}(\mathsf{pp},\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}) = 1 & (\mathsf{pp},\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, f) \\ \wedge \forall\, \widetilde{m} : (\widetilde{m}[\mathsf{Q}] \neq \mathsf{a} \ \vee\ f(\widetilde{m}) \neq y) & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}) \coloneqq A_{\mathsf{VC}}(\mathsf{pp}) \\ & y \coloneqq \mathsf{VC.Extract}(\mathsf{td},\mathsf{cm}) \end{array}\right] \leq \kappa_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}})\,.$$

Similarly to the FHE randomness hiding property Lemma 3.25, the following lemma states that no efficient adversary can guess the randomness of the function with high probability. This property is useless in the analysis of SNARG soundness in Section 6.2.

Lemma 5.4 (Randomness hiding)**.** *Fix* VC*. For every $\lambda \in \mathbb{N}$, randomized function f, adversary size $t_{\mathsf{VC}} \in \mathbb{N}$, and t_{VC}-time adversary A, the following holds:*

$$\Pr\left[\rho' = \rho \;\middle|\; \begin{array}{l} \rho \leftarrow \{0,1\}^r \\ (\mathsf{pp},\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, f_\rho) \\ \rho' \leftarrow A(\mathsf{pp}) \end{array}\right] \leq \frac{1}{2^r} + z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}})\,.$$

Proof. Assume for contradiction that there exists an adversary A such that the probability in the lemma statement is greater than $\frac{1}{2^r} + z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}})$. We construct an adversary A' that breaks the function hiding of VC. Let ρ be a randomly sampled element from $\{0,1\}^r$.

$A'(\mathsf{pp})$:
1. Run $\rho' \leftarrow A(\mathsf{pp})$.
2. If ρ' is such that $f_\rho = f_{\rho'}$, output 1.
3. Otherwise, output 0.

The following holds:

$$\begin{aligned} &\left|\Pr\left[b = 1 \;\middle|\; \begin{array}{l} \rho \leftarrow \{0,1\}^r \\ (\mathsf{pp},\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, \mathbf{0}) \\ b \leftarrow A'(\mathsf{pp}) \end{array}\right] - \Pr\left[b = 1 \;\middle|\; \begin{array}{l} \rho \leftarrow \{0,1\}^r \\ (\mathsf{pp},\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, f_\rho) \\ b \leftarrow A'(\mathsf{pp}) \end{array}\right]\right| \\ &= \left|\Pr\left[b = 1 \;\middle|\; \begin{array}{l} \rho \leftarrow \{0,1\}^r \\ (\mathsf{pp},\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, f_\rho) \\ b \leftarrow A'(\mathsf{pp}) \end{array}\right] - \frac{1}{2^r}\right| \\ &> \frac{1}{2^r} + z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}}) - \frac{1}{2^r} \\ &= z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}})\,, \end{aligned}$$

where the equality holds since when pp is generated using $\mathbf{0}$, it contains no information about ρ, and thus the output ρ' of A' is independent from sampling ρ and thus $\rho = \rho'$ with probability exactly 2^{-r}, and the inequality holds because of our assumption. □

5.2 Function VC for circuits

Theorem 5.5. *Assume iO and LWE are the sub-exponentially secure. There exists* VC *for* $\mathsf{P/poly}$ *with circuit size t and state size s such that for every security parameter $\lambda \in \mathbb{N}$, $\lambda_{\mathsf{FHE}} = \lambda_{\mathsf{FHE}}(\lambda)$, and $\lambda_{\mathsf{SNARG}} = \lambda_{\mathsf{SNARG}}(\lambda)$* VC *satisfies the following:*

- *perfect completeness;*
- *for every* $c > 1$ *and* $t_{\mathsf{VC}} \leq 2^{\lambda_{\mathsf{FHE}}^{1/c}}$*, there exists* c' *such that the function hiding error* $z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}}) \leq d \cdot 2^{-\lambda_{\mathsf{FHE}}^{1/c'}}$*;*
- *for every* $c > 1$ *and* t_{VC} *where* $t_{\mathsf{VC}} \leq 2^{\lambda_{\mathsf{SNARG}}^{1/c}}$*, there exists* c' *such that the function statistically binding error is at most* $\kappa_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}}) \leq \mathrm{negl}(\lambda_{\mathsf{SNARG}})$*;*
- *commitment size* $s \cdot \mathsf{poly}(\lambda_{\mathsf{FHE}})$*;*
- *public parameter size* $\mathsf{poly}(\lambda, \lambda_{\mathsf{SNARG}}, \lambda_{\mathsf{FHE}}, t)$*;*
- *trapdoor size* $\mathsf{poly}(\lambda_{\mathsf{FHE}})$*;*
- *proof size* $\mathsf{poly}(\lambda_{\mathsf{SNARG}})$.

Let $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Dec})$ be the FHE in Theorem 3.23. For every x, let $\mathcal{U}_x$ be the universal circuit such that on input a circuit description f, $\mathcal{U}_x(f) \coloneqq f(x)$. Let $\mathsf{SNARG} = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ be the SNARG from Theorem 3.12 for the following language L in NP:

$$L \coloneqq \{(\mathsf{pp} = (\mathsf{pk}, \mathsf{ct}_f), \mathsf{cm}, \mathsf{Q}, \mathsf{a}) : \exists\, m, m[\mathsf{Q}] = \mathsf{a}, \mathsf{FHE.Eval}(\mathsf{pk}, \mathcal{U}_m, \mathsf{ct}_f) = \mathsf{cm}\}\,.$$

We construct VC for circuits as follows:

- $\mathsf{VC.Gen}(1^\lambda, f)$:
  1. Compute $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.Gen}(1^{\lambda_{\mathsf{FHE}}})$.
  2. Compute $\mathsf{ct}_f \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, f)$.
  3. Compute $\mathsf{pp}_{\mathsf{SNARG}} \leftarrow \mathsf{Setup}(1^{\lambda_{\mathsf{SNARG}}})$.
  4. Output $(\mathsf{pp} \coloneqq (\mathsf{pk}, \mathsf{ct}_f, \mathsf{pp}_{\mathsf{SNARG}}), \mathsf{td} \coloneqq \mathsf{sk})$.
- $\mathsf{VC.Commit}(\mathsf{pp}, m)$:
  1. Parse pp as $(\mathsf{pk}, \mathsf{ct}_f, \mathsf{pp}_{\mathsf{SNARG}})$.
  2. Compute $\mathsf{cm} \coloneqq \mathsf{FHE.Eval}(\mathsf{pk}, \mathcal{U}_m, \mathsf{ct}_f)$.
  3. Output $(\mathsf{cm}, \mathsf{aux} \coloneqq (\mathsf{cm}, m))$.
- $\mathsf{VC.Open}(\mathsf{pp}, \mathsf{aux}, \mathsf{Q})$:
  1. Parse pp as $(\mathsf{pk}, \mathsf{ct}_f, \mathsf{pp}_{\mathsf{SNARG}})$.
  2. Set $\mathsf{a} \coloneqq m[\mathsf{Q}]$.
  3. Output $\mathsf{pf} \leftarrow \mathcal{P}(\mathsf{pp}_{\mathsf{SNARG}}, \mathsf{cm}, \mathsf{Q}, \mathsf{a})$.
- $\mathsf{VC.Check}(\mathsf{pp}, \mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf})$:
  1. Parse pp as $(\mathsf{pk}, \mathsf{ct}_f, \mathsf{pp}_{\mathsf{SNARG}})$.
  2. Output $\mathcal{V}((\mathsf{pp}_{\mathsf{SNARG}}, \mathsf{cm}, \mathsf{Q}, \mathsf{a}), \mathsf{pf})$.
- $\mathsf{VC.Extract}(\mathsf{td}, \mathsf{cm})$:
  1. Parse td as sk.
  2. Output $\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{cm})$.

Completeness. Completeness follows from completeness of SNARG.

Function hiding. $z_{\mathsf{VC}}(\lambda, A_{\mathsf{VC}}) \leq \epsilon_{\mathsf{FHE}}(\lambda, A_{\mathsf{VC}})$ from semantic security of FHE.

Function statistically binding. We bound the following probability:

$$
\begin{aligned}
&\Pr\left[\begin{array}{l|l} \mathsf{VC.Check}(\mathsf{pp},\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})=1 & ((\mathsf{pk},\mathsf{ct}_f,\mathsf{pp}_{\mathsf{SNARG}}),\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda,f) \\ \wedge \forall\, \widetilde{m} : (\widetilde{m}[\mathsf{Q}] \neq \mathsf{a} \ \vee\ f(\widetilde{m}) \neq y) & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}) \coloneqq A_{\mathsf{VC}}(\mathsf{pp}) \\ & y \coloneqq \mathsf{VC.Extract}(\mathsf{td},\mathsf{cm}) \end{array}\right] \\
&= \Pr\left[\begin{array}{l|l} \mathcal{V}((\mathsf{pp}_{\mathsf{SNARG}},\mathsf{cm},\mathsf{Q},\mathsf{a}),\mathsf{pf})=1 & ((\mathsf{pk},\mathsf{ct}_f,\mathsf{pp}_{\mathsf{SNARG}}),\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda,f) \\ \wedge \forall\, \widetilde{m} : (\widetilde{m}[\mathsf{Q}] \neq \mathsf{a} \ \vee\ f(\widetilde{m}) \neq y) & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}) \coloneqq A_{\mathsf{VC}}(\mathsf{pp}) \\ & y \coloneqq \mathsf{VC.Extract}(\mathsf{td},\mathsf{cm}) \end{array}\right] \\
&= \Pr\left[\begin{array}{l|l} \mathcal{V}((\mathsf{pp}_{\mathsf{SNARG}},\mathsf{cm},\mathsf{Q},\mathsf{a}),\mathsf{pf})=1 & ((\mathsf{pk},\mathsf{ct}_f,\mathsf{pp}_{\mathsf{SNARG}}),\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda,f) \\ \wedge \forall\, \widetilde{m} : (\widetilde{m}[\mathsf{Q}] \neq \mathsf{a} \ \vee\ f(\widetilde{m}) \neq y) & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}) \coloneqq A_{\mathsf{VC}}(\mathsf{pp}) \\ \wedge \exists\, m : (m[\mathsf{Q}] = \mathsf{a} \ \wedge\ \mathsf{FHE.Eval}(\mathsf{pk},\mathcal{U}_m,\mathsf{ct}_f) = \mathsf{cm}) & y \coloneqq \mathsf{VC.Extract}(\mathsf{td},\mathsf{cm}) \end{array}\right] \\
&\quad + \Pr\left[\begin{array}{l|l} \mathcal{V}((\mathsf{pp}_{\mathsf{SNARG}},\mathsf{cm},\mathsf{Q},\mathsf{a}),\mathsf{pf})=1 & (\mathsf{pp},\mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda,f) \\ \wedge \forall\, \widetilde{m} : (\widetilde{m}[\mathsf{Q}] \neq \mathsf{a} \ \vee\ f(\widetilde{m}) \neq y) & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}) \coloneqq A_{\mathsf{VC}}(\mathsf{pp}) \\ \wedge \forall\, m : (m[\mathsf{Q}] \neq \mathsf{a} \ \vee\ \mathsf{FHE.Eval}(\mathsf{pk},\mathcal{U}_m,\mathsf{ct}_f) \neq \mathsf{cm}) & y \coloneqq \mathsf{VC.Extract}(\mathsf{td},\mathsf{cm}) \end{array}\right] \\
&= 0 + \epsilon_{\mathsf{SNARG}}(\lambda, A_{\mathsf{VC}})\,.
\end{aligned}
$$

6 SNARG from function VC and shadow PCP

Theorem 6.1. *Fix a family of functions $\mathcal{F}$. Let* PCP *be a shadow PCP for* 3SAT *with a shadow function in $\mathcal{F}$ that satisfies the following:*

- *perfect completeness;*
- *query complexity* q*;*
- *shadow soundness with:*
  1. *shadow state size* s_{T}*;*
  2. *shadow circuit size* t_{T}*;*
  3. *shadow randomness complexity* r_{F}*;*
  4. *product sparsity error* ϵ_{s} *and product parameter* t*;*
  5. *correctness error* ϵ_{c}*.*

Let VC *be a function VC for $\mathcal{F}$ that satisfies the following:*

- *perfect completeness;*
- *the function hiding error* $z_{\mathsf{VC}} = z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}})$*;*
- *function statistically binding error* $\kappa_{\mathsf{VC}} = \kappa_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}})$*;*
- *commitment size* s_{cm}*;*
- *public parameter size* s_{pp}*;*
- *proof size* s_{pf}*.*

Let $\mathcal{H}_{\mathsf{CI}}$ be a correlation-intractable hash family for sparse product relations that satisfies the following:

- *error* $\epsilon_{\mathsf{CI}} = \epsilon_{\mathsf{CI}}(\lambda, t_{\mathsf{CI}})$*;*
- *hash key size* s_{CI}*.*

Let SNARG *be the argument constructed in Section 6.1. For every $\lambda \in \mathbb{N}$, instance $\mathbb{x} \notin L(R)$, and circuit size bound $t_{\mathsf{SNARG}} \in \mathbb{N}$, assume $t_{\mathsf{SNARG}} = \mathsf{poly}(n)$, the non-adaptive soundness error $\epsilon_{\mathsf{SNARG}}$ of* SNARG *satisfies:*

$$\epsilon_{\mathsf{SNARG}}(\lambda, t_{\mathsf{SNARG}}) \leq z_{\mathsf{VC}}(\lambda, O(t_{\mathsf{SNARG}})) + \kappa_{\mathsf{VC}}(\lambda, O(t_{\mathsf{SNARG}})) + \epsilon_{\mathsf{CI}}(\lambda, O(t_{\mathsf{SNARG}})) + \epsilon_{\mathsf{c}} + \epsilon_{\mathsf{c}} \cdot 2^{\mathsf{r}_{\mathsf{F}}} \cdot z_{\mathsf{VC}}(t_{\mathsf{VC}}) \,,$$

where $t_{\mathsf{VC}} = O(t_{\mathsf{SNARG}} + 2^{\mathsf{r}_{\mathsf{F}}} \cdot \mathsf{s}_{\mathsf{PCP}} + \mathsf{t}_{\mathsf{D}})$. The proof size as *of* SNARG *satisfies:*

$$\mathsf{as} \leq s_{\mathsf{cm}} + \mathsf{q} \cdot \log \ell + \mathsf{q} \cdot \log |\Sigma| + \mathsf{s}_{\mathsf{pf}} \,.$$

The public parameter size s_{pp} of SNARG *satisfies:*

$$s_{\mathsf{pp}} \leq s_{\mathsf{pp}} + s_{\mathsf{CI}} \,.$$

Corollary 6.2. *Assume iO and LWE are sub-exponentially hard. The Micali SNARG has non-adaptive soundness error $\epsilon_{\mathsf{SNARG}} = \mathrm{negl}(\lambda)$, argument size $\tilde{O}(\sqrt{n}) \cdot \mathsf{poly}(\lambda)$, and public parameter size $\mathsf{poly}(\lambda, n)$.*

Proof. We apply Theorem 6.1 with the PCP from Corollary 4.6, the VC from Theorem 5.5, and the CIH from Theorem 3.8. Set $\lambda_{\mathsf{FHE}} \coloneqq \lambda$ and $\lambda_{\mathsf{SNARG}} \coloneqq \lambda$.

Soundness error. Since $t_{\mathsf{SNARG}} = \mathsf{poly}(n) < 2^{\lambda_{\mathsf{FHE}}^{1/c}}$ for some $c > 1$, $z_{\mathsf{VC}}(\lambda, t_{\mathsf{SNARG}}) = \mathrm{negl(n)}$. Similarly, $\kappa_{\mathsf{VC}}(\lambda, t_{\mathsf{SNARG}}) = \mathrm{negl(n)}$. We know that $\epsilon_{\mathsf{CI}}(\lambda, t_{\mathsf{SNARG}}) = \mathrm{negl}(\lambda)$ because $\epsilon_{\mathsf{s}} = 1/2$ with product parameter $\mathsf{polylog}(n)$. Also, $\epsilon_{\mathsf{c}} = 0$.

Proof size. We bound the size of cm, (Q, a), and pf separately.

- *Commitment size:* Note that the PCP shadow has state size $O(\sqrt{n})$. According to Theorem 5.5, the size of the commitment cm is $\sqrt{n} \cdot \mathsf{poly}(\lambda_{\mathsf{FHE}}) = O(\sqrt{n}) \cdot \mathsf{poly}(\lambda)$.
- *Query and answer size:* PCP has proof length $\ell = \tilde{O}(n)$, query complexity $\mathsf{q} \leq O(\sqrt{n})$, and alphabet $\{0,1\}^{O(1)}$. Hence, we need $O(\mathsf{q} \cdot \log \ell) = \tilde{O}(\sqrt{n})$ bits to represent Q and $O(\mathsf{q}) = O(\sqrt{n})$ bits to represent a. In total, $|\mathsf{Q}| + |\mathsf{a}| = \tilde{O}(\sqrt{n})$.
- *Opening proof size:* According to Theorem 5.5, the size of pf is $\mathsf{poly}(\lambda_{\mathsf{SNARG}}) = \mathsf{poly}(\log n, \lambda)$.

Therefore, the argument proof size is $\tilde{O}(\sqrt{n}) \cdot \mathsf{poly}\lambda$.

Public parameter size. It is obvious that the size of public parameter is dominated by the size of public parameter of VC, which is $\mathsf{poly}(\lambda, n)$ by Theorem 5.5.

□

6.1 SNARG construction

We construct $\mathsf{SNARG} = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ as follows:

- $\mathsf{Setup}(1^\lambda)$:
  1. Sample $(\mathsf{pp}, \mathsf{td}) \leftarrow \mathsf{VC.Gen}(1^\lambda, \{0^{s_\mathsf{T}}\})$.
  2. Sample $h_{\mathsf{CI}} \leftarrow \mathcal{H}_{\mathsf{CI}}(\lambda)$.
  3. Output $\mathsf{pp} := (\mathsf{pp}, h_{\mathsf{CI}})$.
- $\mathcal{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w})$:
  1. Parse pp as $(\mathsf{pp}, h_{\mathsf{CI}})$.
  2. Compute $\Pi \leftarrow \mathbf{P}(\mathbb{x}, \mathbb{w})$.
  3. Compute $(\mathsf{cm}, \mathsf{aux}) := \mathsf{VC.Commit}(\mathsf{pp}, \Pi)$.
  4. Compute $\rho := h_{\mathsf{CI}}(\mathsf{cm})$.
  5. Compute the PCP verifier query set Q using $\mathbb{x}$, ρ and Π.
  6. Compute $\mathsf{pf} := \mathsf{VC.Open}(\mathsf{pp}, \mathsf{aux}, \mathsf{Q})$ and $\mathsf{a} := \Pi[\mathsf{Q}]$.
  7. Output $\pi := (\mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf})$.
- $\mathcal{V}(\mathsf{pp}, \mathbb{x}, \pi)$:
  1. Parse pp as $(\mathsf{pp}, h_{\mathsf{CI}})$.
  2. Parse π as $(\mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf})$.
  3. Check the following:
     - $\rho = h_{\mathsf{CI}}(\mathsf{cm})$;
     - $\mathsf{VC.Check}(\mathsf{pp}, \mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf}) = 1$;
     - $\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x}; \rho) = 1$.
  4. If any of the above checks fail, output 0, otherwise, output 1.

6.2 SNARG soundness

Let F and D be the PCP function and decider, respectively. Let $\widetilde{\mathcal{P}}$ be an adversary for the SNARG with running time t_{SNARG}. Our goal is to bound the following probability:

$$\Pr\left[\mathcal{V}(\mathsf{pp}, \mathbb{x}, \pi, \rho) = 1 \;\middle|\; \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ \pi := \widetilde{\mathcal{P}}(\mathsf{pp}) \\ \rho := h_{\mathsf{CI}}(\mathsf{cm}) \end{array}\right].$$

By definition of the construction, the above can be written as

$$\Pr\left[\begin{array}{l|l} \mathsf{VC.Check}(\mathsf{pp},\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})=1 & (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,\{0^{s_\mathsf{T}}\}) \\ \wedge\,\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1 & h_\mathsf{CI}\leftarrow\mathcal{H}_\mathsf{CI}(\lambda) \\ & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}):=\widetilde{\mathcal{P}}(\mathsf{pp},h_\mathsf{CI}) \\ & \rho:=h_\mathsf{CI}(\mathsf{cm}) \end{array}\right].$$

Consider the following adversary A_VC for function hiding of VC:

$A_\mathsf{VC}(\mathsf{pp})$:
1. Sample $h_\mathsf{CI}\leftarrow\mathcal{H}_\mathsf{CI}(\lambda)$.
2. Compute $(\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}):=\widetilde{\mathcal{P}}(\mathsf{pp},h_\mathsf{CI})$.
3. Compute $\rho:=h_\mathsf{CI}(\mathsf{cm})$.
4. Output 1 iff the following checks pass:
   (a) $\mathsf{VC.Check}(\mathsf{pp},\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})=1$;
   (b) $\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x};\rho)=1$.

The running time of A_VC is $O(t_\mathsf{SNARG})$. By function hiding of VC, the following holds:

$$\begin{aligned}
&\Pr\left[\begin{array}{l|l} \mathsf{VC.Check}(\mathsf{pp},\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})=1 & (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,\{0^{s_\mathsf{T}}\}) \\ \wedge\,\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x};\rho)=1 & h_\mathsf{CI}\leftarrow\mathcal{H}_\mathsf{CI}(\lambda) \\ & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}):=\widetilde{\mathcal{P}}(\mathsf{pp},h_\mathsf{CI}) \\ & \rho:=h_\mathsf{CI}(\mathsf{cm}) \end{array}\right] \\
&=\Pr\left[A_\mathsf{VC}(\mathsf{pp})=1 \;\middle|\; (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,\{0^{s_\mathsf{T}}\})\right] \\
&\le\Pr\left[A_\mathsf{VC}(\mathsf{pp})=1 \;\middle|\; (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,\ell,\mathsf{F}(\mathbb{x},\cdot))\right]+z_\mathsf{VC}(\lambda,O(t_\mathsf{SNARG})) \\
&=\Pr\left[\begin{array}{l|l} \mathsf{VC.Check}(\mathsf{pp},\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})=1 & (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,\ell,\mathsf{F}(\mathbb{x},\cdot)) \\ \wedge\,\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x};\rho)=1 & h_\mathsf{CI}\leftarrow\mathcal{H}_\mathsf{CI}(\lambda) \\ & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}):=\widetilde{\mathcal{P}}(\mathsf{pp},h_\mathsf{CI}) \\ & \rho:=h_\mathsf{CI}(\mathsf{cm}) \end{array}\right]+z_\mathsf{VC}(\lambda,O(t_\mathsf{SNARG}))\,.
\end{aligned}$$

Using Definition 5.3, the following upper bounds the probability above:

$$\Pr\left[\begin{array}{l|l} \mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x};\rho)=1 & (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,\ell,\mathsf{F}(\mathbb{x},\cdot)) \\ \wedge\,\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\;\wedge\;\mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma)=z & h_\mathsf{CI}\leftarrow\mathcal{H}_\mathsf{CI}(\lambda) \\ & (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}):=\widetilde{\mathcal{P}}(\mathsf{pp},h_\mathsf{CI}) \\ & \rho:=h_\mathsf{CI}(\mathsf{cm}) \\ & z:=\mathsf{VC.Extract}(\mathsf{td},\mathsf{cm}) \end{array}\right]+\kappa_\mathsf{VC}(\lambda,O(t_\mathsf{SNARG}))\,.$$

Let us define the experiment above as Exp. Formally, we have:

$$\mathsf{Exp}:=\left[\begin{array}{l} (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,\ell,\mathsf{F}(\mathbb{x},\cdot)) \\ h_\mathsf{CI}\leftarrow\mathcal{H}_\mathsf{CI}(\lambda) \\ (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}):=\widetilde{\mathcal{P}}(\mathsf{pp},h_\mathsf{CI}) \\ \rho=(\rho_1,\ldots,\rho_t):=h_\mathsf{CI}(\mathsf{cm}) \\ z:=\mathsf{VC.Extract}(\mathsf{td},\mathsf{cm}) \end{array}\right].$$

By the law of total probability, the above probability is equal to:

$$\Pr_{\mathsf{Exp}}\left[\begin{array}{l} \mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1 \\ \wedge\,\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\;\wedge\;\mathsf{F}(\mathbb{x},\widetilde{\Pi})=z \\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=1 \end{array}\right]+\Pr_{\mathsf{Exp}}\left[\begin{array}{l} \mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1 \\ \wedge\,\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\;\wedge\;\mathsf{F}(\mathbb{x},\widetilde{\Pi})=z \\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=0 \end{array}\right].$$

We use the security of the CIH and the function VC to bound the above probabilities, respectively.

6.2.1 Security from CIH

Claim 6.3. *Let ϵ_{CI} be the error for the CI hash family $\mathcal{H}_{\mathsf{CI}}$, the following holds:*

$$\Pr_{\mathsf{Exp}}\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\ \wedge\ \mathsf{F}(\mathbb{x},\widetilde{\Pi})=z\\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=1\end{array}\right]\leq\epsilon_{\mathsf{CI}}(\lambda,O(t_{\mathsf{SNARG}}))\,.$$

Proof. Using the product sparsity of PCP shadow soundness (Definition 4.2), there exists a deterministic algorithm D' such that for every $\mathbb{x}$, z, and $\rho=(\rho_1,\ldots,\rho_t)\in\{0,1\}^{t\cdot\frac{\mathsf{r}}{t}}$,

$$\mathsf{D}(\mathbb{x},z,\rho)=\bigwedge_{i\in[t]}\mathsf{D}'(\mathbb{x},z,\rho_i)\,.$$

For every trapdoor td sampled by $\mathsf{VC.Gen}$, we define a sparse relation $R_{\mathbb{x},\mathsf{td}}$:

$$R_{\mathbb{x},\mathsf{td}}\coloneqq\left\{(\mathsf{cm},\rho):\ \begin{array}{l}z\coloneqq\mathsf{VC.Extract}(\mathsf{td},\mathsf{cm})\\ z\in Z\\ \forall i\in[t],\mathsf{D}'(\mathbb{x},z,\rho_i)=1\end{array}\right\}.$$

We show that for any $\mathbb{x}\notin L(R)$, the relation $R_{\mathbb{x},\mathsf{td}}$ is a ϵ_{s}-sparse product relation. For any $\mathsf{cm}\in L(R_{\mathbb{x},\mathsf{td}})$, (i.e., $\mathsf{VC.Extract}(\mathsf{td},\mathsf{cm})\in Z$), its witness set can be written as follows:

$$\{\rho=(\rho_1,\ldots,\rho_t)\in\{0,1\}^{\mathsf{r}}:\forall i\in[t],\mathsf{D}'(\mathbb{x},z,\rho_i)=1\}=S^t\,,$$

where $S\coloneqq\{\rho\in\{0,1\}^{\mathsf{r}/t}:\mathsf{D}(\mathbb{x},z,\rho)=1\}$. Moreover, by the product sparsity of the shadow soundness of PCP,

$$\begin{aligned}&\Pr\left[(\mathsf{cm},\rho_i)\in S\ \middle|\ \rho_i\leftarrow\{0,1\}^{\mathsf{r}/t}\right]\\ =&\Pr\left[\mathsf{D}'(\mathbb{x},z,\rho_i)=1\ \middle|\ \begin{array}{l}z\coloneqq\mathsf{VC.Extract}(\mathsf{td},\mathsf{cm})\\ \rho_i\leftarrow\{0,1\}^{\mathsf{r}/t}\end{array}\right]\\ \leq&\epsilon_{\mathsf{s}}\,.\end{aligned}$$

We construct an adversary $\mathsf{A}_{\mathbb{x}}$ against the CIH hash using the relation $R_{\mathbb{x},\mathsf{td}}$ as follows:

- $\mathsf{A}_{\mathbb{x}}$:
  1. Sample $\gamma\leftarrow\{0,1\}^{\mathsf{r}_{\mathsf{F}}}$.
  2. Sample $(\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^\lambda,\mathsf{F}(\mathbb{x},\cdot;\gamma))$.
  3. Output $\mathsf{aux}\coloneqq(\mathsf{pp},\mathsf{td})$.
- $\mathsf{A}_{\mathbb{x}}(\mathsf{aux},h_{\mathsf{CI}})$:
  1. Parse aux as $(\mathsf{pp},\mathsf{td})$.
  2. Compute $(\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf})\coloneqq\widetilde{\mathcal{P}}(\mathsf{pp},h_{\mathsf{CI}})$.
  3. Output cm.

We can conclude that:

$$\begin{aligned}
&\Pr_{\mathsf{Exp}}\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\ \wedge\ \mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma)=z\\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=1\end{array}\right]\\
&\leq\Pr\left[(\mathsf{cm},\rho)\in R_{\mathbb{x},\mathsf{td}}\;\middle|\;\begin{array}{l}(\mathsf{pp},\mathsf{td})\leftarrow\mathsf{A}_{\mathbb{x}}\\ h_{\mathsf{CI}}\leftarrow\mathcal{H}_{\mathsf{CI}}(\lambda)\\ \mathsf{cm}:=\mathsf{A}_{\mathbb{x}}((\mathsf{pp},\mathsf{td}),h_{\mathsf{CI}})\\ \rho:=h_{\mathsf{CI}}(\mathsf{cm})\end{array}\right]\\
&=\Pr\left[(\mathsf{cm},h_{\mathsf{CI}}(\mathsf{cm}))\in R_{\mathbb{x},\mathsf{td}}\;\middle|\;\begin{array}{l}(\mathsf{pp},\mathsf{td})\leftarrow\mathsf{A}_{\mathbb{x}}\\ h_{\mathsf{CI}}\leftarrow\mathcal{H}_{\mathsf{CI}}(\lambda)\\ \mathsf{cm}:=\mathsf{A}_{\mathbb{x}}((\mathsf{pp},\mathsf{td}),h_{\mathsf{CI}})\end{array}\right]\\
&\leq\epsilon_{\mathsf{CI}}(\lambda,\epsilon_{\mathsf{s}},O(t_{\mathsf{SNARG}}))\,.
\end{aligned}$$

□

6.2.2 Security from FVC

Claim 6.4. *Let z_{VC} be the function hiding error of* VC*, the following holds:*

$$\Pr_{\mathsf{Exp}}\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\ \wedge\ \mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma)=z\\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=0\end{array}\right]\leq\epsilon_{\mathsf{c}}+\epsilon_{\mathsf{c}}\cdot 2^{\mathsf{r}_{\mathsf{F}}}\cdot z_{\mathsf{VC}}(t_{\mathsf{VC}})\,,$$

where $t_{\mathsf{VC}}=O(t_{\mathsf{SNARG}}+2^{\mathsf{r}_{\mathsf{F}}}\cdot\mathsf{s}_{\mathsf{PCP}}+\mathsf{t}_{\mathsf{D}})$.

Proof. According to PCP shadow consistency Definition 4.1, the following holds:

$$\begin{aligned}
&\Pr_{\mathsf{Exp}}\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\ \wedge\ \mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma)=z\\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=0\end{array}\right]\\
&=\Pr\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\ \wedge\ \mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma)=z\\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=0\end{array}\;\middle|\;\begin{array}{l}\gamma\leftarrow\{0,1\}^{\mathsf{r}_{\mathsf{F}}}\\ (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^{\lambda},\ell,\mathsf{F}(\mathbb{x},\cdot;\gamma))\\ h_{\mathsf{CI}}\leftarrow\mathcal{H}_{\mathsf{CI}}(\lambda)\\ (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}):=\widetilde{\mathcal{P}}(\mathsf{pp},h_{\mathsf{CI}})\\ \rho:=h_{\mathsf{CI}}(\mathsf{cm})\\ z:=\mathsf{VC.Extract}(\mathsf{td},\mathsf{cm})\end{array}\right]\\
&\leq\Pr\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\exists\,\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\ \wedge\ \mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma)=z\\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=0\\ \wedge\,\mathsf{D}(\mathbb{x},z',\rho)=0\end{array}\;\middle|\;\begin{array}{l}\gamma\leftarrow\{0,1\}^{\mathsf{r}_{\mathsf{F}}}\\ (\mathsf{pp},\mathsf{td})\leftarrow\mathsf{VC.Gen}(1^{\lambda},\ell,\mathsf{F}(\mathbb{x},\cdot;\gamma))\\ h_{\mathsf{CI}}\leftarrow\mathcal{H}_{\mathsf{CI}}(\lambda)\\ (\mathsf{cm},\mathsf{Q},\mathsf{a},\mathsf{pf}):=\widetilde{\mathcal{P}}(\mathsf{pp},h_{\mathsf{CI}})\\ \rho:=h_{\mathsf{CI}}(\mathsf{cm})\\ z:=\mathsf{VC.Extract}(\mathsf{td},\mathsf{cm})\\ \widetilde{\Pi}':=\sigma^{\ell}\\ \widetilde{\Pi}'[\mathsf{Q}]:=\mathsf{a}\\ z':=\mathsf{F}(\mathbb{x},\widetilde{\Pi}';\gamma)\end{array}\right].
\end{aligned}$$

We define the last experiment above as Exp'.

First note that

$$\Pr_{\mathsf{Exp}'}\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\exists\widetilde{\Pi}:\widetilde{\Pi}[\mathsf{Q}]=\mathsf{a}\ \wedge\ \mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma)=z\\ \wedge\,\mathsf{D}(\mathbb{x},z,\rho)=0\\ \wedge\,\mathsf{D}(\mathbb{x},z',\rho)=0\end{array}\right]\leq\Pr_{\mathsf{Exp}'}\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\,\mathsf{D}(\mathbb{x},z',\rho)=0\end{array}\right].$$

For every $\mathbb{x}$, ρ, and $\widetilde{\Pi}$, we define the set

$$S_{\mathbb{x},\rho,\widetilde{\Pi}}\coloneqq\{\gamma\in\{0,1\}^{\mathsf{r}_\mathsf{F}}:\mathsf{D}(\mathbb{x},\mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma),\rho)=0\}\,.$$

By correctness of PCP shadow soundness (Definition 4.1), we know that for every $\mathbb{x}\notin L(R)$, $\widetilde{\Pi}$ and every ρ such that $\mathbf{V}^{\widetilde{\Pi}}(\mathbb{x};\rho)=1$,

$$\Pr\left[\mathsf{D}(\mathbb{x},z,\rho)=0\;\middle|\;\begin{array}{l}\gamma\leftarrow\{0,1\}^{\mathsf{r}_\mathsf{T}}\\ z\coloneqq\mathsf{F}(\mathbb{x},\widetilde{\Pi};\gamma)\end{array}\right]\leq\epsilon_\mathsf{c}(\mathbb{x})\,.$$

Therefore,

$$\left|S_{\mathbb{x},\rho,\widetilde{\Pi}}\right|\leq\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}\,.$$

We can deduce that,

$$\Pr_{\mathsf{Exp}'}\left[\begin{array}{l}\mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x},\rho)=1\\ \wedge\,\mathsf{D}(\mathbb{x},z',\rho)=0\end{array}\right]\leq\Pr_{\mathsf{Exp}'}\left[\begin{array}{l}\mathbf{V}^{\widetilde{\Pi}'}(\mathbb{x};\rho)=1\\ \wedge\,\gamma\in S_{\mathbb{x},\rho,\widetilde{\Pi}'}\end{array}\right]\leq\Pr_{\mathsf{Exp}'}\left[\begin{array}{l}\left|S_{\mathbb{x},\rho,\widetilde{\Pi}'}\right|\leq\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}\\ \wedge\,\gamma\in S_{\mathbb{x},\rho,\widetilde{\Pi}'}\end{array}\right].$$

By definition of conditional probability, we can deduce that

$$\begin{aligned}&\Pr_{\mathsf{Exp}'}\left[\begin{array}{l}\left|S_{\mathbb{x},\rho,\widetilde{\Pi}'}\right|\leq\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}\\ \wedge\,\gamma\in S_{\mathbb{x},\rho,\widetilde{\Pi}'}\end{array}\right]\\ &=\Pr_{\substack{\mathsf{Exp}'\\ \gamma^\star\leftarrow S_{\mathbb{x},\rho,\widetilde{\Pi}'}}}\left[\begin{array}{l}\gamma^\star=\gamma\\ \wedge\,\left|S_{\mathbb{x},\rho,\widetilde{\Pi}'}\right|\leq\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}\\ \wedge\,\gamma\in S_{\mathbb{x},\rho,\widetilde{\Pi}'}\end{array}\right]\left(\Pr_{\substack{\mathsf{Exp}'\\ \gamma^\star\leftarrow S_{\mathbb{x},\boldsymbol{\rho},\widetilde{\Pi}'}}}\left[\begin{array}{l}\gamma^\star=\gamma\\ \text{conditioned on}\\ \left|S_{\mathbb{x},\rho,\widetilde{\Pi}'}\right|\leq\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}\\ \wedge\,\gamma\in S_{\mathbb{x},\rho,\widetilde{\Pi}'}\end{array}\right]\right)^{-1}\\ &\leq\Pr_{\substack{\mathsf{Exp}'\\ \gamma^\star\leftarrow S_{\mathbb{x},\rho,\widetilde{\Pi}'}}}[\gamma^\star=\gamma]\left(\Pr_{\substack{\mathsf{Exp}'\\ \gamma^\star\leftarrow S_{\mathbb{x},\rho,\widetilde{\Pi}'}}}\left[\begin{array}{l}\gamma^\star=\gamma\\ \text{conditioned on}\\ \left|S_{\mathbb{x},\rho,\widetilde{\Pi}'}\right|\leq\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}\\ \wedge\,\gamma\in S_{\mathbb{x},\rho,\widetilde{\Pi}'}\end{array}\right]\right)^{-1}\\ &\leq\Pr_{\substack{\mathsf{Exp}'\\ \gamma^\star\leftarrow S_{\mathbb{x},\rho,\widetilde{\Pi}'}}}[\gamma^\star=\gamma]\cdot\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}\,,\end{aligned}$$

where the last inequality follows because

$$\Pr_{\substack{\mathsf{Exp}'\\ \gamma^\star\leftarrow S_{\mathbb{x},\boldsymbol{\rho},\widetilde{\Pi}'}}}\left[\begin{array}{l}\gamma^\star=\gamma\\ \text{conditioned on}\\ \left|S_{\mathbb{x},\boldsymbol{\rho},\widetilde{\Pi}'}\right|\leq\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}\\ \wedge\,\gamma\in S_{\mathbb{x},\rho,\widetilde{\Pi}'}\end{array}\right]\geq\frac{1}{\epsilon_\mathsf{c}(\mathbb{x})\cdot 2^{\mathsf{r}_\mathsf{F}}}\,.$$

We proceed to bound the remaining probability above.

Consider the following function VC adversary A_{VC}:

$A_{\mathsf{VC}}(\mathsf{pp})$:
1. Run $(\mathsf{cm}, \mathsf{Q}, \mathsf{a}, \mathsf{pf}) \coloneqq \widetilde{\mathcal{P}}(\mathsf{pp}, h_{\mathsf{CI}})$.
2. Initialize $\widetilde{\Pi}' \coloneqq (\sigma)^{\ell}$, where σ is an arbitrary element in Σ.
3. Set $\widetilde{\Pi}'[\mathsf{Q}] \coloneqq \mathsf{a}$.
4. Sample uniformly at random $\gamma^{\star} \leftarrow S_{\mathbb{x},\rho,\widetilde{\Pi}'}$.
5. Output $\gamma^{\star}$.

Notice that Step 4 in A_{VC} can be implemented as follows:

1. Initialize an empty set $S \coloneqq \emptyset$.
2. For every $\gamma \in \{0,1\}^{\mathsf{r_T}}$:
   - Compute $z \coloneqq \mathsf{F}(\mathbb{x}, \widetilde{\Pi}'; \gamma)$.
   - If $\mathsf{D}(\mathbb{x}, z, \rho) = 0$ then add γ to S.
3. Sample uniformly at random $\gamma^{\star} \leftarrow S$.
4. Output $\gamma^{\star}$.

Hence, the running time of A_{VC} is $t_{\mathsf{VC}} = O(t_{\mathsf{SNARG}} + 2^{\mathsf{r_F}} \cdot \mathsf{s_{PCP}} + \mathsf{t_D})$.

From Lemma 5.4 and construction of A_{VC},

$$\Pr_{\substack{\mathsf{Exp}' \\ \gamma^{\star} \leftarrow S_{\mathbb{x},\rho,\widetilde{\Pi}'}}} [\gamma^{\star} = \gamma] = \Pr_{\substack{\mathsf{Exp}' \\ \gamma^{\star} \leftarrow A_{\mathsf{VC}}(\mathsf{pp_{VC}})}} [\gamma^{\star} = \gamma] \leq \frac{1}{2^{\mathsf{r_F}}} + z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}}) \,.$$

Hence,

$$\Pr_{\mathsf{Exp}'} \left[\begin{array}{l} \mathbf{V}^{[\mathsf{Q},\mathsf{a}]}(\mathbb{x}, \rho) = 1 \\ \wedge \exists \widetilde{\Pi} : \widetilde{\Pi}[\mathsf{Q}] = \mathsf{a} \ \wedge \ \mathsf{F}(\mathbb{x}, \widetilde{\Pi}; \gamma) = z \\ \wedge \mathsf{D}(\mathbb{x}, z, \rho) = 0 \\ \wedge \mathsf{D}(\mathbb{x}, z', \rho) = 0 \end{array} \right] \leq \left(\frac{1}{2^{\mathsf{r_F}}} + z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}}) \right) \cdot \epsilon_{\mathsf{c}}(\mathbb{x}) \cdot 2^{\mathsf{r_F}}$$

$$= \epsilon_{\mathsf{c}}(\mathbb{x}) + \epsilon_{\mathsf{c}}(\mathbb{x}) \cdot 2^{\mathsf{r_F}} \cdot z_{\mathsf{VC}}(\lambda, t_{\mathsf{VC}}) \,.$$

□

Acknowledgments

We are grateful to Ignacio Manzur and Zhengzhong Jin for their meticulous feedback on an earlier draft of this paper. We additionally thank Zvika Brakerski and Vinod Vaikuntanathan for insightful discussions on fully homomorphic encryption, and Karthik C.S. and Dor Minzer for valuable discussions regarding the notion of shadow PCPs. We thank Gal Arnon and Ariel Gabizon for invaluable feedback. We thank David Wu for pointing out missing citations regarding function vector commitments.

Ziyi Guan is partially supported by the Ethereum Foundation. Eylon Yogev is supported by the Israel Science Foundation (Grant No. 2302/22) and the European Research Union (ERC, CRYPTOPROOF, 101164375). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

[ACFY24] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. "STIR: Reed–Solomon Proximity Testing with Fewer Queries". In: *Proceedings of the 44th Annual International Cryptology Conference*. CRYPTO '24. 2024, pp. 380–413.

[ACFY25] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. "WHIR: Reed-Solomon Proximity Testing with Super-Fast Verification". In: *Proceedings of the 44th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '25. 2025, pp. 214–243.

[ACL+23] Martin R. Albrecht, Valerio Cini, Russell W. F. Lai, Giulio Malavolta, and Sri AravindaKrishnan Thyagarajan. "Lattice-Based Succinct Arguments from Vanishing Polynomials - (Extended Abstract)". In: *Proceedings of the 43rd Annual International Cryptology Conference*. CRYPTO '23. 2023, pp. 72–105.

[ACY23] Gal Arnon, Alessandro Chiesa, and Eylon Yogev. "IOPs with Inverse Polynomial Soundness Error". In: *Proceedings of the 64th Annual IEEE Symposium on Foundations of Computer Science*. FOCS '23. 2023, pp. 752–761.

[AFLN24] Martin R. Albrecht, Giacomo Fenzi, Oleksandra Lapiha, and Ngoc Khanh Nguyen. "SLAP: Succinct Lattice-Based Polynomial Commitments from Standard Assumptions". In: *Proceedings of the 43rd Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '24. 2024, pp. 90–119.

[AGL+23] Arasu Arun, Chaya Ganesh, Satya V. Lokam, Tushar Mopuri, and Sriram Sridhar. "Dew: A Transparent Constant-Sized Polynomial Commitment Scheme". In: *Proceedings of the 26th International Conference on Practice and Theory in Public Key Cryptography*. PKC '23. 2023, pp. 542–571.

[AHIV17] Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. "Ligero: Lightweight Sublinear Arguments Without a Trusted Setup". In: *Proceedings of the 24th ACM Conference on Computer and Communications Security*. CCS '17. 2017, pp. 2087–2104.

[AY25] Gal Arnon and Eylon Yogev. "Towards a White-Box Secure Fiat-Shamir Transformation". In: *Proceedings of the 45th Annual International Cryptology Conference*. CRYPTO '25. 2025, pp. 27–56.

[Bar01] Boaz Barak. "How to Go Beyond the Black-Box Simulation Barrier". In: *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*. FOCS '11. 2001, pp. 106–115.

[BBB+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. "Bulletproofs: Short Proofs for Confidential Transactions and More". In: *Proceedings of the 39th IEEE Symposium on Security and Privacy*. Oakland '18. 2018, pp. 315–334.

[BBH+19] James Bartusek, Liron Bronfman, Justin Holmgren, Fermi Ma, and Ron D. Rothblum. "On the (In)security of Kilian-Based SNARGs". In: *Proceedings of the 17th Theory of Cryptography Conference*. TCC '19. 2019, pp. 522–551.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. "Fast Reed–Solomon Interactive Oracle Proofs of Proximity". In: *Proceedings of the 45th International Colloquium on Automata, Languages and Programming*. ICALP '18. 2018, 14:1–14:17.

[BBK+23] Zvika Brakerski, Maya Farber Brodsky, Yael Tauman Kalai, Alex Lombardi, and Omer Paneth. "SNARGs for Monotone Policy Batch NP". In: *Proceedings of the 43rd Annual International Cryptology Conference*. CRYPTO '23. 2023, 252–283.

[BC25] Dan Boneh and Binyi Chen. "LatticeFold+: Faster, Simpler, Shorter Lattice-Based Folding for Succinct Proof Systems". In: *Proceedings of the 45th Annual International Cryptology Conference*. CRYPTO '25. 2025, pp. 327–361.

[BCC+14] Nir Bitansky, Ran Canetti, Alessandro Chiesa, Shafi Goldwasser, Huijia Lin, Aviad Rubinstein, and Eran Tromer. *The Hunting of the SNARK*. ePrint 2014/580. 2014.

[BCCT12] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. "From Extractable Collision Resistance to Succinct Non-Interactive Arguments of Knowledge, and Back Again". In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. ITCS '12. 2012, pp. 326–349.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. "Recursive composition and bootstrapping for SNARKS and proof-carrying data". In: *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*. STOC '13. 2013, pp. 111–120.

[BCFL23] David Balbás, Dario Catalano, Dario Fiore, and Russell W. F. Lai. "Chainable Functional Commitments for Unbounded-Depth Circuits". In: *Proceedings of the 21st Theory of Cryptography Conference*. TCC '23. 2023, pp. 363–393.

[BCI+13] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. "Succinct Non-Interactive Arguments via Linear Interactive Proofs". In: *Proceedings of the 10th Theory of Cryptography Conference*. TCC '13. 2013, pp. 315–333.

[BCP13] Nir Bitansky, Ran Canetti, and Omer Paneth. *How To Construct Extractable One-Way Functions Against Uniform Adversaries*. Cryptology ePrint Archive, Report 2013/468. 2013.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. "Interactive Oracle Proofs". In: *Proceedings of the 14th Theory of Cryptography Conference*. TCC '16-B. 2016, pp. 31–60.

[BDGM19] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. "Leveraging Linear Decryption: Rate-1 Fully-Homomorphic Encryption and Time-Lock Puzzles". In: *Proceedings of the 17th Theory of Cryptography Conference*. TCC '19. 2019, pp. 407–437.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) fully homomorphic encryption without bootstrapping". In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. ITCS '12. 2012, 309–325.

[BISW17] Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu. "Lattice-Based SNARGs and Their Application to More Efficient Obfuscation". In: *Proceedings of the 36th Annual International Conference on Theory and Applications of Cryptographic Techniques*. EUROCRYPT '17. 2017, pp. 247–277.

[BISW18] Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu. "Quasi-Optimal SNARGs via Linear Multi-Prover Interactive Proofs". In: *Proceedings of the 37th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '18. 2018, pp. 222–255.

[BKM20] Zvika Brakerski, Venkata Koppula, and Tamer Mour. "NIZK from LPN and Trapdoor Hash via Correlation Intractability for Approximable Relations". In: *Proceedings of the 40th Annual International Cryptology Conference*. CRYPTO '20. 2020, pp. 738–767.

[BMVY25] Mitali Bafna, Dor Minzer, Nikhil Vyas, and Zhiwei Yun. "Quasi-Linear Size PCPs with Small Soundness from HDX". In: *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*. STOC '25. 2025, pp. 45–53.

[Bra12] Zvika Brakerski. "Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP". In: *Proceedings of the 32nd Annual International Cryptology Conference*. CRYPTO '12. 2012, pp. 868–886.

[BS05] Eli Ben-Sasson and Madhu Sudan. "Simple PCPs with poly-log rate and query complexity". In: *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*. STOC '05. 2005, pp. 266–275.

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. "Fiat-Shamir: from practice to theory". In: *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*. STOC '19. 2019, pp. 1082–1090.

[CCR16] Ran Canetti, Yilei Chen, and Leonid Reyzin. "On the Correlation Intractability of Obfuscated Pseudorandom Functions". In: *Proceedings of the 13th Theory of Cryptography Conference*. TCC '16. 2016, pp. 389–415.

[CCRR18] Ran Canetti, Yilei Chen, Leonid Reyzin, and Ron D. Rothblum. "Fiat-Shamir and Correlation Intractability from Strong KDM-Secure Encryption". In: *Proceedings of the 37th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '18. 2018, pp. 91–122.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. "The random oracle methodology, revisited". In: *Journal of the ACM* 51.4 (2004), pp. 557–594.

[CGJ+23] Arka Rai Choudhuri, Sanjam Garg, Abhishek Jain, Zhengzhong Jin, and Jiaheng Zhang. "Correlation Intractability and SNARGs from Sub-exponential DDH". In: *Proceedings of the 43rd Annual International Cryptology Conference*. CRYPTO '23. 2023, 635–668.

[CGKS23] Matteo Campanelli, Chaya Ganesh, Hamidreza Khoshakhlagh, and Janno Siim. "Impossibilities in Succinct Arguments: Black-Box Extraction and More". In: *Proceedings of the 14th International Conference on Cryptology in Africa*. AFRICACRYPT' 23. 2023, pp. 465–489.

[CJJ21] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. "Non-interactive Batch Arguments for NP from Standard Assumptions". In: *Proceedings of the 41st Annual International Cryptology Conference*. CRYPTO '21. 2021, pp. 394–423.

[CJJ22] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. "SNARGs for $\mathcal{P}$ from LWE". In: *Proceedings of the 62nd Annual IEEE Symposium on Foundations of Computer Science*. FOCS '22. 2022, pp. 68–79.

[CLM23] Valerio Cini, Russell W. F. Lai, and Giulio Malavolta. "Lattice-Based Succinct Arguments from Vanishing Polynomials - (Extended Abstract)". In: *Proceedings of the 43rd Annual International Cryptology Conference*. CRYPTO '23. 2023, pp. 72–105.

[CY21a] Alessandro Chiesa and Eylon Yogev. "Subquadratic SNARGs in the Random Oracle Model". In: *Proceedings of the 41st Annual International Cryptology Conference*. CRYPTO '21. 2021, pp. 711–741.

[CY21b] Alessandro Chiesa and Eylon Yogev. "Tight Security Bounds for Micali's SNARGs". In: *Proceedings of the 19th Theory of Cryptography Conference*. TCC '21. 2021, pp. 401–434.

[CY24] Alessandro Chiesa and Eylon Yogev. *Building Cryptographic Proofs from Hash Functions*. 2024. URL: `https://github.com/hash-based-snargs-book`.

[DFH12] Ivan Damgård, Sebastian Faust, and Carmit Hazay. "Secure Two-Party Computation with Low Communication". In: *Proceedings of the 9th Theory of Cryptography Conference*. TCC '12. 2012, pp. 54–74.

[DGKV22] Lalita Devadas, Rishab Goyal, Yael Kalai, and Vinod Vaikuntanathan. "Rate-1 Non-Interactive Arguments for Batch-NP and Applications". In: *Proceedings of the 63rd Annual IEEE Symposium on Foundations of Computer Science*. FOCS '22. 2022, pp. 1057–1068.

[DHK+26] Lalita Devadas, Samuel B. Hopkins, Yael Tauman Kalai, Pravesh K. Kothari, Alex Lombardi, and Surya Mathialagan. *SNARGs for NP and Non-Signaling PCPs, Revisited*. Cryptology ePrint Archive, Paper 2026/006. 2026. URL: `https://eprint.iacr.org/2026/006`.

[Din07] Irit Dinur. "The PCP theorem by gap amplification". In: *Journal of the ACM* 54.3 (2007), p. 12.

[DWW24] Lalita Devadas, Brent Waters, and David J. Wu. "Batching Adaptively-Sound SNARGs for NP". In: *Proceedings of the 22nd Theory of Cryptography Conference*. TCC '24. 2024, 339–370.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. "Quadratic Span Programs and Succinct NIZKs without PCPs". In: *Proceedings of the 32nd Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '13. 2013, pp. 626–645.

[GK03] Shafi Goldwasser and Yael Tauman Kalai. "On the (In)security of the Fiat-Shamir Paradigm". In: *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*. FOCS '03. 2003, pp. 102–113.

[Gro10] Jens Groth. "Short Pairing-Based Non-interactive Zero-Knowledge Arguments". In: *Proceedings of the 16th International Conference on the Theory and Application of Cryptology and Information Security*. Ed. by Masayuki Abe. ASIACRYPT '10. 2010, pp. 321–340.

[Gro16] Jens Groth. "On the Size of Pairing-Based Non-interactive Arguments". In: *Proceedings of the 35th Annual International Conference on Theory and Applications of Cryptographic Techniques*. EUROCRYPT '16. 2016, pp. 305–326.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. "Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based". In: *Proceedings of the 33rd Annual International Cryptology Conference*. CRYPTO '13. 2013, pp. 75–92.

[GSWW22] Rachit Garg, Kristin Sheridan, Brent Waters, and David J. Wu. "Fully Succinct Batch Arguments for sfNP from Indistinguishability Obfuscation". In: *Proceedings of the 20th Theory of Cryptography Conference*. TCC '22. 2022, pp. 526–555.

[GW11] Craig Gentry and Daniel Wichs. "Separating Succinct Non-Interactive Arguments From All Falsifiable Assumptions". In: *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*. STOC '11. 2011, pp. 99–108.

[GZ21] Alonso González and Alexandros Zacharakis. "Fully-Succinct Publicly Verifiable Delegation from Constant-Size Assumptions". In: *Proceedings of the 19th Theory of Cryptography Conference*. TCC '21. 2021, pp. 529–557.

[HJKS22] James Hulett, Ruta Jawale, Dakshita Khurana, and Akshayaram Srinivasan. "SNARGs for P from Sub-exponential DDH and QR". In: *Proceedings of the 41st Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '22. 2022, pp. 520–549.

[HL18] Justin Holmgren and Alex Lombardi. "Cryptographic Hashing from Strong One-Way Functions (Or: One-Way Product Functions and Their Applications)". In: *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science*. FOCS '18. 2018, pp. 850–858.

[HLR21] Justin Holmgren, Alex Lombardi, and Ron D. Rothblum. "Fiat-Shamir via list-recoverable codes (or: parallel repetition of GMW is not zero-knowledge)". In: *Proceedings of the 53rd Annual ACM Symposium on Theory of Computing*. STOC '21. 2021, pp. 750–760.

[HW15] Pavel Hubácek and Daniel Wichs. "On the Communication Complexity of Secure Function Evaluation with Long Output". In: *Proceedings of the 6th Innovations in Theoretical Computer Science Conference*. ITCS '15. 2015, pp. 163–172.

[JJ21] Abhishek Jain and Zhengzhong Jin. "Non-interactive Zero Knowledge from Sub-exponential DDH". In: *Proceedings of the 40th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '21. 2021, pp. 3–32.

[JKLM25] Zhengzhong Jin, Yael Tauman Kalai, Alex Lombardi, and Surya Mathialagan. "Universal SNARGs for NP from Proofs of Correctness". In: *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*. STOC '25. 2025, 933–943.

[Kil92] Joe Kilian. "A note on efficient zero-knowledge proofs and arguments". In: *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*. STOC '92. 1992, pp. 723–732.

[KLV23] Yael Tauman Kalai, Alex Lombardi, and Vinod Vaikuntanathan. "SNARGs and PPAD Hardness from the Decisional Diffie-Hellman Assumption". In: *Proceedings of the 42nd Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '23. 2023, pp. 470–498.

[KLVW23] Yael Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. "Boosting Batch Arguments and RAM Delegation". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. STOC '23. 2023, pp. 1545–1552.

[KMN+22] Ilan Komargodski, Tal Moran, Moni Naor, Rafael Pass, Alon Rosen, and Eylon Yogev. "One-Way Functions and (Im)perfect Obfuscation". In: *SIAM J. Comput.* 51.6 (2022), pp. 1769–1795.

[KNY18] Ilan Komargodski, Moni Naor, and Eylon Yogev. "Collision Resistant Hashing for Paranoids: Dealing with Multiple Collisions". In: *Proceedings of the 37th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '18. 2018, pp. 162–194.

[KRR17] Yael Tauman Kalai, Guy N. Rothblum, and Ron D. Rothblum. "From Obfuscation to the Security of Fiat-Shamir for Proofs". In: *Proceedings of the 37th Annual International Cryptology Conference*. CRYPTO '17. 2017, pp. 224–251.

[KRS25] Dmitry Khovratovich, Ron D. Rothblum, and Lev Soukhanov. "How to Prove False Statements: Practical Attacks on Fiat-Shamir". In: *Proceedings of the 45th Annual International Cryptology Conference*. CRYPTO '25. 2025, pp. 3–26.

[KVZ21] Yael Tauman Kalai, Vinod Vaikuntanathan, and Rachel Yun Zhang. "Somewhere Statistical Soundness, Post-Quantum Security, and SNARGs". In: *Proceedings of the 19th Theory of Cryptography Conference*. TCC '21. 2021, pp. 330–368.

[Lip13] Helger Lipmaa. "Succinct Non-interactive Zero Knowledge Arguments from Span Programs and Linear Error-Correcting Codes". In: *Proceedings of the 19th International Conference on the Theory and Application of Cryptology and Information Security*. ASIACRYPT '13. 2013, pp. 41–60.

[Mic00] Silvio Micali. "Computationally Sound Proofs". In: *SIAM Journal on Computing* 30.4 (2000). Preliminary version appeared in FOCS '94., pp. 1253–1298.

[NWW24] Shafik Nassar, Brent Waters, and David J. Wu. "Monotone Policy BARGs from BARGs and Additively Homomorphic Encryption". In: *Proceedings of the 22nd Theory of Cryptography Conference*. TCC '24. 2024, pp. 399–430.

[NWW25] Shafik Nassar, Brent Waters, and David J. Wu. "Monotone-Policy BARGs and More from BARGs and Quadratic Residuosity". In: *Proceedings of the 28th International Conference on Practice and Theory in Public Key Cryptography*. PKC '25. 2025, pp. 283–313.

[OPWW15] Tatsuaki Okamoto, Krzysztof Pietrzak, Brent Waters, and Daniel Wichs. "New Realizations of Somewhere Statistically Binding Hashing and Positional Accumulators". In: *Proceedings of the 21st International Conference on the Theory and Application of Cryptology and Information Security*. ASIACRYPT '15. 2015, pp. 121–145.

[PP22] Omer Paneth and Rafael Pass. "Incrementally Verifiable Computation via Rate-1 Batch Arguments". In: *Proceedings of the 63rd Annual IEEE Symposium on Foundations of Computer Science*. FOCS '22. 2022, pp. 1045–1056.

[PS19] Chris Peikert and Sina Shiehian. "Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors". In: *Proceedings of the 39th Annual International Cryptology Conference*. CRYPTO '19. 2019, pp. 89–114.

[SW14] Amit Sahai and Brent Waters. "How to use indistinguishability obfuscation: deniable encryption, and more". In: *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*. STOC '14. 2014, pp. 475–484.

[Wee25a] Hoeteck Wee. "Almost Optimal KP and CP-ABE for Circuits from Succinct LWE". In: *Proceedings of the 44th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '25. 2025, pp. 34–62.

[Wee25b] Hoeteck Wee. "Functional Commitments and SNARGs for P/poly from SIS". In: *Proceedings of the 45th Annual International Cryptology Conference*. CRYPTO '25. 2025, 617–640.

[WW15] Hoeteck Wee and David J. Wu. "Lattice-Based Functional Commitments: Fast Verification and Cryptanalysis". In: *Proceedings of the 21st International Conference on the Theory and Application of Cryptology and Information Security*. ASIACRYPT '15. 2015, pp. 121–145.

[WW22] Brent Waters and David J. Wu. "Batch Arguments for NP and More from Standard Bilinear Group Assumptions". In: *Proceedings of the 42nd Annual International Cryptology Conference*. CRYPTO '22. 2022, 433–463.

[WW23] Hoeteck Wee and David J. Wu. "Succinct Vector, Polynomial, and Functional Commitments from Lattices". In: *Proceedings of the 42nd Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '23. 2023, pp. 385–416.

[WW24a] Brent Waters and David J. Wu. "Boosting Batch Arguments and RAM Delegation". In: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. STOC '24. 2024, 387–398.

[WW24b] Hoeteck Wee and David J. Wu. "Succinct Functional Commitments for Circuits from k-sfLin". In: *Proceedings of the 43rd Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '24. 2024, pp. 280–310.

[WW25] Brent Waters and David J. Wu. "A Pure Indistinguishability Obfuscation Approach to Adaptively-Sound SNARGs for NP". In: *Proceedings of the 45th Annual International Cryptology Conference*. CRYPTO '21. 2025, pp. 394–423.

[WZ24] Brent Waters and Mark Zhandry. "Adaptive Security in SNARGs via iO and Lossy Functions". In: *Proceedings of the 44th Annual International Cryptology Conference*. CRYPTO '24. 2024, pp. 72–104.

[ZCF24] Hadas Zeilberger, Binyi Chen, and Ben Fisch. "BaseFold: Efficient Field-Agnostic Polynomial Commitment Schemes from Foldable Codes". In: *Proceedings of the 44th Annual International Cryptology Conference*. CRYPTO '24. 2024, pp. 138–169.