Fiat-Shamir for Bounded-Depth Adversaries

Liyan Chen[*] Yilei Chen[†] Zikuan Huang[‡] Nuozhou Sun[§] Tianqi Yang[¶]
Yiding Zhang[‖]

February 16, 2024

Abstract

We study how to construct hash functions that can securely instantiate the Fiat-Shamir transformation against *bounded-depth* adversaries. The motivation is twofold. First, given the recent fruitful line of research of constructing cryptographic primitives against bounded-depth adversaries under *worst-case complexity assumptions*, and the rich applications of Fiat-Shamir, instantiating Fiat-Shamir hash functions against bounded-depth adversaries under worst-case complexity assumptions might lead to further applications (such as SNARG for P, showing the cryptographic hardness of PPAD, etc.) against bounded-depth adversaries. Second, we wonder whether it is possible to overcome the impossibility results of constructing Fiat-Shamir for arguments [Goldwasser, Kalai, FOCS '03] in the setting where the depth of the adversary is bounded, given that the known impossibility results (against p.p.t. adversaries) are contrived.

Our main results give new insights for Fiat-Shamir against bounded-depth adversaries in both the positive and negative directions. On the positive side, for Fiat-Shamir for proofs with certain properties, we show that weak worst-case assumptions are enough for constructing explicit hash functions that give $\mathsf{AC}^0[2]$-soundness. In particular, we construct an $\mathsf{AC}^0[2]$-computable correlation-intractable hash family for constant-degree polynomials against $\mathsf{AC}^0[2]$ adversaries, assuming $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \widetilde{\mathsf{Sum}}_{n^{-c}} \circ \mathsf{AC}^0[2]$ for some $c > 0$. This is incomparable to all currently-known constructions, which are typically useful for larger classes and against stronger adversaries, but based on arguably stronger assumptions. Our construction is inspired by the Fiat-Shamir hash function by Peikert and Shiehian [CRYPTO '19] and the fully-homomorphic encryption scheme against bounded-depth adversaries by Wang and Pan [EUROCRYPT '22].

On the negative side, we show Fiat-Shamir for arguments is still impossible to achieve against bounded-depth adversaries. In particular,

- Assuming the existence of $\mathsf{AC}^0[2]$-computable CRHF against p.p.t. adversaries, for every poly-size hash function, there is a (p.p.t.-sound) interactive argument that is not $\mathsf{AC}^0[2]$-sound after applying Fiat-Shamir with this hash function.
- Assuming the existence of $\mathsf{AC}^0[2]$-computable CRHF against $\mathsf{AC}^0[2]$ adversaries, there is an $\mathsf{AC}^0[2]$-sound interactive argument such that for every hash function computable by $\mathsf{AC}^0[2]$ circuits the argument does not preserve $\mathsf{AC}^0[2]$-soundness when applying Fiat-Shamir with this hash function. This is a low-depth variant of Goldwasser and Kalai.

[*]Tsinghua University. Email: chen-ly21@mails.tsinghua.edu.cn.

[†]Tsinghua University, Shanghai Artificial Intelligence Laboratory, and Shanghai Qi Zhi Institute. Email: chenyilei@mail.tsinghua.edu.cn. Research supported by Tsinghua University startup funding.

[‡]Tsinghua University. Email: hzk21@mails.tsinghua.edu.cn.

[§]Tsinghua University. Email: snz21@mails.tsinghua.edu.cn.

[¶]Columbia University. Email: tianqi@cs.columbia.edu.

[‖]Boston University. Email: zyding@bu.edu.

Contents

1 Introduction

The Fiat-Shamir transform [FS86] is a generic method for converting public-coin interactive protocols into non-interactive ones while preserving the original protocol's functionality. First proposed as a practical method to compile identification schemes into digital signatures, the Fiat-Shamir transform was then realized to be an extremely general technique for minimizing interaction in any public-coin protocols. The basic idea of Fiat-Shamir is to replace the verifier's message in each round (which consists of random coin tosses) with a deterministic hash of the protocol transcript so far. As a typical example, consider a three-round public-coin interactive proof/argument system (P, V) for a language L. When proving a statement $x \in L$, the prover P first sends a message α, the verifier V responds with random coins β, and then P sends the last message γ. To convert the interactive protocol (P, V) into a non-interactive protocol (P^{FS}, V^{FS}), the Fiat-Shamir transform uses a hash function h (randomly chosen from a hash family $\mathcal{H}$) and removes the verifier's coin tosses β by setting $\beta = h(x\|\alpha)$[1]. Then P^{FS} can directly compute β and send $(\alpha, \beta = h(x\|\alpha), \gamma)$ in one shot without any interaction (actually β can be omitted since it is clear from x, α and the public hash function h). See Figure 1 below for a visual explanation.

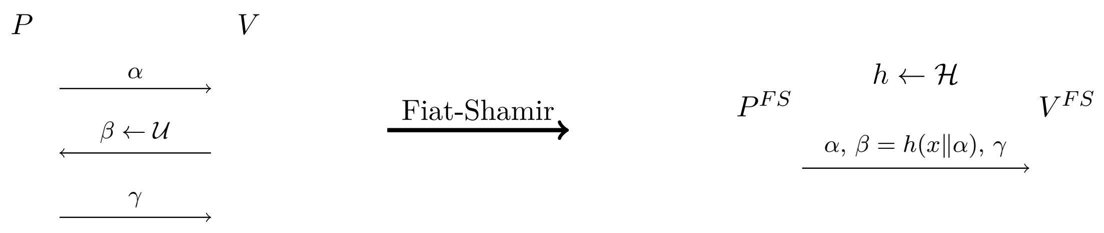

P
V
α
$\beta \leftarrow \mathcal{U}$
γ
Fiat-Shamir
$h \leftarrow \mathcal{H}$
P^{FS}
V^{FS}
$\alpha,\ \beta = h(x\|\alpha),\ \gamma$

Figure 1: Instantiating Fiat-Shamir on (P, V) for proving $x \in L$.

1.1 The soundness of Fiat-Shamir

The central problem of Fiat-Shamir is its soundness – justifying whether and when the Fiat-Shamir transform results in a sound non-interactive protocol has long been a focus of research in cryptography. It was shown by Pointcheval and Stern [PS96] that the soundness of Fiat-Shamir holds in the *random oracle model* (ROM), i.e., the soundness of any constant-round public-coin protocol can be preserved if we model the Fiat-Shamir hash function as a random oracle. However, it is more desirable to prove the soundness of Fiat-Shamir transformation in the *plain model*: the Fiat-Shamir hash should be instantiated by an explicit hash function in contrast to an idealized one, and the security in the random oracle model cannot provide any guarantee here.

In the plain model, the difference between *proofs* and *arguments* becomes crucial. By using proofs, we mean protocols that are sound against *any* cheating provers (can be computationally unbounded), while by using arguments, we mean protocols that are sound against only computationally bounded (probabilistic polynomial-time) cheating provers. We summarize the main positive results here.

- **Fiat-Shamir for proofs.** If a hash function is correlation intractable (CI) then it can securely instantiate Fiat-Shamir for proofs [CCRR18]. In the past few years there has been a long line of work on correlation intractable (CI) hash functions [CCR16; KRR17; CCRR18;

[1] "$\|$" means the concatenation of two strings. Here we use $h(x\|\alpha)$ but not $h(x, \alpha)$ to emphasize that h takes as input the entire transcript till now (consisting of x and α) instead of taking x and α separately as two inputs.

HL18; Can+19; PS19; BKM20; HLR21; JJ21; LV22; CGJJZ23] showing that we can securely instantiate Fiat-Shamir on interactive proofs of which the "bad relation" (which is roughly the messages (α, β) that may allow the prover to cheat) meets some special constraints. CI hash functions have already been used for applying Fiat-Shamir to some practical interactive proofs, e.g., the sumcheck protocol and the GKR protocol [JKKZ21], and also for constructing non-interactive cryptographic primitives from standard cryptographic assumptions, e.g., non-interactive zero-knowledge (NIZK) [PS19; BKM20; HLR21; JJ21] and succinct non-interactive arguments (SNARGs) [JKKZ21; CJJ21a; CJJ21b; HJKS22; KLV23; CGJJZ23].

- **Fiat-Shamir for arguments.** In the context of arguments, however, the soundness of Fiat-Shamir is still poorly understood. In some applications of Fiat-Shamir, we do know how to instantiate Fiat-Shamir securely on some special arguments using just CI hash functions (see, e.g., the SNARG constructions from [KVZ21; CJJ21a; CJJ21b; HJKS22; CGJJZ23]). However, the security mainly comes from the statistical property in their soundness such that these arguments can somehow be regarded as proofs – actually we are still using the framework for proofs. Outside the scope of "proof-like" arguments, there are only few positive results about some specific protocols, e.g., Fiat-Shamir for Schnorr's or Lyubashevsky's identification schemes [CLMQ21].

As we can see, positive results showing the soundness of Fiat-Shamir in the plain model concentrate almost all on proofs, while the soundness of Fiat-Shamir for arguments remains largely open (instantiating Fiat-Shamir soundly on proofs is intuitively easier since proofs have stronger security guarantees than arguments). Furthermore, there have been impossibility results or strong negative indications on the soundness of Fiat-Shamir for arguments in the plain model. Goldwasser and Kalai [GK03] constructed a set of three (contrived) arguments such that Fiat-Shamir using *any* explicit hash function must be insecure on at least one of them. In addition, Dwork, Naor, Reingold, and Stockmeyer [DNRS99] showed that (constant-round, public-coin) zero-knowledge arguments can also serve as counter-examples for Fiat-Shamir[2], and constructions of such protocols are known based on various cryptographic assumptions (see, e.g., [Bar01; BKP18; Kiy22]).

1.2 Fiat-Shamir and cryptography against bounded-depth adversaries

In this work, we study the soundness of Fiat-Shamir in a new setting where the adversaries's *parallel time* is further limited. In contrast to the standard security notion in cryptography that the adversaries run in probabilistic polynomial time (p.p.t.) with no further limitation, we consider a relaxed version of security that the adversaries are from *bounded-depth* (and thus bounded parallel time) circuit classes like $\mathsf{AC}^0[2]$ and TC^0. If we can get positive results for Fiat-Shamir against bounded-depth adversaries, it will be very exciting : given the rich applications of (or inspired by) Fiat-Shamir in recent years (such as SNARGs for P, showing the cryptographic hardness of PPAD [Cho+19], etc.), instantiating Fiat-Shamir hash functions against bounded-depth adversaries might lead to further applications against bounded-depth adversaries. Furthermore, given the strong negative indications and the extreme lack of positive results on Fiat-Shamir for arguments against p.p.t adversaries, we would like to see if it is possible to bypass the known impossibility results in the bounded-depth setting.

[2]The intuition of [DNRS99] is: we can construct a cheating verifier that sends messages according to the Fiat-Shamir hash function, and the zero-knowledge property guarantees that a simulator can efficiently simulate the view of such a cheating verifier. If the protocol is sound after Fiat-Shamir, the simulator can then be used to decide the language. Therefore, Fiat-Shamir can only be sound when the zero-knowledge protocol proves a language in BPP.

Recently there has been a fruitful sequence of work on constructing cryptographic primitives with security against only bounded-depth adversaries[3], including one-way functions, pseudorandom generators, collision-resistant hash functions [DVV16], one-way permutations [EWT21], fully homomorphic encryption schemes [CG18; WP22], and non-interactive zero-knowledge [WP22]. Compared to their analogs with the standard p.p.t. security, these primitives are indeed less secure but can rely only on worst-case complexity assumptions or even be unconditional (see, e.g., the construction of CRHF against AC^0 in [DVV16]).

Back to the context of Fiat-Shamir, our work focus on the *low-end* of Fiat-Shamir: the soundness is rather weak (but still non-trivial) against circuit classes as low as $\mathsf{AC}^0[2]$, while the underlying assumption is likewise weak. The choice of bounded-depth circuit classes is also motivated by the developments in circuit complexity, where strong lower bounds for bounded-depth classes are known. Therefore, it is very likely that we can translate these lower bounds into good hash constructions that are only useful for Fiat-Shamir against the same bounded-depth classes.

Also note that the lowest circuit class we care about is $\mathsf{AC}^0[2]$. We choose "$\mathsf{AC}^0[2]$" because our results (as shown in the next section) can indeed work for classes as low as $\mathsf{AC}^0[2]$, and $\mathsf{AC}^0[2]$ is arguably the smallest circuit class that makes Fiat-Shamir reasonable. In cryptography, almost all hash functions (and other cryptographic primitives) use some kind of linearity or algebra over finite fields, and constructions without the use of parity or modulo gates are rare. To the best of our knowledge, there are only two exceptions about classes lower than $\mathsf{AC}^0[2]$. The first one is the cryptography in AC^0 by Degwekar, Vaikuntanathan, and Vasudevan [DVV16], but their construction is still inherently using parity, except they use some clever tricks to bound the number of bits in the parity by $\mathsf{polylog}$. The other one is the cryptography in NC^0 by Applebaum, Ishai, and Kushilevitz [AIK06], but their construction is not robust enough, e.g., their PRG can only have additive stretch.

1.3 Our results

Our results include two independent parts that make progress on both the positive side and the negative side. On the positive side, we give new constructions of Fiat-Shamir hash functions for proofs in the bounded-depth setting. Inspired by the correlation intractable hash by Peikert and Shiehian [PS19] and the FHE with bounded-depth security by Wang and Pan [WP22], we construct a new family of correlation intractable hash functions against bounded-depth adversaries. It allows us to generally instantiate Fiat-Shamir, at least on a class of interactive proofs, to obtain non-interactive cryptographic primitives in the bounded-depth setting. On the negative side, unfortunately, we observe that our positive results on proofs are almost the best we can do in the bounded-depth setting. In particular, we show that the known impossibility results for Fiat-Shamir for arguments like Goldwasser and Kalai [GK03] can be extended to the bounded-depth setting.

1.3.1 Positive results

We first present the positive results for Fiat-Shamir for *proofs* against bounded-depth cheating provers. In particular, we show new constructions of correlation-intractable (CI) hash functions for relations representable by constant-degree polynomials that are secure against bounded-depth adversaries.

[3]This research topic is also referred to as "fine-grained cryptography" like in [DVV16]. We do not use the notion "fine-grained" because in complexity theory it usually refers to unrobust classes like $\mathsf{DTIME}[n^2]$. We use the name "cryptography against bounded-depth adversaries" to avoid any possible ambiguity.

Our construction is inspired by the CI hash function constructed by Peikert and Shiehian [PS19], which uses the FHE scheme of Gentry, Sahai, Waters [GSW13] (henceforth GSW-FHE) as a building block. Our main observation is that the FHE scheme against low-depth adversaries constructed by Wang and Pan [WP22] bares a striking similarity with GSW-FHE. The similarity allows us to replace GSW-FHE with WP-FHE to obtain a CI hash function against low-depth adversaries assuming worst-case complexity assumptions. Our main positive result is:

Theorem 1.1 (Informally stated, see Theorem 4.18). Assuming $\mathsf{NC}^1 \neq \oplus\mathsf{L}/\mathsf{poly}$, there exists a CI hash function family for constant-degree polynomials against NC^1 that is computable in $\mathsf{AC}^0[2]$. $\diamondsuit$

The circuit class NC^1 here is just a typical example and can actually be replaced with lower circuit classes. See Section 4.3 for more details. The worst-case complexity assumption is a standard assumption in the area of cryptography against bounded-depth adversaries: previous constructions of, e.g., one-way functions, pseudorandom generators, collision-resistant hash functions [DVV16] are all based on the same assumption.

1.3.2 Negative results

On the negative side, we get two (incomparable) impossibility results in the bounded-depth setting. The first one shows the non-existence of a *universal* Fiat-Shamir hash against bounded-depth adversaries; while the second one extends the work by Goldwasser and Kalai [GK03] showing the existence of a protocol that makes every Fiat-Shamir hash fail even in the bounded-depth setting.

$\mathcal{C}$-soundness and $\mathcal{C}$-arguments. For simplicity, we say a protocol has $\mathcal{C}$-soundness if it is sound against adversaries computable in the complexity class $\mathcal{C}$. Also, we use the notation $\mathcal{C}$-arguments for protocols with $\mathcal{C}$-soundness. Typical choices of $\mathcal{C}$ include bounded-depth circuit classes $\mathsf{AC}^0[2]$, TC^0, or NC^1. When just saying an argument, we refer to a P/poly-argument.

We first (informally) define what is a universal Fiat-Shamir hash in the bounded-depth setting (see definition 5.1 for the detailed definition):

Definition 1.2 (Universal Fiat-Shamir hash, informal). For circuit classes $\mathcal{D} \subseteq \mathcal{C}$, we say a hash function family $\mathcal{H}$ is a universal Fiat-Shamir hash from $\mathcal{C}$ to $\mathcal{D}$ if Fiat-Shamir using $\mathcal{H}$ always results in a sound non-interactive $\mathcal{D}$-argument starting from any (3-round public-coin) interactive $\mathcal{C}$-argument that matches the input and output lengths of $\mathcal{H}$ (i.e., the protocol's instance + first message length should be $\mathcal{H}$'s input length, and the protocol's second message length should be $\mathcal{H}$'s output length). $\diamondsuit$

Note that Definition 1.2 is already enough to capture a large class of hash functions, which is a fixed hash function family as long as the input and output lengths are determined. As an example, the CI hash function in [CCRR18] fits well into this format. However, there also exist hash functions outside the scope of Definition 1.2. For example, the CI hash from [PS19] does not fit into this definition[4] because its key length depends on some specific property of the interactive proofs we want to collapse, which cannot be fixed in advance given only the input and output

[4]Here we are only talking about the formats of these hash functions (e.g., how the key length is chosen) instead of their specific functionalities – of course, we do not have such a universal Fiat-Shamir hash with security against p.p.t. due to [GK03].

lengths (more precisely, its key length depends on how fast we can compute the "bad function" of the protocol). Based on this definition, we give our first negative result:

Theorem 1.3 (Informally stated, see Theorem 5.2). Assuming the existence of a CRHF family that is computable in $\mathcal{D}$ and collision-resistant against $\mathcal{C}$ with any polynomial shrinkage, there does not exist a universal Fiat-Shamir hash from $\mathcal{C}$ to $\mathcal{D}$ with any polynomial input and output lengths. $\diamondsuit$

Here we further justify our assumption of a low-complexity CRHF. One can examine that the CRHF construction $f_A(x) = Ax \bmod q$ based on the hardness of short integer solution (SIS) [Ajt96] is actually computable in TC^0 (and has polynomial shrinkage). Also, the discrete-log (DLOG)-based CRHF construction $f_{g,h}(x_1, x_2) = g^{x_1} \cdot h^{x_2}$ can also be computed by a TC^0 circuit using some standard pre-processing techniques. Therefore, low-complexity CRHFs are known from these well-studied cryptographic assumptions, which leads to the following corollary:

Corollary 1.4. Assuming SIS/DLOG, there does not exist a universal Fiat-Shamir hash that can soundly transform any interactive argument into a non-interactive TC^0-argument. $\diamondsuit$

Actually, similar results hold also for circuit classes lower than TC^0. We present a candidate $\mathsf{AC}^0[2]$-computable poly-shrinkage CRHF based on a variant of SIS, indicating the impossibility of a universal Fiat-Shamir hash achieving even only $\mathsf{AC}^0[2]$-soundness. See Appendix B for more discussions on the $\mathsf{AC}^0[2]$-computable CRHF.

Even more generally, we show that the the result of Goldwasser and Kalai [GK03] – albeit looks complicated and requires super-constant depth at a first glance – can be generalized to the bounded-depth setting with adversaries from circuit classes as low as $\mathsf{AC}^0[2]$.

Theorem 1.5 (Informally stated, see Theorem 6.1). For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, assuming the existence of a CRHF family with arbitrary polynomial shrinkage that is computable in $\mathcal{C}$ and collision-resistant against $\mathcal{C}$, there exists a 3-round public-coin $\mathcal{C}$-argument such that instantiating Fiat-Shamir using any hash function computable in $\mathcal{C}$ does not result in a sound non-interactive $\mathcal{C}$-argument. $\diamondsuit$

Compared to the previous result in Theorem 1.3 showing impossibility even when we require only some weaker soundness on the Fiat-Shamir-collapsed protocol, Theorem 1.5 shows only the impossibility of Fiat-Shamir hash functions preserving the same level of soundness due to some technical issues. However, instead of saying every FS hash must fail on some tailored protocol, Theorem 1.5 is stronger in the sense that it shows the existence of some fixed protocol that makes every Fiat-Shamir hash fail. The Fiat-Shamir hash is allowed to depend arbitrarily on the fixed interactive protocols (e.g., the third message's length $|\gamma|$ or the prover/verifier running time, which are not allowed in Definition 1.2 and Theorem 1.3).

Also note that the CRHF in our assumption can be instantiated (at least for $\mathsf{AC}^0[2] \subseteq \mathcal{C} \subseteq \mathsf{NC}^1$) by the CRHF from [DVV16] that is computable in $\mathsf{AC}^0[2]$ and secure against NC^1 based on the worst-case complexity assumption $\mathsf{NC}^1 \neq \oplus\mathsf{L/poly}$.

1.4 Related works

Aside from the results like Goldwasser and Kalai [GK03] showing that there is an interactive argument such that no explicit hash function can collapse the protocol while preserving soundness, there have also been negative results for some restricted classes of protocols, in particular, for protocols related to SNARGs like Kilian's protocol.

Kilian's protocol [Kil92] is a 4-message succinct argument for any NP language. The idea of collapsing Kilian's protocol by Fiat-Shamir comes from Micali's CS proofs [Mic00], which are now usually referred to as succinct non-interactive arguments (SNARGs). While Micali's construction applies Fiat-Shamir in the random oracle model, there have been negative results on Fiat-Shamir for Kilian's protocol in the plain model. Gentry and Wichs [GW11] showed a substantial barrier to constructing SNARGs in general (not limited to collapsing Kilian's protocol), and the problem of securely instantiating Fiat-Shamir on Kilian's protocol also suffers from this barrier. More recently, Bartusek, Bronfman, Holmgren, Ma, and Rothblum [BBHMR19] showed that Fiat-Shamir with any explicit hash function can never be secure on some contrived instantiation of Kilian's protocol.

Organization. The rest of this paper is organized as follows. We first give an overview of the intuition and techniques underlying our results in Section 2. In the subsequent sections, we present the details of our main results: on the positive side, we present our construction of CI hash functions in Section 4; on the negative side, we first show the impossibility of universal Fiat-Shamir hash functions in Section 5 (which corresponds to Theorem 1.3), and then present the stronger negative result of a universal counter-example for Fiat-Shamir in Section 6 (which corresponds to Theorem 1.5).

2 Technical Overview

In this section, we give an overview of the main ideas and techniques underlying our proofs.

2.1 Positive results

Our positive result is inspired by the CI hash function in [PS19], which crucially relies on the FHE scheme of Gentry, Sahai, Waters [GSW13] (henceforth GSW-FHE). Our main observation is that the FHE scheme against low-depth adversaries constructed by Wang and Pan [WP22] (henceforth WP-FHE) bares a striking similarity with GSW-FHE, therefore it can be used as in [PS19] to obtain a CI hash function against low-depth adversaries.

Let us start by explaining the intuition of [PS19]. They consider sparse relations defined by efficient functions i.e. $\{(x, f(x))\}$ for some efficiently computable function f. Designing a hash function $h(k, x)$ that is correlation intractable for a single function-defined relation is trivial: simply define $h(k, x) = f(x) + 1$. The construction [PS19] utilized the idea: for any f, a random hash key will look indistinguishable from a hash key that is specifically designed to be correlation intractable with respect to the relation defined by f, i.e. the construction achieves the notion of "somewhere correlation intractability". This idea can be implemented by using any fully homomorphic encryption scheme, i.e., the hash key is the homomorphic encryption of the circuit that computes a fixed function g_0 (hence for any f, it looks indistinguishable from the encryption of the circuit that computes f), and the hash function homomorphically computes $g_0(x) + 1$ using a universal circuit $\mathcal{U}(x, \cdot)$ s.t. $\mathcal{U}(x, f) = f(x)$. The result of homomorphic evaluation would be a ciphertext of $g_0(x)+1$, so it remains as the last step to decrypt this ciphertext. One idea, as shown in [Can+19], is to put the encryption of the FHE decryption function inside the public key.

$$h(k, x) = \mathsf{Eval}(\mathsf{pk}, \mathcal{U}(x, \cdot), \mathsf{ct}), \text{ where } k = (\mathsf{pk}, \mathsf{ct}), \mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, g_0) \approx \mathsf{Enc}(\mathsf{pk}, \mathsf{Dec}(\mathsf{sk}, f(\cdot)) \oplus 1).$$

Then for any function f, if k is sampled according to $(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{Dec}(\mathsf{sk}, f(\cdot)) \oplus 1))$, we have

$$f(x) = h(k, x) \implies f(x) = \mathsf{Enc}(\mathsf{pk}, \mathsf{Dec}(\mathsf{sk}, f(x)) \oplus 1) \implies \mathsf{Dec}(\mathsf{sk}, f(x)) = \mathsf{Dec}(\mathsf{sk}, f(x)) \oplus 1.$$

However, in this approach, in order to show the hash key k does not leak FHE secret key, we need to assume circular security. [PS19] bypasses this "decryption problem" based on an observation about the ciphertext of GSW-FHE. Namely, for any vector $(y_i)_{i\in[n]}$, if we have the ciphertext of each y_i, then we can compute $Ar + y$ where A is the FHE public key and r is derived from the randomness used to encrypt each y_i. Coarsely, if $y = f(x)$ as the result of homomorphic evaluation, then from $f(x) = Ar + f(x)$ we find a solution for $Ar = 0$. [PS19] manages to reduce breaking SIS assumption to breaking correlation intractability of their construction.

Our construction of CI against low-depth adversary can be explained in a single sentence: replace GSW-FHE in the construction [PS19] by WP-FHE. We claim that these two FHE schemes, though under different assumptions and are against different adversaries, are similar to each other in a precise sense: WP-FHE can be regarded as the mod-2, noise-free version of GSW-FHE. We provide the following table to sustain this intuition.

	GSW-FHE	WP-FHE
$(\mathsf{pk}, \mathsf{sk})$	$k^T A = e \approx 0$	$k^T A = 0$
Enc	$AR + mG$	$AR + mI$
Dec	$k^T \mathsf{ct} \cdot G^{-1}((0, \dots, 0, q/2)^T)$	$k^T \mathsf{ct} \cdot (0, \dots, 0, 1)^T$
$\mathsf{Eval}^{\mathrm{add}}$	$\mathsf{ct}_0 + \mathsf{ct}_1 - 2\mathsf{ct}_0 \cdot G^{-1}(\mathsf{ct}_1)$	$\mathsf{ct}_0 + \mathsf{ct}_1$
$\mathsf{Eval}^{\mathrm{mul}}$	$\mathsf{ct}_0 \cdot G^{-1}(\mathsf{ct}_1)$	$\mathsf{ct}_0 \cdot \mathsf{ct}_1$

Table 1: Comparison between GSW-FHE and WP-FHE

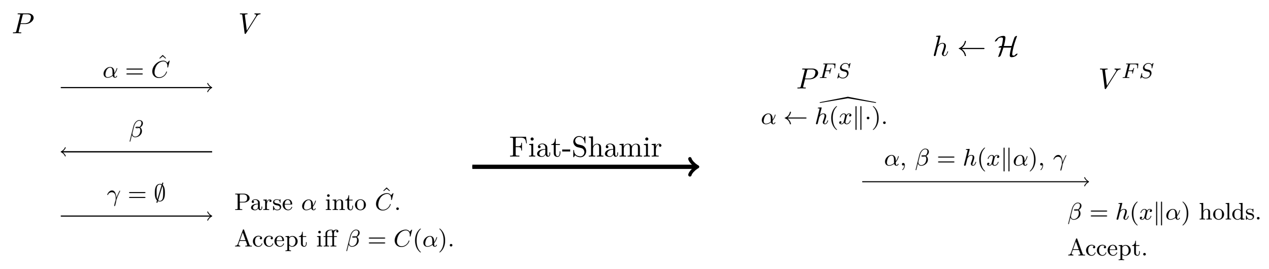
The replacement allows the hash function to be computable in $\mathsf{AC}^0[2]$. Also, we use a result in [EWT21] to show that the security of our construction follows from the worst-case complexity assumption.

2.2 Negative results

Our main impossibility result (Theorem 6.1) comes from adapting the impossibility result by [GK03] to the bounded-depth settings. Before stepping into the technical details of the modification, we first explain the intuition behind the impossibility results of Fiat-Shamir for arguments and show a much simpler (albeit weaker) impossibility result (Theorem 5.2) which directly follows the intuition.

There is a simple intuition behind all existing contrived impossibility results: the prover in an interactive protocol is never able to predict the verifier's choice of the next message, but Fiat-Shamir makes the verifier's messages "predictable". Consider the standard 3-round public-coin setting where the protocol transcript is (α, β, γ). While sending the first message α, the prover can never predict the next message β in advance – no matter what prediction it makes, the verifier's random choice of β coincides with the prediction only with negligible probability. However, things become totally different when we collapse the protocol by Fiat-Shamir: with the help of a randomly chosen hash function $h \leftarrow \mathcal{H}_{FS}$, the prover now directly generates $(\alpha, \beta = h(x\|\alpha), \gamma)$ in one shot without the verifier's random choice. In other words, the next "verifier's message" $\beta = h(x\|\alpha)$ is just a fixed value after generating α, and of course, the malicious prover can predict β.

Based on this observation, we can design an interactive protocol that simply asks the prover to predict the next message: in the first round, the prover tries to predict β by submitting the description of a function $\alpha = \hat{C}$ (we use C to denote the circuit of the function and $\hat{C}$ to denote the encoding of C), which means his prediction of β is $C(\alpha) = C(\hat{C})$. Then the verifier randomly chooses β and sends it to the prover, ignores the prover's third message, and finally accepts iff $\beta = C(\alpha)$ holds (which means the prediction is correct). This is an interactive proof for the empty language: the verifier always rejects except the randomly-chosen β happens to be the fixed value $C(\alpha)$, which holds with only negligible probability[5]. After applying Fiat-Shamir, unfortunately, the non-interactive protocol becomes totally broken. Given a Fiat-Shamir hash $h \leftarrow \mathcal{H}_{FS}$, the cheating prover can simply send the description of this hash function (i.e., send $\alpha = \widehat{h(x\|\cdot)}$ with the instance x hardcoded), and then the Fiat-Shamir transform indeed chooses β as $\beta = h(x\|\alpha)$, which makes the verifier accept.



P
V
$\alpha = \hat{C}$
β
$\gamma = \emptyset$
Parse α into $\hat{C}$.
Accept iff $\beta = C(\alpha)$.
Fiat-Shamir
$h \leftarrow \mathcal{H}$
P^{FS}
V^{FS}
$\alpha \leftarrow \widehat{h(x\|\cdot)}$.
$\alpha,\ \beta = h(x\|\alpha),\ \gamma$
$\beta = h(x\|\alpha)$ holds.
Accept.

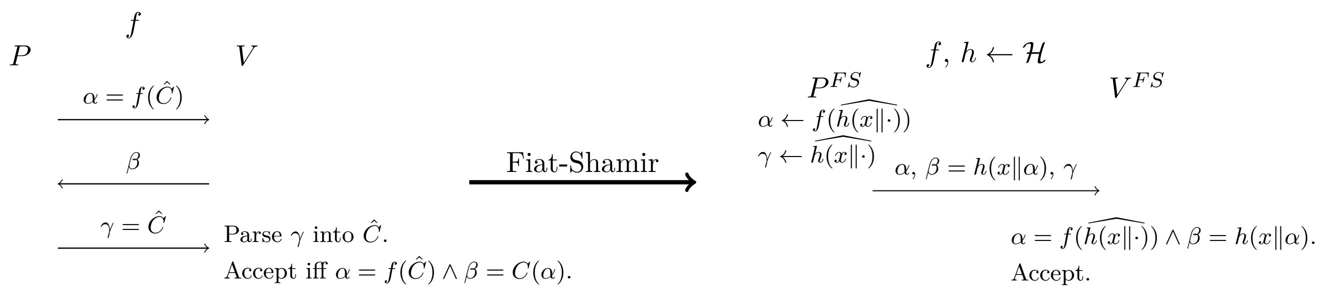
Figure 2: A protocol for the empty language that is not sound after Fiat-Shamir.

This counter-example can work in our bounded-depth setting (since what the cheating prover does is simply copying the Fiat-Shamir hash function). However, there is an implicit limitation: when saying $\alpha = \widehat{h(x\|\cdot)}$, $\beta = h(x\|\alpha)$, we implicitly require the Fiat-Shamir hash h to have a short enough description that can fit into its input. In the case of choosing $h \leftarrow \mathcal{H}_{FS}$, the description size can be roughly regarded as the length of the key k of the hash family $\mathcal{H}_{FS}$, and thus the hash function here must satisfy $|k| \leq |\alpha|$. As a result, this counter-example fails to capture a large class of Fiat-Shamir hash functions with the key length longer than the input length. This limitation somehow makes the counter-example worthless, since Fiat-Shamir hash functions usually have long keys: as shown in the work of Canetti, Goldreich, and Halevi [CGH04], a hash function cannot even be correlation intractable (i.e., preserving soundness for proofs) if its key length is shorter than the input length.

In order to bypass the limitation on the key length, we made some modifications based on the protocol above. Instead of sending the whole circuit description $\hat{C}$ in the first round, the prover only commits to such a description by sending a succinct commitment $\alpha = f(\hat{C})$ (f is the commitment function) – although the description may be long, its commitment can still be short enough to fit into the Fiat-Shamir hash's input. After that, the prover in the third round sends $\gamma = \hat{C}$ in clear to open the commitment, and the verifier decides to accept or reject by checking if $\alpha = f(\hat{C})$ (the opening is correct) and $\beta = C(\alpha)$ (the prediction is correct). See Figure 3 for an illustration of the modified protocol.

Note that this protocol is an interactive *argument* for the empty language, since a succinct

[5]Note that in an interactive protocol for the empty language, there is no *honest* prover since no instance can be proved (except with negligible probability). Or we can say the honest prover is just a machine that sends all-0 strings of some fixed lengths in each round. We clarify here that when describing the prover's behaviors, we actually mean what the prover is supposed to do (and the verifier will check it later). For example, by saying the prover sends $\alpha = \hat{C}$, we actually mean the cheating prover sends an arbitrarily α and the verifier will parse α into the description of a circuit C when deciding to accept/reject.



f
P
V
$\alpha = f(\hat{C})$
β
$\gamma = \hat{C}$
Parse γ into $\hat{C}$.
Accept iff $\alpha = f(\hat{C}) \wedge \beta = C(\alpha)$.
Fiat-Shamir
$f, h \leftarrow \mathcal{H}$
P^{FS}
V^{FS}
$\alpha \leftarrow f(\widehat{h(x\|\cdot)})$
$\gamma \leftarrow \widehat{h(x\|\cdot)}$
$\alpha,\ \beta = h(x\|\alpha),\ \gamma$
$\alpha = f(\widehat{h(x\|\cdot)}) \wedge \beta = h(x\|\alpha)$.
Accept.

Figure 3: The modified protocol for the empty language.

commitment inherently cannot have statistical binding. In order to fit into the bounded-depth setting, we instantiate the succinct commitment f with a CRHF that can be computed in low depth and has enough shrinkage, which leads to our first negative result as shown in Theorem 1.3. Note that the exact shrinkage of CRHF matters here because we cannot boost the shrinkage arbitrarily in the bounded-depth setting: the circuit depth grows as we stack a CRHF several times to build a Merkle tree. The detailed security proof of this protocol is deferred to Section 5.

To obtain a more general negative result, we also extend the result of [GK03] and similarly get a set of three protocols that Fiat-Shamir must fail on at least one of them. We first summarize the high-level idea of the construction in [GK03]. Among the three protocols, the first protocol is almost the same as the one in Figure 3, except that we also need to limit the length of γ in order to be more general. To obtain a shorter γ, the prover now does not send the opening $\hat{C}$ in clear and sends a "SNARK" instead to prove that he knows a valid opening. The use of such a strong primitive may seem ridiculous, but we bypass the problem of constructing a "SNARK" by applying a win-win argument: we construct a second protocol (e.g., Kilian's protocol [Kil92]) that results in a "SNARK" if Fiat-Shamir can be securely instantiated on it. Then for any explicit Fiat-Shamir hash function, if it can be soundly applied to the second protocol, we get a construction of "SNARK" and the first protocol becomes a valid counter-example; if not, the second protocol itself is a counter-example for Fiat-Shamir – anyway, we can get at least one protocol that is not sound after Fiat-Shamir. Note that the "SNARK" here is just an intuitive example of the condition we set in a win-win argument (actually it does not work – Kilian's protocol does not necessarily preserve proof of knowledge after Fiat-Shamir). The real construction is much more complicated with win-win arguments among three protocols, and we defer the detailed construction to Section 6. In the rest part of this section, we give an overview the main technical issues we need to resolve in adapting the [GK03] construction to the more restricted bounded-depth setting:

- **Merkle tree hash**: The construction of [GK03] relies on a Merkle tree hash to succinctly commit to messages that may have an arbitrary polynomial size. In our bounded-depth setting, however, the depth of the Merkle tree is inherently limited to a constant. To handle limitation on depth, we base our construction on the stronger assumption of a CRHF with arbitrary polynomial shrinkage – then a constant-depth Merkle tree is enough to handle any polynomial-size message.

- $\mathsf{AC}^0[2]$**-computable PCPs**: [GK03] also uses probabilistic checkable proofs (PCPs) to guarantee the efficiency of the verifier. Since the prover in our setting is only allowed to do the bounded-depth computation, we therefore need a bounded-depth computable PCP. We

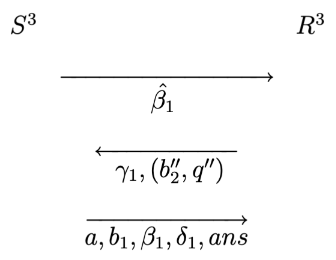


Figure 4: The picture of Protocol 3 from [GK03, Page 32]. Here q'' denotes the CRS of a Fiat-Shamir reduced protocol used in their Protocol 1. Since q'' is sent in the second message of the verifier R^3, q'' is supposed to be uniformly random. However, this implies (in a contrived manner) that the Fiat-Shamir hash function applied to Protocol 1 must be a public-coin hash function.

present the construction of an $\mathsf{AC}^0[2]$-computable PCP in Appendix C, which meets our requirements here.

- **Public-coin hash v.s. private-coin hash**: Note that the original construction in [GK03] works only for *public-coin* Fiat-Shamir hash functions, i.e., the hash key must be a uniformly generated random string instead of a structured one with the generator's randomness remain hidden. In order to achieve a win-win situation, the [GK03] construction needs to plug the non-interactive Fiat-Shamir-collapsed version of one interactive protocol into another interactive protocol, and the public-coin verifier is responsible for generating the common reference string (which contains the Fiat-Shamir hash function). See Figure 4 for an illustration. A later work by Goldwasser and Kalai [GK] fixed this issue by changing the way of "plugging in" non-interactive protocols. To obtain a more general impossibility result that works also for private-coin Fiat-Shamir hash functions, we use a similar modified method of plugging in protocols for our construction in the bounded-depth setting.

3 Preliminaries

3.1 Circuit

In the context of this paper, when we refer to circuits, we are specifically referring to Boolean circuits designed to accept a fixed number of input bits and produce a fixed number of output bits, with gates inside as computation units. For any given circuit denoted as C, we employ the notation $C(x)$ to signify the result produced by circuit C when it is given input x. Within the scope of a defined circuit class $\mathcal{C}$, we employ the notation $\hat{C}$ to represent the encoded representation of circuit C, and we denote the length of this encoding as $|\hat{C}|$. Here, we require the encoding satisfying that:

- Hard-coding any bits into the input of circuit C will not affect the length of the encoding.
- For all polynomials $S(n), d(n)$, there exists a universal circuit class $\mathcal{U} = \{U_n\}_{n\in\mathbb{N}}$ in $\mathcal{C}$, such that for any circuit families $\{C_n\}_{n\in\mathbb{N}}$ in $\mathcal{C}$ with encoding length $S(n)$ and depth $d(n)$, U_n can evaluate C_n .

It is worth noting that universal circuits exist in all commonly encountered circuit classes, which ensures the existence of required encoding. We sometimes refer to the length of encoding as circuit size, when we do not explicitly specify encoding of the circuit.

Circuit classes. We frequently use the following circuit classes in this paper (cf. [AB09] for a standard reference):

Definition 3.1 (NC^d). For every d, a language L is in NC^d if L can be decided by a family of circuits $\{C_n\}$ where C_n has $\mathsf{poly}(n)$ size and depth $O(\log^d n)$. $\diamondsuit$

Definition 3.2 (AC^d). The class AC^i is defined similarly to NC^i except gates are allowed to have unbounded fan-in (i.e., the OR and AND gates can be applied to more than two bits). $\diamondsuit$

We frequently use variants of AC^0 in this paper. For every integer $q \geq 2$, a language L is in $\mathsf{AC}^0[q]$ if L can be decided by circuits consisting of OR, AND, MOD_q gates with unbounded fan-in, and NOT gates, of $\mathsf{poly}(n)$ size and depth $O(1)$. For a constant $c \in \mathbb{N}^+$, AC^0_c denotes AC^0 circuits with an explicit depth c.

Definition 3.3 ($\oplus\mathsf{L/poly}$). A language L is in $\oplus\mathsf{L/poly}$ if the characteristic function $f(x) = [x \in L]$ satisfies the following: there is a constant c such that for each n, there is a non-deterministic Turing Machine M_n such that for all $x \in \{0,1\}^n$, $M_n(x)$ uses at most $c \log n$ space, and $f(x)$ equals to the parity of the number of accepting paths of $M_n(x)$. $\diamondsuit$

Definition 3.4 ($\widetilde{\mathsf{Sum}}_\delta \circ \mathcal{C}$). For δ being a constant or a function of n, we say a function $f : \{0,1\}^* \to \{0,1\}$ admits a family of $\widetilde{\mathsf{Sum}}_\delta \circ \mathcal{C}$ circuits, if $\exists s(n) \in \mathsf{poly}(n)$ and $\{C_n : \{0,1\}^n \to \{0,1\}^{s(n)}\} \in \mathcal{C}$, such that for each n, there exists a length-$s(n)$ vector $\alpha_n = (\alpha_{s(n),1}, \alpha_{s(n),2}, \ldots, \alpha_{s(n),s(n)}) \in \mathbb{R}^{s(n)}$, $\forall x \in \{0,1\}^n$,

$$|\langle \alpha_n, C_n(x) \rangle - f(x)| \leq \delta. \qquad \diamondsuit$$

Definition 3.5 (Collision resistant hash function family). For a function family

$$\mathcal{F} = \left\{ f_n : \{0,1\}^{l_k(n)} \times \{0,1\}^{l_{in}(n)} \to \{0,1\}^{l_{out}(n)} \right\}_{n \in \mathbb{N}}$$

with the key generation algorithm

$$\mathsf{Gen}_\mathcal{F} = \left\{ \mathsf{Gen}_{\mathcal{F},n} : \{0,1\}^* \to \{0,1\}^{l_k(n)} \right\},$$

we say it is a collision resistant hash function family, if for any adversary $\mathcal{A} = \{\mathcal{A}_n\}_{n \in \mathbb{N}}$, there exists negligible function $\mathrm{negl}(n)$ that $\forall n \in \mathbb{N}$,

$$\Pr\left[f_n(k, x_1) = f_n(k, x_2) \wedge x_1 \neq x_2 \;\middle|\; \begin{array}{l} k \leftarrow \mathsf{Gen}_{\mathcal{F},n} \\ x_1, x_2 \leftarrow \mathcal{A}_n(k) \end{array} \right] < \mathrm{negl}(n) \qquad \diamondsuit$$

We say that a collision resistant hash function family is in $\mathcal{C}$ iff $\mathcal{F}$ and $\mathsf{Gen}_\mathcal{F}$ are in $\mathcal{C}$. In other words, we require $\mathcal{F}$ to be uniform, in the sense that each $\mathcal{C}$ circuit computing f_n takes both the key $k \in \{0,1\}^{l_k(n)}$ and $x \in \{0,1\}^{l_{in}(n)}$ as input.

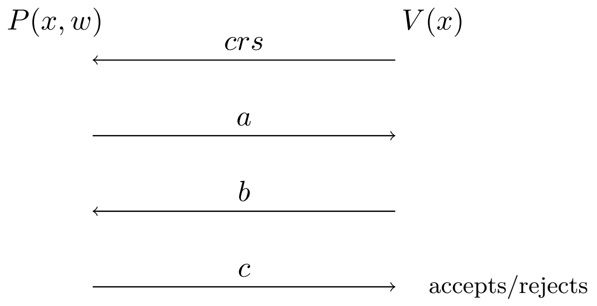
For simplicity, we will also denote this function family as

$$\mathcal{F} = \left\{ \mathcal{F}_n : \{0,1\}^{l_{in}(n)} \to \{0,1\}^{l_{out}(n)} \right\}_{n \in \mathbb{N}}.$$

We use $f \leftarrow \mathcal{F}_n$ to represent the process of generating a key k using $\mathsf{Gen}_{\mathcal{F},n}$ and get a circuit $f_n(k,\cdot)$ with a key hard-coded in it. The key generation circuit size of $\mathcal{F}$ is the size of circuit computing $f \leftarrow \mathcal{F}_n$, and the evaluation circuit size is the circuit size of $f \in \mathcal{F}_n$. These two sum up to the circuit size of $\mathcal{F}$. We use $l_{\mathcal{F}}(n)$ to represent the evaluation circuit size. Sometimes, we may use $f(x)$ for some x with length $|x| < l_{in}(n)$. You can interpret the input as padding x to $l_{in}(n)$ with 0s.

3.2 Interactive protocol

Definition 3.6 (3-round public coin protocol with common reference string (CRS)). A 3-round protocol with common reference string consists of 2 entities: the prover P and the verifier V. Here, the prover P consists of an interactive circuit family $\{P_n\}_{n\in\mathbb{N}}$, and the verifier V consists of an interactive circuit family $\{V_n\}_{n\in\mathbb{N}}$. Here, we let V_n generate the common reference string. The common input is x and the auxiliary input for the prover is w. The process of interaction is shown below:



We may omit the indices if they are clear from the content. We will use $P(x,w,crs)$ to represent the process of generating the first message and use $P(x,w,crs,a,b)$ to represent the process of generating the third message when the previous state of P is clear from the context. We will use $V(x,crs,a,b,c)$ to represent the process of verification. We will introduce the notation $V|_a$, signifying the interactive circuit V with the first message restricted to a. Also, we denote by $V|_{a_1,\cdots,a_t}$ the interactive circuit V with first t messages fixed. Note that the verifier will not keep the state after sending each message, so this notation is well-defined. We use $l_{crs}(n), l_a(n), l_b(n), l_c(n)$ to indicate the length of common reference string and each message and use $l_x(n), l_w(n)$ to represent the length of instance and witness when there is no ambiguity. We say that a 3-round public coin protocol is in $\mathcal{C}$ if there exists $\{P_{1,n}\}_{n\in\mathbb{N}}, \{P_{2,n}\}_{n\in\mathbb{N}}$ and $\{V_{1,n}\}_{n\in\mathbb{N}}, \{V_{2,n}\}_{n\in\mathbb{N}}$ in $\mathcal{C}$ such that the prover P_n behaves as follows:

1. Uniformly selects $r_1, r_2 \leftarrow \{0,1\}^*$.
2. $a \leftarrow P_{1,n}(x,w,crs,r_1)$, and outputs a, waits for b.
3. $c \leftarrow P_{2,n}(x,w,crs,a,b,r_1,r_2)$ and outputs c.

The verifier V_n behaves as follows:

1. Uniformly selects $r \leftarrow \{0,1\}^*$.

2. $crs \leftarrow V_{1,n}(r)$ and outputs crs, waits for a.

3. Uniformly selects $b \leftarrow \{0,1\}^{l_b(n)}$ and outputs b, waits for c.

4. Accepts if $V_{2,n}(x, crs, a, b, c) = 1$.

The total circuit size of (P, V) is defined to be the summation of circuit size of $P_{1,n}, P_{2,n}$ and $V_{1,n}, V_{2,n}$. We sometimes abbreviate the notation for the interaction for two parties P, V with common input x and auxiliary input to the prover w as $(P, V)(x, w)$, and we will use $(P, V)(x, w) = 1$ to denote the event that V accepts after the interaction.

Definition 3.7 (3-round public coin argument with CRS for language $\mathcal{L}$ against $\mathcal{D}$). Let $l_x(n)$ be the length of the instance from $\mathcal{L}$ on security parameter n. Let $\mathcal{R}$ be the relation underlying $\mathcal{L}$ (i.e., $x \in \mathcal{L}$ iff $\exists w : (x, w) \in \mathcal{R}$) and $l_w(n)$ be the witness length on security parameter n. A 3-round public coin argument with CRS for language $\mathcal{L}$ against $\mathcal{D}$ is a 3-round public-coin protocol (P, V) with common reference string that satisfies:

- **Completeness**: $\forall n \in \mathbb{N}$, $\forall x \in \{0,1\}^{l_x(n)}$, $\forall w \in \{0,1\}^{l_w(n)}$ s.t. $(x, w) \in \mathcal{R}$,
$$\Pr[(P, V)(x, w) = 1] = 1,$$
where the probability is over the randomness of P, V.

- **Computational soundness against $\mathcal{D}$**: For any cheating prover $\tilde{P} = \left\{\tilde{P}_n\right\}_{n\in\mathbb{N}}$ computable in $\mathcal{D}$, there exists a negligible function negl(n) that, $\forall n \in \mathbb{N}$, $\forall x \in \{0,1\}^{l_x(n)}$ s.t. $x \notin \mathcal{L}$,
$$\Pr\left[(\tilde{P}, V)(x) = 1\right] < \text{negl}(n),$$
where the probability is over the randomness of $\tilde{P}, V$. Sometimes we also say (P, V) is secure against $\mathcal{D}$ for the same meaning.

Additionally, we say (P, V) is an argument of knowledge in $\mathcal{C}$ if it satisfies:

- **Knowledge extraction in $\mathcal{C}$ against $\mathcal{D}$**: For all polynomial $p(n)$, there exists an extractor $E^{\tilde{P}}$ in $\mathcal{C}$ with oracle access to some circuit $\tilde{P}$ such that for any prover $\tilde{P} = \left\{\tilde{P}_n\right\}_{n\in\mathbb{N}}$ computable in $\mathcal{D}$ such that for every $x \in \{0,1\}^{l_x(n)}$, if
$$\Pr[(\tilde{P}, V)(x) = 1] \geq \frac{1}{p(n)},$$
then
$$\Pr[(x, w) \in \mathcal{R} \mid w \leftarrow E^{\tilde{P}}(x)] = 1.$$
Here the relation $\mathcal{R}$ is the same as the one in **Completeness**. $\diamondsuit$

3.3 Fiat-Shamir for 3-round public coin argument with common reference string

Definition 3.8 (Fiat-Shamir hash function). Given four polynomials $l_k(n), l_x(n), l_a(n), l_b(n)$, a function family of the form

$$\mathcal{H} = \left\{ h_n : \{0,1\}^{l_k(n)} \times \{0,1\}^{l_x(n)+l_a(n)} \to \{0,1\}^{l_b(n)} \right\}_{n\in\mathbb{N}}$$

with the key generation algorithm

$$\mathsf{Gen}_{\mathcal{H}} = \left\{ \mathsf{Gen}_{\mathcal{H},n} : \{0,1\}^* \to \{0,1\}^{l_k(n)} \right\}_{n\in\mathbb{N}}$$

is called a Fiat-Shamir hash function family. Here, l_x is the input length of the protocol, l_a is the length of the first message, and l_b is the length of the second message. $\diamondsuit$

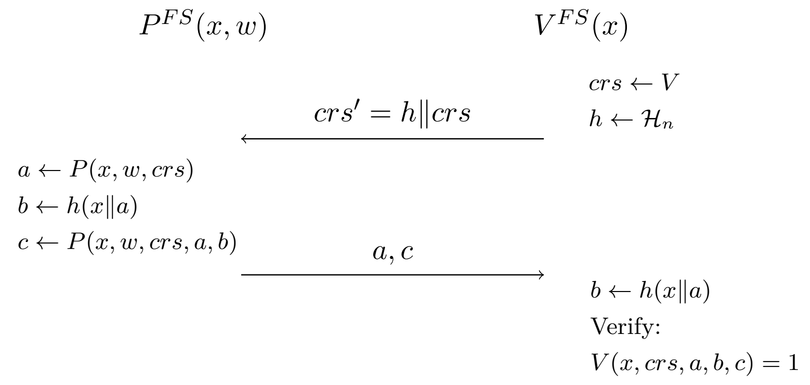
We say that a Fiat-Shamir hash function is in $\mathcal{C}$ iff $\mathcal{H}$ and $\mathsf{Gen}_{\mathcal{H}}$ are computable in $\mathcal{C}$. In other words, we require the hash $\mathcal{H}$ to be uniform, in the sense that each $\mathcal{C}$ circuit computing h_n takes both the key $k \in \{0,1\}^{l_k(n)}$ and the instance concatenating the first message $x\|a \in \{0,1\}^{l_x(n)+l_a(n)}$ as input.

For simplicity, we will also denote this function family as

$$\mathcal{H} = \left\{ \mathcal{H}_n : \{0,1\}^{l_x(n)+l_a(n)} \to \{0,1\}^{l_b(n)} \right\}_{n\in\mathbb{N}}.$$

We use $h \leftarrow \mathcal{H}_n$ to represent the process of generating a key k using $\mathsf{Gen}_{\mathcal{H},n}$ and get a circuit $h_n(k,\cdot)$ with a key hard-coded in it. The key generation circuit size of $\mathcal{H}$ is the size of the circuit computing $h \leftarrow \mathcal{H}_n$, and the evaluation circuit size is the circuit size of $h \in \mathcal{H}_n$. These two sum up to the circuit size of $\mathcal{H}$. We use $l_{\mathcal{H}}(n)$ to represent the evaluation circuit size.

Definition 3.9 (The Fiat-Shamir transform). Given a 3-round public coin protocol (P, V) with common reference string and a Fiat-Shamir hash function family $\mathcal{H}, \mathsf{Gen}_{\mathcal{H}}$. The Fiat-Shamir transform on (P, V) with respect to $\mathcal{H}$ results in a new protocol (P^{FS}, V^{FS}). The prover P^{FS} consists of a circuit family $\{P_n^{FS}\}_{n\in\mathbb{N}}$ and the verifier V^{FS} consists of a circuit family $\{V_n^{FS}\}_{n\in\mathbb{N}}$. The protocol interacts as follows:



For (P^{FS}, V^{FS}) proving language $\mathcal{L}$, we define several properties:

- **Completeness**: Let $\mathcal{R}$ be the relation underlying $\mathcal{L}$ and $l_w(n)$ be the witness length. Then $\forall n \in \mathbb{N}$, $\forall x \in \{0,1\}^{l_x(n)}$, $\forall w \in \{0,1\}^{l_w(n)}$ s.t. $(x,w) \in \mathcal{R}$
$$\Pr\left[(P^{FS}, V^{FS})(x,w) = 1\right] = 1,$$
where the probability is over the randomness of P^{FS}, V^{FS}.

- **Computational Soundness against** $\mathcal{D}$: For each adversary $P^* = \{P_n^*\}_{n\in\mathbb{N}}$ computable in $\mathcal{D}$, there exists a negligible function $\mathrm{negl}(n)$ that, $\forall n \in \mathbb{N}$, $\forall x \in \{0,1\}^{l_x(n)}$ s.t. $x \notin \mathcal{L}$,
$$\Pr\left[(P^*, V^{FS})(x) = 1\right] < \mathrm{negl}(n),$$
where the probability is over the randomness of P^*, V^{FS}.

If (P,V) is 3-round public coin argument with CRS for language $\mathcal{L}$ against $\mathcal{D}$, and (P^{FS}, V^{FS}) satisfies **completeness** and **computational soundness against** $\mathcal{D}$, we say that applying Fiat-Shamir transformation to the argument with $\mathcal{H}$ gives an argument against adversary in $\mathcal{D}$ for $\mathcal{L}$. Sometimes we also say that it is secure against $\mathcal{D}$ for the same meaning.

4 Correlation Intractability Against Bounded-Depth Adversaries

In this section we show positive results for Fiat-Shamir for proofs against bounded-depth adversaries. We first observe that the construction of correlation intractable (CI) hash functions of Peikert and Shiehian [PS19] is computable in TC^0, therefore it has already implied the existence of a CI hash function family computable in TC^0 against adversaries in circuit class $\mathcal{D} \supseteq \mathsf{TC}^0$, assuming SIS is hard against $\mathcal{D}$. We then show our main result: the construction of CI hash functions computable in $\mathsf{AC}^0[2]$ against adversaries in any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, assuming $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \widetilde{\mathsf{Sum}}_{n^{-c}} \circ \mathcal{C}$ for some constant $c > 0$, which is a worst-case complexity assumption.

The rest of Section 4 is organized as follows. In Section 4.1 we provide the background needed for this section. In Section 4.2 we present the construction in [PS19]. The reasons we choose to present the construction in [PS19] are:

- We emphasize that the construction and hardness reduction can be computed in TC^0;
- The full construction makes the latter construction (which is our main result) more understandable.

Then in Section 4.3 we present our main construction of CI hash function computable in $\mathsf{AC}^0[2]$ against low-depth adversaries.

4.1 Backgrounds needed in this section

4.1.1 Hard Lattice Problems

Definition 4.1. The $\mathsf{SIS}_{n,m,q,\beta}$ problem is, given a uniformly random matrix $A \in \mathbb{Z}_q^{n\times m}$, find a non-zero integral vector $z \in \mathbb{Z}^m$ such that $Az \equiv 0 \pmod{q}$ and $\|z\| \leq \beta$. $\diamondsuit$

Definition 4.2. The $\mathsf{LWE}_{n,m,q,\chi}$ problem is to distinguish, with non-negligible advantage, between m independent samples for a single $s \leftarrow \mathbb{Z}_q^n$, each sampled by choosing a uniformly random $a \leftarrow \mathbb{Z}_q^n$, $e \leftarrow \chi$ and outputing $(a, b = a \cdot s + e) \in \mathbb{Z}_q^{n+1}$, and m uniformly random and independent samples over $\mathbb{Z}_q^{n+1}$. $\diamondsuit$

4.1.2 Fully Homomorphic Encryption

Definition 4.3 (Strongly Fully Homomorphic Encryption (sFHE) for $\mathcal{D}$ against $\mathcal{C}$). A sFHE scheme for $\mathcal{D}$ circuits is a function family $\mathsf{FHE} = \{\mathsf{FHEGen}_n, \mathsf{Enc}_n, \mathsf{Dec}_n, \mathsf{Eval}_n\}_{n\in\mathbb{N}}$ with the following syntax.

- FHEGen_n returns a public/secret key pair $(\mathsf{pk}, \mathsf{sk})$.
- FHEGen'_n returns a public key pk.
- $\mathsf{Enc}_n(\mathsf{pk}, m \in \{0,1\}; r \in \{0,1\}^*)$ returns a ciphertext ct using random bits r.
- $\mathsf{Dec}_n(\mathsf{sk}, \mathsf{ct})$ (deterministically) returns a message $m \in \{0,1\}$.
- $\mathsf{Eval}_n(\mathsf{pk}, f \in \mathcal{D}, (\mathsf{ct}_1, \mathsf{ct}_2, \ldots, \mathsf{ct}_s))$ (deterministically) return a ciphertext ct (s is the input size of f).
- $\mathsf{RandEval}_n(\mathsf{pk}, f \in \mathcal{D}, (m_1, \ldots, m_s), (r_1, \ldots, r_s))$ (deterministially) returns a randomness $r \in \mathcal{R}$ (again s is the input size of f).

Furthermore, it satisfies the following properties:

- **Correctness**: we have $m = \mathsf{Dec}_n(\mathsf{sk}, \mathsf{Enc}_n(\mathsf{pk}, m; r))$ for all $n \in \mathbb{N}$ and $m \in \{0,1\}$ with probability 1 over $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHEGen}_n$ and $r \leftarrow \{0,1\}^*$.
- **CPA security**: For any adversary $\mathcal{A} = \{\mathcal{A}_n\} \in \mathcal{C}$,

  $$\left| \Pr\left[\mathcal{A}_n(\mathsf{pk}) = 1 \mid (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHEGen}_n\right] - \Pr\left[\mathcal{A}_n(\mathsf{pk}) = 1 \;\middle|\; \mathsf{pk} \leftarrow \mathsf{FHEGen}'_n\right] \right| < \mathsf{negl}(n),$$

  and the two distribution families

  $$\{(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, 0)) : \mathsf{pk} \leftarrow \mathsf{FHEGen}'_n\}, \{(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, 1)) : \mathsf{pk} \leftarrow \mathsf{FHEGen}'_n\}$$

  are statistically indistinguishable.
- **Strong homomorphism**: for every function familly $f = \{f_n\} \in \mathcal{D}$, all $n \in \mathbb{N}$, all $(\mathsf{pk}, \mathsf{sk})$ generated by FHEGen_n and all $m_1, \ldots, m_s \in \{0,1\}$ and $r_1, \ldots, r_s \in \mathcal{R}$ ($s = s(n)$ is the input size of f_n), we have

  $$\begin{aligned} &\mathsf{Eval}_n(\mathsf{pk}, f_n, (\mathsf{Enc}(\mathsf{pk}, m_1; r_1), \ldots, \mathsf{Enc}(\mathsf{pk}, m_s; r_s))) \\ =&\mathsf{Enc}_n(\mathsf{pk}, f_n(m_1, \ldots, m_s); \mathsf{RandEval}_n(\mathsf{pk}, f_n, (m_1, \ldots, m_s), (r_1, \ldots, r_s))). \end{aligned} \qquad \diamondsuit$$

Remark. Both GSW-FHE and WP-FHE defined later are sFHE schemes for at least bounded-degree polynomials under certain assumptions, although our proof does not rely on this fact. Also, in these two schemes, $\mathsf{RandEval}$ is given implicitly by Eval.

Definition 4.4 (Lattice Gadgets over $\mathbb{Z}_q$). For a positive integer modulus q, let $l = \lceil \lg q \rceil$. We define the "gadget" vector as

$$g^T = (1, 2, \ldots, 2^{l-1}) \in \mathbb{Z}_q^l.$$

For every $u \in \mathbb{Z}_q$, there is an efficiently computable binary vector $g^{-1}(u) \in \{0,1\}^l$ such that $g \cdot g^{-1}(u) \equiv u \pmod{q}$, i.e., the binary representation of u. We treat $g^{-1} : \mathbb{Z}_q \to \{0,1\}^l$ as a function that computes binary decomposition.

The gadget matrix is defined as $G_n = I_n \otimes g^T$, and sometimes we drop the subscript n. Similarly, we define the function $G^{-1} = (I \otimes g^{-1}) : \mathbb{Z}_q^n \to \{0,1\}^{nl}$, which applies g^{-1} to each coordinate and concatenates the results. For any $u \in \mathbb{Z}_q^n$, we have

$$G \cdot G^{-1}(u) = u. \qquad \diamondsuit$$

Construction 1 (GSW-FHE). GSW-FHE is given by the function family $\mathsf{GSW\text{-}FHE} = \{\mathsf{GSW\text{-}FHEGen}_n, \mathsf{GSW\text{-}Enc}_n, \mathsf{GSW\text{-}Dec}_n, \mathsf{GSW\text{-}Eval}_n\}_{n \in \mathbb{N}}$, parameterized by the lattice dimension n, the error distribution χ, the modulus q, and the number of samples $m = n \lceil \log q \rceil$. We assume these parameters are chosen properly so that $\mathrm{LWE}_{n-1,2m,q,\chi}$ is believed to be hard.

- $\mathsf{GSW\text{-}FHEGen}_n$:
  1. Take $A' \leftarrow \mathbb{Z}_q^{(n-1)\times 2m}$, $s' \leftarrow \mathbb{Z}_q^{n-1}$, and $e \leftarrow \chi^{2m}$.
  2. Compute $b = A'^T s' + e \in \mathbb{Z}_q^{2m}$.
  3. Return
  $$\mathsf{pk} = A = \begin{pmatrix} A' \\ b^T \end{pmatrix} \in \mathbb{Z}_q^{n \times 2m}, \mathsf{sk} = k = \begin{pmatrix} -s' \\ 1 \end{pmatrix} \in \mathbb{Z}_q^n.$$

  (Note that $k^T A = e^T \approx 0$.)

- $\mathsf{GSW\text{-}FHEGen}'_n$: return $\mathsf{pk} = A \leftarrow \mathbb{Z}_q^{n \times 2m}$ sampled uniformly at random.

- $\mathsf{GSW\text{-}Enc}_n(\mathsf{pk} = A, \mu \in \{0,1\})$:
  1. Take $R \leftarrow \{0,1\}^{2m \times m} \in \mathbb{Z}_q^{2m \times m}$.
  2. Return $\mathsf{ct} = AR + \mu G \in \mathbb{Z}_q^{n \times m}$.

  We allow $\mathsf{GSW\text{-}Enc}_n$ to take a vector in $\mathbb{Z}_2^m$ as the input, instead of a single bit. In this case, the encryption of each dimension of that vector (using independent randomness) is concatenated together by each row. That is, for a length-m vector x, $\mathsf{GSW\text{-}Enc}(\mathsf{pk} = A, x) = AR + x^T \otimes I_n \otimes g^T$, where $R \leftarrow \mathbb{Z}_2^{2m \times m^2}$.

- $\mathsf{GSW\text{-}Dec}_n(\mathsf{sk} = k, \mathsf{ct})$:
  1. Take $w = (0, 0, \ldots, 0, \lceil q/2 \rceil)^T \in \mathbb{Z}_q^n$.
  2. Compute the value of $V = k^T \mathsf{ct} \cdot G^{-1}(w)$, output the decrypted message as $\left\lceil \frac{V}{q/2} \right\rfloor$.

- $\mathsf{GSW\text{-}Eval}_n$. We describe it by giving how to homomorphically compute addition and multiplication in $\mathbb{Z}_q$.
  - $\mathsf{GSW\text{-}Eval}_n^{\mathrm{add}}(\mathsf{pk} = A, (\mathsf{ct}_0, \mathsf{ct}_1))$: return $\mathsf{ct}_0 + \mathsf{ct}_1 - 2\mathsf{ct}_0 \cdot G^{-1}(\mathsf{ct}_1) \in \mathbb{Z}_q^{n \times m}$.
  - $\mathsf{GSW\text{-}Eval}_n^{\mathrm{mul}}(\mathsf{pk} = A, (\mathsf{ct}_0, \mathsf{ct}_1))$: return $\mathsf{ct}_0 \cdot G^{-1}(\mathsf{ct}_1) \in \mathbb{Z}_q^{n \times m}$. $\diamondsuit$

Remark. For any constant-degree polynomial function family $\{f_n\}$, the corresponding circuit family computing $\mathsf{GSW\text{-}Eval}_n(\mathsf{pk}, f_n, (\mathsf{ct}_1, \ldots))$ is in TC^0. WP-FHE, which will be defined later, has a stronger property: the circuit family computing homomorphic evaluation of constant-degree polynomial function is in $\mathsf{AC}^0[2]$.

4.1.3 Inert Commitment

Let $A \in \mathbb{Z}_q^{n \times 2m}$ be the public key of GSW-FHE, and let $C_i = AR_i + x_i G$ be the homomorphic encryption of scalar $x_i \in \{0,1\}$, for $i \in [m]$. Then we concatenate all C_i's together, and view it as a commitment (relative to A) to $x^T = (x_1, \cdots, x_m)^T$ under randomness $R = \big(R_1, \quad \cdots, \quad R_m\big)$, i.e.

$$C = \big(C_1, \quad \cdots \quad, C_m\big) = AR + x^T \otimes G = AR + x^T \otimes I_n \otimes g^T = \mathsf{GSW\text{-}Enc}(A, x).$$

Since any matrix $M \in \mathbb{Z}_q^{n \times m}$ can be "vectorized" as an $v_M \in \mathbb{Z}_q^{nm}$, so that $(x^T \otimes I_n) \cdot v_M = Mx$, we have

$$c_M = C \cdot G^{-1}(v_M) = A \underbrace{(R \cdot G^{-1}(v_M))}_{r_M} + (x^T \otimes I_n \otimes g^T) \cdot (I_m \otimes I_n \otimes g^{-t}) v_M = Ar_M + Mx \in \mathbb{Z}_q^n.$$

We view c_M as an "inert commitment" to $Mx \in \mathbb{Z}_q^n$, and we write it as $\mathsf{InertEval}(M, C)$.

Definition 4.5 (Inert Commitment). Given $C \in \mathbb{Z}_q^{n \times m^2}$ as the homomorphic encryption of some length-m vector, and $M \in \mathbb{Z}_q^{n \times m}$ as a matrix, we write

$$\mathsf{InertEval}(M, C) := c_M = C \cdot G^{-1}(v_M),$$

where $v_M \in \mathbb{Z}_2^{nm}$ is the vector satisfies $(x^t \otimes I_n) \cdot v_M = Mx$ for any length-m vector x, obtained by reordering the elements of M. $\diamondsuit$

4.1.4 Correlation intractability

Definition 4.6. We say a relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{Y}$ is searchable in a set of functions $\mathcal{F}$ if there exists a function $f : \mathcal{X} \to \mathcal{Y}$ in $\mathcal{F}$ such that if $(x, y) \in \mathcal{R}$ then $y = f(x)$.

We say a relation class $\mathcal{S} = \{\mathcal{S}_n\}_{n \in \mathbb{N}}$ (i.e. a set of relations for each n) is searchable in a function family $\mathcal{F} = \{\mathcal{F}_n\}_{n \in \mathbb{N}}$ if for each $n \in \mathbb{N}$, for all $\mathcal{R} \in \mathcal{S}_n$, $\mathcal{R}$ is searchable in $\mathcal{F}_n$. $\diamondsuit$

Definition 4.7. Let $\mathcal{S} = \{\mathcal{S}_n\}$ be a relation class and $\mathcal{C}$ be a circuit class. A hash function family

$$\mathcal{H} = \left\{h_n : \{0,1\}^{l_k(n)} \times \{0,1\}^{l_{in}(n)} \to \{0,1\}^{l_{out}(n)}\right\}$$

with a key generation algorithm

$$\mathsf{Gen}_{\mathcal{H}} = \left\{\mathsf{Gen}_{\mathcal{H},n} : \{0,1\}^* \to \{0,1\}^{l_k(n)}\right\}$$

is correlation intractable for $\mathcal{S}$ against $\mathcal{C}$ if for every circuit family $\mathcal{A} = \{\mathcal{A}_n\} \in \mathcal{C}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $n \in \mathbb{N}$, for every $\mathcal{R} \in \mathcal{S}_n$,

$$\Pr\left[(x, h_n(k, x)) \in \mathcal{R} \;\middle|\; \begin{array}{l} k \leftarrow \mathsf{Gen}_{\mathcal{H},n} \\ x \leftarrow \mathcal{A}_n(k) \end{array}\right] < \mathsf{negl}(n).$$

We omit the polynomials $l_k(n), l_{in}(n), l_{out}(n)$ when they are clear from the explicit construction. $\diamondsuit$

4.1.5 Cryptographic primitives against bounded-depth circuits

We consider the capability of adversaries as some circuit class that at least contains $\mathsf{AC}^0[2]$ (thus can compute matrix multiplication in $\mathbb{Z}_2$), with the assumption that the same circuit class augmented by an error-tolerant majority gate is not able to compute $\oplus\mathsf{L}/\mathsf{poly}$.

Assumption 4.8. For a certain circuit class $\mathcal{C}$ s.t. $\mathsf{AC}^0[2] \subseteq \mathcal{C}$, there exists a constant $c > 0$ s.t. $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \widetilde{\mathsf{Sum}}_{1/n^c} \circ \mathcal{C}$.

For example, Assumption 4.8 is widely believed to be true for $\mathcal{C} \in \{\mathsf{AC}^0[2], \mathsf{TC}^0, \mathsf{NC}^1\}$.

Remark. When $\mathcal{C} = \mathsf{NC}^1$, the assumption above is equivalent to $\oplus\mathsf{L}/\mathsf{poly} \neq \mathsf{NC}^1$ as in [DVV16].

Definition 4.9 (ZeroSamp, OneSamp). Let n be the security parameter. For $r^{(1)} \in \mathbb{Z}_2^{n(n-1)/2}$, and $r^{(2)} \in \mathbb{Z}_2^{n-1}$, denote

$$L(r^{(1)}) = \begin{pmatrix} 1 & r_1^{(1)} & r_2^{(1)} & \cdots & r_{n-1}^{(1)} \\ & 1 & r_n^{(1)} & \cdots & r_{2n-3}^{(1)} \\ & & \ddots & \ddots & \vdots \\ & & & 1 & r_{n(n-1)/2}^{(1)} \\ & & & & 1 \end{pmatrix}, R(r^{(2)}) = \begin{pmatrix} 1 & 0 & 0 & 0 & r_1^{(2)} \\ & 1 & 0 & 0 & r_2^{(2)} \\ & & \ddots & \vdots & \vdots \\ & & & 1 & r_{n-1}^{(2)} \\ & & & & 1 \end{pmatrix}.$$

Also, let M_0^n, M_1^n be the following $n \times n$ matrices:

$$M_0 = \begin{pmatrix} \mathbf{0}^T & 0 \\ I_{n-1} & \mathbf{0} \end{pmatrix}, M_1 = \begin{pmatrix} \mathbf{0}^T & 1 \\ I_{n-1} & \mathbf{0} \end{pmatrix} \quad (I_{n-1} \text{ is the identity matrix of rank } n-1).$$

Then we define two sampling procedures:

- $\mathsf{ZeroSamp}(n)$: take $r^{(1)} \leftarrow \{0,1\}^*, r^{(2)} \leftarrow \{0,1\}^*$ and output $L(r^{(1)})M_0^n R(r^{(2)})$.
- $\mathsf{OneSamp}(n)$: take $r^{(1)} \leftarrow \{0,1\}^*, r^{(2)} \leftarrow \{0,1\}^*$ and output $L(r^{(1)})M_1^n R(r^{(2)})$.

Note that the output of $\mathsf{ZeroSamp}(n)$ always has rank $n-1$, and the output of $\mathsf{OneSamp}(n)$ always has rank n. What's more, the random bits used by $A \leftarrow \mathsf{ZeroSamp}(n)$ also gives the kernel of A, and therefore we sometimes use $A, k \leftarrow \mathsf{ZeroSamp}(n)$ where k s.t. $Ak = 0$ is additionally given by $\mathsf{ZeroSamp}(n)$.

If the security parameter n is clear from context, we may drop it and use the notation $\mathsf{ZeroSamp}$, $\mathsf{OneSamp}$. ♢

Previous literature discovered that NC^1 circuits cannot distinguish $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$ if $\oplus\mathsf{L}/\mathsf{poly} \neq \mathsf{NC}^1$. We extend this idea from NC^1 to other circuit classes that contain $\mathsf{AC}^0[2]$, as stated in the following lemma.

Lemma 4.10. For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, if Assumption 4.8 holds for $\mathcal{C}$, there is a negligible function $\mathsf{negl}(\cdot)$ such that for any family $\mathcal{F} = \{f_n\}$ in $\mathcal{C}$, for an infinite number of values of n,

$$|\Pr[f_n(M) = 1 \mid M \leftarrow \mathsf{ZeroSamp}(n)] - \Pr[f_n(M) = 1 \mid M \leftarrow \mathsf{OneSamp}(n)]| < \mathsf{negl}(n). \quad ♢$$

Remark. As Remark 3.1 of [DVV16] points out, from worst-case complexity assumptions, we can only achieve infinitely-often primitives, as if we assume $\mathcal{D} \nsubseteq \mathcal{C}$ where $\mathcal{C}, \mathcal{D}$ are circuit classes, we only mean that there exists a function family $\{f_n\} \in \mathcal{D}$, such that for any function family $\{g_n\} \in \mathcal{C}$, $f_n \neq g_n$ for infinitely many n.

However, as long as we strengthen the assumption from the previous one to that there exists $\{f_n\} \in \mathcal{D}$, for all $\{g_n\} \in \mathcal{C}$, $\exists n_0$ such that $f_n \neq g_n$ for all $n > n_0$, we can achieve the conventional almost-everywhere security. We state this observation below. All our constructions assuming 4.8 automatically achieve almost-everywhere security using the strengthened assumption, and we only state theorems in the infinitely-often secure version for brevity.

Assumption 4.11 (Strengthened assumption). For a certain circuit class $\mathcal{C}$ s.t. $\mathsf{AC}^0[2] \subseteq \mathcal{C}$, there exists $c > 0$ s.t. $\exists \{f_n\} \in \oplus\mathsf{L/poly}$ such that for any function family $\{g_n\} \in \widetilde{\mathsf{Sum}}_{1/n^c} \circ \mathcal{C}$, $\exists n_0$ such that $f_n \neq g_n$ for all $n > n_0$.

Lemma 4.12. For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, if Assumption 4.11 holds for $\mathcal{C}$, there is a negligible function $\mathsf{negl}(\cdot)$ such that for any family $\mathcal{F} = \{f_n\}$ in $\mathcal{C}$, for all but infinitely many n,

$$| \Pr [f_n(M) = 1 \mid M \leftarrow \mathsf{ZeroSamp}(n)] - \Pr [f_n(M) = 1 \mid M \leftarrow \mathsf{OneSamp}(n)] | < \mathsf{negl}(n). \qquad \diamondsuit$$

Since we extend previous work from NC^1 to general circuit classes $\mathcal{C}$, and our previous lemmas 4.10, 4.12 are slightly different from those in literature (see lemma 4.3 of [DVV16], where $n = n(\lambda)$ is polynomial of security parameter), we include proofs of those lemmas in Appendix A.

Construction 2 (WP-FHE [WP22]). WP-FHE is given by the function family $\mathsf{WP\text{-}FHE} = \{ \mathsf{WP\text{-}FHEGen}_n, \mathsf{WP\text{-}Enc}_n, \mathsf{WP\text{-}Dec}_n, \mathsf{WP\text{-}Eval}_n\}_{n\in\mathbb{N}}$.

- $\mathsf{WP\text{-}FHEGen}_n$: Take $A^T, k \leftarrow \mathsf{ZeroSamp}(n)$, and set $\mathsf{pk} = A, \mathsf{sk} = k$.
- $\mathsf{WP\text{-}FHEGen}'_n$: Take $A^T \leftarrow \mathsf{OneSamp}(n)$, and then output $\mathsf{pk} = A$.
- $\mathsf{WP\text{-}Enc}_n(\mathsf{pk} = A, m \in \{0,1\})$: Take $R \leftarrow \mathbb{Z}_2^{n\times n}$, then return $\mathsf{ct} = AR + mI_n \in \mathbb{Z}_2^{n\times n}$. We allow $\mathsf{WP\text{-}Enc}_n$ to take a vector in $\mathbb{Z}_2^n$ as input, instead of a single bit. In this case, the encryption of each dimension of that vector (using independent randomness) is concatenated together by each row. That is, for a length-n vector x, $\mathsf{WP\text{-}Enc}(\mathsf{pk} = A, x) = AR + x^T \otimes I_n$, where $R \leftarrow \mathbb{Z}_2^{n\times n^2}$.
- $\mathsf{WP\text{-}Dec}_n(\mathsf{sk} = k, \mathsf{ct})$: Let c be the n-th column vector of ct, and then return $k \cdot c$.
- $\mathsf{WP\text{-}Eval}_n$. We describe it by giving how to homomorphically compute addition and multiplication in $\mathbb{Z}_2$.
  - $\mathsf{WP\text{-}Eval}_n^{\text{add}}(\mathsf{pk} = A, (\mathsf{ct}_0, \mathsf{ct}_1))$: return $\mathsf{ct}_0 + \mathsf{ct}_1 \in \mathbb{Z}_2^{n\times n}$.
  - $\mathsf{WP\text{-}Eval}_n^{\text{mul}}(\mathsf{pk} = A, (\mathsf{ct}_0, \mathsf{ct}_1))$: return $\mathsf{ct}_0\mathsf{ct}_1 \in \mathbb{Z}_2^{n\times n}$. $\diamondsuit$

Remark. We emphasize the striking similarity between GSW-FHE and WP-FHE. The $(\mathsf{pk}, \mathsf{sk})$ pairs of both schemes are (pk= A, sk= k) where A is a matrix, k is a vector with $k^TA \approx 0$ in GSW-FHE, and $k^TA = 0$ in WP-FHE. What's more, if we treat the identity matrix I as G's analog in $\mathbb{Z}_2$, then the encryption of both schemes is $\mathsf{Enc}(\mathsf{pk} = A, m) = AR + mG$. Similarly, homomorphic addition of both schemes is adding two ciphertexts together, and $\mathsf{Eval}^{\text{mul}}(\mathsf{ct}_0, \mathsf{ct}_1) = \mathsf{ct}_0G^{-1}(\mathsf{ct}_1)$ for both schemes. Recall Table 1 as a summary of the above observations.

Our proof uses the following lemma:

Lemma 4.13 ([EWT21]). For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, if Assumption 4.8 holds for $\mathcal{C}$, then $\{p_A(x) = Ax : A \leftarrow \text{OneSamp}\}$ is a collection of one-way permutations against $\mathcal{C}$, that is, for any adversary family $\{\mathcal{A}_n\}_{n\in\mathbb{N}} \in \mathcal{C}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for infinitely many n,

$$\Pr\left[Ay = Ax \;\middle|\; \begin{array}{l} A \leftarrow \mathsf{OneSamp}(n) \\ x \leftarrow \mathbb{Z}_2^n \\ y \leftarrow \mathcal{A}_n(A, Ax) \end{array}\right] < \mathsf{negl}(n). \qquad \diamondsuit$$

[EWT21] only proves the lemma for $\mathcal{C} = \mathsf{NC}^1$, but actually their proof can be applied to any circuit class $\mathcal{C}$ that satisfies requirement 4.8. We give a sketch of their proof.

Proof sketch. The idea is to use the adversary $\{\mathcal{A}_n\}_{n\in\mathbb{N}}$ that breaks one-wayness with probability $1/p(n)$ to distinguish $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$. First, checking $Ay = Ax$ involves only matrix multiplication, so we can assume $\mathcal{A}_n$ returns y with $Ay = Ax$ or $\perp$. Since $\{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ cannot distinguish $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$, we have

$$\Pr\left[Ay = Ax \;\middle|\; \begin{array}{l} A \leftarrow \mathsf{ZeroSamp}(n) \\ x \leftarrow \mathbb{Z}_2^n \\ y \leftarrow \mathcal{A}_n(A, Ax) \end{array}\right] > \frac{1}{p(n)} - \mathsf{negl}(n) > \frac{1}{p'(n)}.$$

For A sampled from $\mathsf{ZeroSamp}$, whenever $\mathcal{A}_n$ finds y s.t. $Ay = Ax$, we have probability $\frac{1}{2}$ that $x \neq y$ due to the randomness of x (recall A is of rank $n-1$, so y has two solutions), in which case we have $x - y$ as a non-trivial kernel of A. Therefore by invoking $\mathcal{A}_n$ we have non-negligible probability to obtain the kernel of A, which can be used to distinguish $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$ (recall that the output of $\mathsf{OneSamp}$ is always full-rank).

We emphasize that only matrix multiplication in $\mathbb{Z}_2$ is used in this reduction, and thus the reduction is computable in $\mathcal{C} \supseteq \mathsf{AC}^0[2]$. $\square$

4.2 CI from SIS against bounded-depth adversaries

In this subsection, we follow the construction of [PS19] to show that it can be computed in TC^0, yielding a CI hash against bounded-depth adversary under the super-weak assumption of SIS against bounded-depth adversaries.

Theorem 4.14. For any circuit class $\mathcal{C}$ that contains TC^0, for any polynomial $S(n)$ and constant d, assuming the hardness of $\mathsf{SIS}_{n,m+1,q,\beta}$ for $q = \mathsf{poly}(n), l = \lceil \log q \rceil, m = nl$ and sufficiently large $\beta = m^{O(d)}$ against $\mathcal{C}$, there exists a hash family computable in TC^0 that is CI for any relation class searchable by size-$S(n)$[6], degree-d polynomials $\{f_n : \{0,1\}^n \to \{0,1\}^m\}_{n\in\mathbb{N}}$, against adversaries in $\mathcal{C}$. $\diamondsuit$

By lemma B.3, when n is large enough, by setting $q \geq n^{\Omega(d)}$ s.t. $q \geq \beta \cdot n^{\Omega(1)}$, there is an polynomial time reduction from $n^{\Omega(d)}$-approximate SIVP to $\mathsf{SIS}_{n,m+1,q,\beta}$.

Remark. The main difficulty in computing the construction in [PS19] lies in computing the sum of polynomially many numbers in $\mathbb{Z}_q$, where q is polynomially large.

[6]Here "size" means the encoding length of the constant degree polynomial.

Remark. The hash function is only CI for constant-degree polynomials because TC^0 circuits can only perform a constant number of homomorphic multiplications (which involve matrix multiplications). Therefore, achieving constant-degree polynomials is the best we can do with FHE-based constructions.

Construction 3. The hash family $\mathcal{H} = \{h_n\}_{n\in\mathbb{N}}$ with key generation algorithm $\mathsf{Gen}_{\mathcal{H}} = \{\mathsf{Gen}_{\mathcal{H},n}\}_{n\in\mathbb{N}}$ is parameterized by an arbitrary circuit size $S(n) = \mathsf{poly}(n)$ and depth d. Let $\mathcal{U}(C,x) = C(x)$ denote the universal circuit for size-S, degree-d polynomials (hence, it is also a constant degree polynomial itself). We write $q = q(n), l = \lceil \log q \rceil, m = nl$.

- $\mathsf{Gen}_{\mathcal{H},n}$: generate $A \leftarrow \mathbb{Z}_q^{n\times 2m}$ and $C \leftarrow \mathsf{GSW\text{-}Enc}(A, 0^{S(n)}) = AR + 0^{S(n)} \otimes G$, where $R \in \mathbb{Z}^{2m\times m}$. Choose a uniform random $a \leftarrow \mathbb{Z}_q^n$, output the hash key $k = (a, C)$.
- $h_n(k = (a,C), x \in \{0,1\}^n)$: let circuit $U_x(\cdot) = \mathcal{U}(x,\cdot)$, and output
$$G_n^{-1}\left[a + \mathsf{InertEval}(G_n, \mathsf{GSW\text{-}Eval}(U_x, C))\right] \in \{0,1\}^m. \qquad \diamondsuit$$

Remark. We do not use the term "fully homomorphic commitment" (which is implicitly given by sFHE schemes) in [PS19], since computational binding is not satisfied for that "commitment". However, the construction is essentially the same with the one in [PS19].

We first show this hash function family can be computed in TC^0, then show its correlation intractability. The latter property directly follows the proof of Theorem 3.4 in [PS19].

Lemma 4.15. Construction 3 can be computed by TC^0 circuits. $\diamondsuit$

Proof. Computing $\mathsf{Gen}_{\mathcal{H},n}$ and h_n only requires:

- Computing iterative sum mod $q = \mathsf{poly}(n)$; (In this construction, we only need to compute the product of a normal matrix (with each element in $\mathbb{Z}_q$) and a binary matrix (with each element in $\{0,1\}$), so it suffices to compute iterative sum in $\mathbb{Z}_q$.)
- Generating universal circuit U_x from x; (we only need to replace some input gates with binary values.)
- Computing G_n^{-1}, i.e. bit decomposition, which is trivial since we store a number in $\mathbb{Z}_q$ as a binary number.

All these tasks can be computed in TC^0. $\square$

Lemma 4.16. Assuming the hardness of $\mathsf{SIS}_{n,m,q,\beta}$ for a sufficiently large $\beta = m^{O(d)}$ against $\mathcal{C}$ adversary, Construction 3 is correlation intractable (cf. Theorem 4.14). $\diamondsuit$

Proof. Let $\mathcal{A} = \{\mathcal{A}_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ be any adversary that breaks correlation intractability against $\{f_n\}$, i.e.

$$\Pr\left[h_n(k = (a,C), x) = f_n(x) \;\middle|\; \begin{array}{l} A \leftarrow \mathbb{Z}_q^{n\times 2m} \\ C \leftarrow \mathsf{GSW\text{-}Enc}(A, 0^{S(n)}) \\ a \leftarrow \mathbb{Z}_q^n \\ x \leftarrow \mathcal{A}_n(k) \end{array} \right]$$

is non-negligible.

First, we change $C \leftarrow \mathsf{GSW\text{-}Enc}(A, 0^{S(n)})$ to $C \leftarrow \mathsf{GSW\text{-}Enc}(A, f)$ for $f = f_n$ by hybrid argument. It can be done because the distribution of C is statistically indistinguishable by the following lemma.

Lemma 4.17 (Item 1 of Proposition 2.10 in [PS19]). For any $x \in \mathbb{Z}_q^n$, the statistical distance between the distribution of $(A \leftarrow \mathbb{Z}_q^{n\times 2m}, C = \mathsf{GSW\text{-}Enc}(A, x))$ and uniformly random is in $\mathsf{negl}(m)$ by the leftover hash lemma. $\diamondsuit$

Therefore,

$$\Pr\left[h_n(k = (a, C), x) = f_n(x) \;\middle|\; \begin{array}{l} A \leftarrow \mathbb{Z}_q^{n\times 2m} \\ C \leftarrow \mathsf{GSW\text{-}Enc}(A, f_n) \\ a \leftarrow \mathbb{Z}_q^n \\ x \leftarrow \mathcal{A}_n(k) \end{array} \right]$$

is also non-negligible. Then we can use $\mathcal{A}$ to break SIS. Since $h_n(k, x) = f_n(x)$ implies that

$$\begin{aligned} G_n \cdot f_n(x) =& G_n \cdot h_n(k, x) \\ =& a + \mathsf{InertEval}(G_n, \mathsf{GSW\text{-}Eval}(U_x, C)) \\ =& a + (Ar + G_n \cdot f_n(x)) \\ =& A'z + G_n \cdot f_n(x), \end{aligned}$$

where

$$A' = \begin{bmatrix} a & A \end{bmatrix} \in \mathbb{Z}_q^{n\times(m+1)}, z = (1, r) \in \mathbb{Z}^{m+1}.$$

Note that $||z|| = m^{O(d)}$ as U_x is done by universally computing depth-d circuits.

So our SIS attacker works as follows: given an SIS instance A', the attacker takes A as the last m columns of A' and generates $C \leftarrow \mathsf{GSW\text{-}Enc}(A, f)$ with randomness R retained. After taking $x \leftarrow \mathcal{A}_n(k)$, it simulates the computation of $h_n(k, x)$ to get r, then outputs $z = (1, r)$. The attacker is in $\mathcal{C}$ since the reduction is in TC^0. $\square$

4.3 CI from worst-case complexity assumptions

In this subsection, we bypass the reliance on the lattice assumptions, which therefore also allows the hash function to be computable in $\mathsf{AC}^0[2]$, by applying techniques from [PS19] to WP-FHE (see Construction 2) instead of GSW-FHE (see Construction 1). This approach provides the construction of a hash function family that is computable in $\mathsf{AC}^0[2]$ and correlation-intractable for any relation searchable in constant-degree polynomials (with bounded circuit size) against any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, assuming $\oplus\mathsf{L/poly} \not\subseteq \widetilde{\mathsf{Sum}}_{n^{-c}} \circ \mathcal{C}$ for some $c > 0$. The result can be stated as follows:

Theorem 4.18. For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, if Assumption 4.8 holds for $\mathcal{C}$, for any polynomial $S(n)$ and constant d, there exists a hash family computable in $\mathsf{AC}^0[2]$ that is CI for any relation class searchable by degree-d polynomials $\{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ computable by circuits of size $S(n)$, against adversaries in $\mathcal{C}$ (following the "infinitely many" version of security). $\diamondsuit$

4.3.1 Adaptation of inert commitment to $\mathbb{Z}_2$

In our construction, all computation are performed in $\mathbb{Z}_2$, hence we restate the inert commitment in $\mathbb{Z}_2$, i.e, set $q = 2$. Although it is just the special case for $q = 2$, we emphasize that in this case, G is simplified to the identify matrix (as what we observed in WP-FHE), hence the inert commitment is greatly simplified.

Definition 4.19 (Inert Commitment in $\mathbb{Z}_2$). Given $C = \mathsf{WP\text{-}Enc}(A, x) \in \mathbb{Z}_2^{n \times n^2}$ as the homomorphic encryption of some length-n vector, and $M \in \mathbb{Z}_2^{n \times n}$ as a matrix, we write

$$\mathsf{InertEval}(M, C) := c_M = C \cdot v_M,$$

where $v_M \in \mathbb{Z}_2^{n^2}$ is the vector satisfies $(x^T \otimes I_n) \cdot v_M = Mx$ for any length-n vector x, obtained by reordering the elements of M. $\diamondsuit$

4.3.2 Construction

Here we present the construction of the CI hash family to prove Theorem 4.18.

Construction 4. The hash family $\mathcal{H} = \{h_n\}_{n \in \mathbb{N}}$ with the key generation algorithm $\mathsf{Gen}_{\mathcal{H}} = \{\mathsf{Gen}_{\mathcal{H},n}\}_{n \in \mathbb{N}}$ is parameterized by an arbitrary circuit size $S(n) = \mathsf{poly}(n)$ and depth d. Let $U(C, x) = C(x)$ denote a universal circuit for size-S, degree-d polynomials.

- $\mathsf{Gen}_{\mathcal{H},n}$: generate $A \leftarrow \mathsf{OneSamp}$ and $C \leftarrow \mathsf{WP\text{-}Enc}(A, 0^{S(n)})$, choose a uniformly random $a \leftarrow \mathbb{Z}_2^n$ and output the hash key $k = (a, C)$.
- $h_n(k = (a, C), x \in \{0,1\}^n)$: define the circuit $U_x(\cdot)$ as $U_x(\cdot) = U(x, \cdot)$ and output

$$a + \mathsf{InertEval}(I_n, \mathsf{WP\text{-}Eval}(U_x, C)) \in \{0,1\}^n\,. \qquad \diamondsuit$$

Proof of Theorem 4.18. We prove that for any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, Construction 4 is a CI hash family against $\mathcal{C}$ if Assumption 4.8 holds for $\mathcal{C}$. Let $\mathcal{A} = \{\mathcal{A}_n\} \in \mathcal{C}$ be any adversary family that breaks CI for some sequence of polynomials $\{f_n\}$. Since given x, we can check whether $h_n((a, C), x) = f_n(x)$ in $\mathcal{C}$ (which only involves matrix multiplication and computing $f_n(x)$), w.l.o.g., we assume that $\mathcal{A}_n$ always outputs some $x \in \{0,1\}^n$ s.t. $h_n((a, C), x) = f_n(x)$ with non-negligible probability or outputs $\perp$ after taking $k = (a, C)$ sampled according to $\mathsf{Gen}_{\mathcal{H},n}$ as input.

We abuse the notation to let $f_n \in \{0,1\}^{S(n)}$ also denote the encoding of the circuit that computes this function. The idea is to apply the hybrid argument to change the input distribution of $\mathcal{A}_n$. We observe that the following two distribution are equivalent, since A (sampled from $\mathsf{OneSamp}$) is always full-rank. Hence the two distributions are indistinguishable for $\mathcal{C}$ circuits.

- $\mathcal{D}_n^0 = \{(a, \mathsf{WP\text{-}Enc}(A, 0^{S(n)}) = AR + 0^{S(n)} \otimes I_n) : a \leftarrow \mathbb{Z}_2^n, A \leftarrow \mathsf{OneSamp}, R \leftarrow \mathbb{Z}_2^{n \times nS(n)}\}$.
- $\mathcal{D}_n^1 = \{(a, \mathsf{WP\text{-}Enc}(A, f_n) = AR + f_n \otimes I_n) : a \leftarrow \mathbb{Z}_2^n, A \leftarrow \mathsf{OneSamp}, R \leftarrow \mathbb{Z}_2^{n \times nS(n)}\}$.

Lemma 4.20. For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, if Assumption 4.8 holds for $\mathcal{C}$, any circuit family $\{\mathcal{A}_n\} \in \mathcal{C}$ cannot distinguish $\mathcal{D}_n^0$ and $\mathcal{D}_n^1$, i.e,

$$\left| \Pr\left[\mathcal{A}_n(k) = 1 \;\middle|\; k \leftarrow \mathcal{D}_n^0\right] - \Pr\left[\mathcal{A}_n(k) = 1 \;\middle|\; k \leftarrow \mathcal{D}_n^1\right] \right| < \mathsf{negl}(n). \qquad \diamondsuit$$

So after switching the input to $\mathcal{A}_n$ from $k \leftarrow \mathcal{D}_n^0$ to $k' \leftarrow \mathcal{D}_n^1$, $\mathcal{A}_n$ should also output $x \in \{0,1\}^n$ s.t. $h_n(k' = (a, \mathsf{WP\text{-}Enc}(A, f_n)), x) = f_n(x)$ with non-negligible probability. Since

$$\begin{aligned} f_n(x) =& h_n(k', x) \\ =& a + \mathsf{InertEval}(I_n, \mathsf{WP\text{-}Eval}(U_x, \mathsf{WP\text{-}Enc}(A, f_n))) \end{aligned}$$

$$=a + \mathsf{InertEval}(I_n, AR + f_n(x)^t \otimes I_n)$$
$$=a + Ar_{I_n} + f_n(x),$$

we have $Ar_{I_n} = a$ (in $\mathbb{Z}_2$). So if we record the randomness used to compute $\mathsf{WP\text{-}Enc}(A, f_n)$, by calculating $h_n(k', x)$, we are able to compute $r_{I_n} = A^{-1}a$. However, this allows us to invert $p_A(x) = Ax$, which is impossible by Lemma 4.13. □

5 Impossibility Result for Universal Fiat-Shamir Hash Functions

In this section, we show how we prove the impossibility result for the universal Fiat-Shamir hash function. We show that for any function family $\mathcal{H}$, we can construct an argument whose soundness cannot be preserved after applying the Fiat-Shamir transformation with $\mathcal{H}$.

Definition 5.1 ((l_x, l_a, l_b)-universal Fiat-Shamir hash from $\mathcal{C}$ to $\mathcal{D}$). Given circuit classes $\mathcal{D} \subseteq \mathcal{C}$ and functions $l_x(n)$, $l_a(n)$, $l_b(n)$, we say a Fiat-Shamir hash function family

$$\mathcal{H} = \left\{ \mathcal{H}_n : \{0,1\}^{l_x(n)+l_a(n)} \to \{0,1\}^{l_b(n)} \right\}_{n \in \mathbb{N}}$$

is an (l_x, l_a, l_b)-universal Fiat-Shamir hash from $\mathcal{C}$ to $\mathcal{D}$ if for every 3-round public-coin $\mathcal{C}$-argument with first message length $l_a(n)$ and second message length $l_b(n)$ that proves instances of length $l_x(n)$, applying Fiat-Shamir with $\mathcal{H}$ always gives a sound non-interactive $\mathcal{D}$-argument. ♢

Theorem 5.2. For circuit classes $\mathcal{D} \subseteq \mathcal{C}$, assume there exists collision-resistant hash function family computable in $\mathcal{D}$ and collision resistant against $\mathcal{C}$ with arbitrary polynomial compression. Then for any functions $l_a(n), l_b(n) \in \mathsf{poly}(n)$ such that $l_x(n), l_a(n) \in \omega(n^{\varepsilon})$ for some $\varepsilon > 0$ and $l_b(n) \in \omega(\log(n))$, there is no (l_x, l_a, l_b)-universal Fiat-Shamir hash from $\mathcal{C}$ to $\mathcal{D}$ that is computable in $\mathcal{C}$. ♢

The proof of Theorem 5.2 is done by showing that for any hash family $\mathcal{H}$, there is an interactive $\mathcal{C}$-argument for the empty language such that an adversary in $\mathcal{D}$ can break the reduced protocol after Fiat-Shamir with $\mathcal{H}$. Note that the empty language here can be generalized to any language $\mathcal{L}$ (except some corner cases that the soundness is trivial, like $\mathcal{L}$ does not have a $\mathcal{C}$-argument or there is no $x \notin \mathcal{L}$). This can be achieved by modifying the verifier of the interactive $\mathcal{C}$-argument for $\mathcal{L}$ to additionally accept the view that convinces the verifier of the empty language's protocol.

Here we first give the intuition before stating the proof. For the empty language, there exists a trivial protocol that the verifier just rejects anything. But to make the protocol insecure after the Fiat-Shamir transformation, we will use a verifier that accepts with negligible probability. Intuitively, the verifier will accept the view which convinces him that the prover can predict the second message before it is revealed. Here, the prover can convince the verifier by showing that he knows a circuit that can output the second message if it takes the first message as the input before the second message is revealed. As the second message is chosen at random, the prover cannot predict it during the real interaction. So, with this verifier, the argument is still secure.

After applying the Fiat-Shamir transformation, the Fiat-Shamir hash function can be used by the malicious prover as the circuit to predict the second message.

However, there exists a little problem. There is a limitation on the length of the first message, so the prover cannot send the circuit before the second message is revealed if it is too long. To bypass this problem, we allow the prover to first send a commitment of the circuit and reveal the circuit in the third message.

This idea is similar to the first idea of [GK03], which lets the verifier accept the view which convinces him the prover can predict the verifier's next message. But our idea is much simpler because currently the circuit size is fixed before we decide the protocol.

Proof of theorem 5.2. In the whole proof, all encoding of circuits or circuit sizes are considered under circuit class $\mathcal{C}$.

For any $\mathcal{D} \subseteq \mathcal{C}$, for any $l_x(n), l_a(n), l_b(n) \in \mathsf{poly}(n)$, consider any hash function family $\mathcal{H} = \mathcal{H}_n : \{0,1\}^{l_x(n)+l_a(n)} \to \{0,1\}^{l_b(n)}{}_{n\in\mathbb{N}}$ computable in $\mathcal{C}$. From the assumptions, we know there exists a collision resistant hash function family

$$\mathcal{F} = \left\{ \mathcal{F}_n : \{0,1\}^{l_{\mathcal{H}}(n)} \to \{0,1\}^{l_a(n)} \right\}_{n\in\mathbb{N}}.$$

Here we will construct an argument (P, V) for the empty language that applying Fiat-Shamir with $\mathcal{H}$ cannot preserve soundness (and thus $\mathcal{H}$ is not a universal Fiat-Shamir hash):

- Common input: $x \in \{0,1\}^{l_x(n)}$.
- Auxiliary input to the prover (or witness): none

1. V: Compute $f \leftarrow \mathcal{F}_n$ Output $crs = f$.
2. P: On input (x, w, crs):
   (a) Parse crs as f.
   (b) Let $\hat{C}$ be an arbitrary string $\in \{0,1\}^{l_{\mathcal{H}}(n)}$, which denotes a circuit.
   (c) Compute $a = f(\hat{C})$ and output it.
3. V: On input (x, crs, a), get public coin $b \leftarrow \{0,1\}^{l_b(n)}$ and output it.
4. P: On input (x, w, crs, a, b), output $c = \hat{C}$.
5. V: On input (x, crs, a, b, c):
   (a) Parse crs as f.
   (b) Parse c as $\hat{C} \in \{0,1\}^{l_{\mathcal{H}}(n)}$.
   (c) Check whether $a = f(\hat{C})$, reject if it is not the case.
   (d) Accept if $\hat{C}$ is a valid encoding with input size $l_a(n)$ and output size $l_b(n)$ and $C(a) = b$. Here, the evaluation of C is calculated by a universal circuit in $\mathcal{C}$, which is capable of computing any $h \in \mathcal{H}_n$.
   (e) Otherwise, reject.

The lengths of four messages are $l_{\mathcal{F}}(n), l_a(n), l_b(n), l_{\mathcal{H}}(n)$ respectively, which satisfy the requirement of applying Fiat-Shamir transformation with $\mathcal{H}$. Let the protocol obtained by applying Fiat-Shamir transformation be (P^{FS}, V^{FS}).

Now, we should examine the security before applying Fiat-Shamir transformation and the insecurity after it. We prove the security by constructing an adversary for the CRHF $\mathcal{F}$.

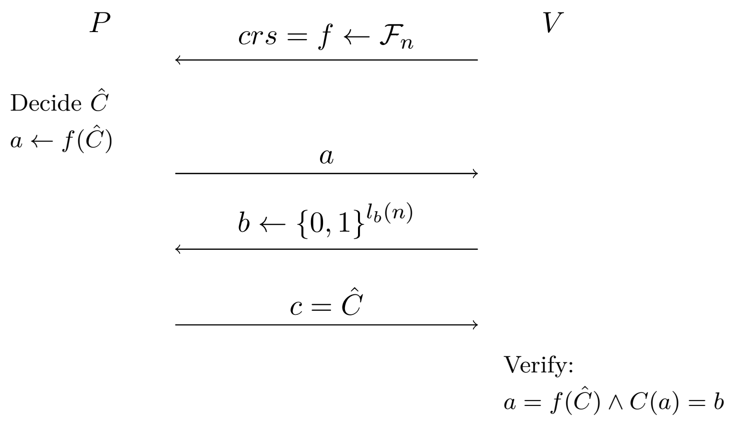

P
V
$crs = f \leftarrow \mathcal{F}_n$
Decide $\hat{C}$
$a \leftarrow f(\hat{C})$
a
$b \leftarrow \{0,1\}^{l_b(n)}$
$c = \hat{C}$
Verify:
$a = f(\hat{C}) \wedge C(a) = b$

Figure 5: Argument (P, V)

Suppose $P^* = \{P_n^*\}_{n\in\mathbb{N}}$ is the adversary that breaks (P, V) with non-negligible probability. The construction of the adversary $\mathcal{A} = \{\mathcal{A}_n\}_{n\in\mathbb{N}}$ for $\mathcal{F}$ is described below:

On input f:

1. Compute $crs = f$ and $a \leftarrow P^*(x, crs)$, the first message of $(P^*, V)(x)$. Here, $x \in \{0,1\}^{l_x(n)}$ is the input that the interaction between P^* and V results in acceptance with the highest probability.
2. Compute $b_1 \leftarrow V(x, crs, a)$ and $b_2 \leftarrow V(x, crs, a)$, the second message of $(P^*, V)(x)$ with different random bits.
3. Compute $c_1 \leftarrow P^*(x, crs, a, b_1)$ and $c_2 \leftarrow P^*(x, crs, a, b_2)$, the third message of $(P^*, V)(x)$.
4. Output c_1 and c_2.

Clearly, the adversary is in $\mathcal{C}$, so we only need to show it can succeed with non-negligible probability.

Claim 5.3. Assume there exists polynomial $q(n)$, for infinite many n, $\exists x \in \{0,1\}^{l_x(n)}$,

$$\Pr[(P^*, V)(x) = 1] > 1/q(n),$$

then A can break $\mathcal{F}$ with non-negligible probability. Formally, there exists a polynomial $p(n)$, for infinite many $n \in \mathbb{N}$,

$$\Pr\left[f(x_1) = f(x_2) \wedge x_1 \neq x_2 \;\middle|\; \begin{array}{l} f \leftarrow \mathcal{F}_n \\ (x_1, x_2) \leftarrow A(f) \end{array}\right] > 1/p(n), \qquad \diamondsuit$$

Proof. The adversary A for breaking the collision resistance of f sets f as the crs in the protocol (P^*, V), then runs P^* on crs to get started. So, for infinite many $n \in \mathbb{N}$, with probability $1/2q(n)$ over the generation of crs and a in adversary A, we have

$$\Pr[(P^*, V(x)|_{crs})(x) = 1] > \frac{1}{2q(n)}. \qquad (1)$$

Here P^* will follow from the state after the generation of crs and a in adversary $\mathcal{A}$, which means P^*'s first message will be restricted to a.

Since b_1, b_2 are generated independently, we have for infinite many $n \in \mathbb{N}$, with probability $1/2q(n)$ over the generation of f and a,

$$\Pr\left[\begin{array}{l}(P^*, V|_{crs,b_1})(x) = 1\wedge \\ (P^*, V|_{crs,b_2})(x) = 1\wedge \\ b_1 \neq b_2\end{array} \middle| \; crs, b_1, b_2 \leftarrow A(f)\right] > \frac{1}{4q^2(n)} - \mathrm{negl}(n).$$

Same as was in Eqn. (1), here both P^* will follow from the state after the generation of crs and a in adversary $\mathcal{A}$, which means they will both reply a as the first message.

As a result, for infinite many $n \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l}V(x, crs, a, b_1, c_1) = 1\wedge \\ V(x, crs, a, b_2, c_2) = 1\wedge \\ b_1 \neq b_2\end{array} \middle| \begin{array}{l} f \leftarrow \mathcal{F}_n \\ crs, a, b_1, b_2, c_1, c_2 \leftarrow A(f)\end{array}\right] > \frac{1}{8q^3(n)} - \mathrm{negl}(n).$$

This indicates that for infinite many $n \in \mathbb{N}$,

$$\Pr\left[f(x_1) = f(x_2) \wedge x_1 \neq x_2 \middle| \begin{array}{l} f \leftarrow \mathcal{F}_n \\ (x_1, x_2) \leftarrow A(f)\end{array}\right] > \frac{1}{8q^3(n)} - \mathrm{negl}(n). \qquad \square$$

Next, we will prove the insecurity of the argument after applying Fiat-Shamir transformation by directly constructing an adversary. The construction of adversary $P^* = \{P_n^*\}_{n\in\mathbb{N}}$ for (P^{FS}, V^{FS}) is shown below:

On input $x, crs' = (k_h, crs)$:

1. Parse crs' as h, f, the Fiat-Shamir hash function and the CRHF for commitment.
2. Let $C = h(x\|\cdot)$, hard-coding x into h. Let $c = \hat{C}$. Note that length of $\hat{C}$ is bounded by $l_{\mathcal{H}}(n)$.
3. Compute $a \leftarrow f(c)$.
4. Output a, c.

Note that the adversary is in $\mathcal{D}$ because it does not need to evaluate h.

Claim 5.4. P^* can break (P^{FS}, V^{FS}) with probability 1. Formally, $\forall n \in \mathbb{N}, \forall x \in \{0,1\}^{l_x(n)}$,

$$\Pr[(P^*, V^{FS})(x) = 1] = 1. \qquad \diamondsuit$$

Proof. Following V_n^{FS}, parsing crs' as h, f, parsing c as $\hat{C}$, we can directly know that

- $a = f(c)$,
- $\hat{C}$ can be calculated by the universal circuit.
- $b = C(a) = h(x\|a)$,

which satisfies the requirement for V_n^{FS} to accept. $\square$

Till now, we prove the security before applying Fiat-Shamir transformation, and the insecurity after applying Fiat-Shamir transformation, which completes the proof of theorem 5.2. $\square$

Remark. Actually, the counter-example in our proof works even if we consider an arbitrary poly-size Fiat-Shamir hash that is not necessarily computable in $\mathcal{C}$. Recall that the prover after Fiat-Shamir does not need to send the second message $b = h(x\|a)$ since it is clear from h, x, and a. Therefore, the cheating prover after Fiat-Shamir can simply copy the description of the Fiat-Shamir hash from the CRS and evaluate the value of f on it – all are computable in $\mathcal{D}$.

6 Stronger Impossibility Results: Universal Counter-Examples for All Fiat-Shamir Hash Functions

In this section, we aim to prove stronger impossibility results by exchanging the order of quantifiers. That is, we want to find a protocol that applying the Fiat-Shamir transformation with any function ensemble cannot preserve the soundness, just like [GK03]. Following the main idea in the last section, we let the adversary show his ability to predict the second message before the verifier reveals it.

Theorem 6.1. For any circuit class $\mathcal{C}$ that $\mathsf{AC}^0[2] \subseteq \mathcal{C}$, if there exists collision resistant hash function family computable in $\mathcal{C}$ and collision resistant against $\mathcal{C}$ with arbitrary polynomial compression, then there exists a 3-round public coin protocol with CRS (P, V) such that for all function ensemble H computable in $\mathcal{C}$ the following conditions are satisfied:

- (P, V) is a 3-round public coin argument for some language $\mathcal{L}$, that is computable in $\mathcal{C}$ and secure against any adversary in $\mathcal{C}$.
- Applying the Fiat-Shamir transformation on (P, V) with $\mathcal{H}$ results in a protocol (P^{FS}, V^{FS}) that is insecure against some adversary in $\mathcal{C}$. $\diamondsuit$

Notations Before starting the proof, we will first clarify some notations or conventions that we will frequently use. Throughout the section, we assume $\mathcal{C}$ is fixed, $\mathcal{F} = \left\{\mathcal{F}_n : \{0,1\}^{n^{0.2}} \to \{0,1\}^{n^{0.1}}\right\}_{n\in\mathbb{N}}$ is a collision resistant hash function family used in a tree commitment, which will be introduced later, $\mathsf{Comm} = \left\{\mathsf{Comm}_n : \{0,1\}^{5n} \to \{0,1\}^{n}\right\}$ is a collision resistant hash function family which is used for another commitment. When we use Comm to commit to some x, we will concatenate x with the randomness $r \in \{0,1\}^n$ and then apply Comm on (x, r). If the input of Comm is shorter than $5n$, we assume it is padded with 0s. Moreover, we will use $\mathcal{G} = \{\mathcal{G}_n\}_{n\in\mathbb{N}}$, $\mathcal{H} = \{\mathcal{H}_n\}_{n\in\mathbb{N}}$ as potential Fiat-Shamir hash function families. Typically, we will use $\mathcal{H}$ as the Fiat-Shamir hash function for our whole protocol, while using $\mathcal{G}$ as the Fiat-Shamir hash function inside the construction. $\mathcal{G}_1, \mathcal{G}_2$ will be used if there are more than one Fiat-Shamir hash functions used in the construction. We require $\mathcal{H}, \mathcal{G}, \mathcal{G}_1, \mathcal{G}_2$ to be computable in $\mathcal{C}$ throughout the whole section. In the whole section, all encodings of circuits and circuit sizes are considered under circuit class $\mathcal{C}$.

Definition 6.2 (Tree Commitment). A tree-commitment to $x \in \{0,1\}^*$, with respect to the function $f : \{0,1\}^{n^{0.2}} \to \{0,1\}^{n^{0.1}}$, is defined as follows. Consider a $n^{0.1}$-branch tree of depth $\log_{n^{0.1}}(|x|)$, where each node has a label in $\{0,1\}^{n^{0.1}}$. The leaves are labeled by the bits of x ($n^{0.1}$ bits per leaf). Each internal node is labeled by the string obtained by first concatenating all labels of its children and applying f to it. More specifically, if the labels of its children is $a_1, a_2, \cdots, a_{n^{0.1}} \in \{0,1\}^{n^{0.1}}$ the label of this node is $f(a_1\|a_2\|\cdots\|a_{n^{0.1}})$. The tree commitment to x, with respect to f, is denoted by $\mathsf{TC}_f(x)$, and consists of the label of the root and the depth of the tree. For a specific bit, suppose the leaf that contains this bit is l. Define the authentication path for that bit to be the concatenation of all labels of nodes that lie between l and the root. Note that the number of bits of an authentication path is $\log_{n^{0.1}}(|x|)n^{0.1}$. For x of length n^c, the authentication path is of length $10cn^{0.1} = O(n^{0.1})$ bits. $\diamondsuit$

Remark. The tree commitment itself can be used as a CRHF, with arbitrary polynomial input size and $n^{0.1}$ output size, when f is sampled from the CRHF family. The advantage of using tree commitment instead of CRHF as commitment is that we can open the string locally on a bit by outputting its authentication path. Also note that the verification of an authentication path can be done in low depth regardless of the depth of the tree. The binding property is satisfied since if an adversary outputs something different from the committed one, a collision can always be found somewhere on the path.

Normally, tree commitment only needs $2n$ to n hash functions, but in low depth setting, this is not enough to achieve constant depth. Thus we need the stretch to be polynomial. Any CRHF with polynomial stretch can be used here, and the only reason for us to use $n^{0.2} \to n^{0.1}$ function is to make the size of authentication path fit in $O(n)$ and facilitate later analysis.

An important idea in this proof follows from the construction in the proof of Theorem 5.2. That is, we want to modify the verifier so that the verifier accepts the view that convinces him the prover knows his second message before it is revealed. In the previous construction, we let the adversary prove this by first sending a commitment of circuit C, and then reveal that C can output the second message b when taking the first message a as the input. This can be abstracted as a relation, which we denote as the *Central Relation*. The Central Relation characterizes the fact that the adversary wants to prove he has the witness for $f\|a\|b$ that satisfies certain requirements, where f is the function used for commitment and a, b is the first and second message respectively. In the construction of the proof of theorem 5.2, the adversary directly shows the witness to prove the knowledge, but we will use some succinct proof to achieve this goal later.

Definition 6.3 (Central Relation). For all $n \in \mathbb{N}$. Define

$$\mathcal{R} = \left\{ (f\|a\|b, \hat{C}\|\Pi) \;\middle|\; \begin{array}{l} \mathsf{TC}_f(\tilde{C}) = a \wedge C(a) = b \wedge |\tilde{C}| \le n^{\ln n} \\ \Pi \text{ is a computation history of } \mathsf{TC}_f(\tilde{C}), C(a) \end{array} \right\}$$

Here, $a, b \in \{0,1\}^n, f \in \mathcal{F}_n$. C is a circuit and $\hat{C}$ is the encoding for C with input size n and output size n regarding to $\mathcal{C}$. $\tilde{C}$ is a special encoding of C that satisfies the following requirements:

- Transformation between $\hat{C}$ and $\tilde{C}$ can be computed in $\mathcal{C}$.
- It has high minimum distance. That is for any different C_1, C_2, we require $\tilde{C}_1$ and $\tilde{C}_2$ differ in at least polynomial fraction of all bits. ♢

Remark. Although verifying $\mathsf{TC}_f(\tilde{C}) = a \wedge C(a) = b$ may be difficult as we do not limit the depth of C for now, we can verify computation history to check those equations. Thus, the central relation can be verified in constant depth. Also, the size of C is bounded by $n^{\ln n}$, which means the relation is $\mathsf{NTIME}(n^{\ln n})$ instead of NP, and we will address this problem with the PCP proof system that we will introduce later.

Now, we will introduce the main process of our proof. We construct three protocols for the empty language. We want to construct the protocol that the verifier only accepts some views that occur with negligible probability, but the adversary can make these views occur frequently after the Fiat-Shamir transform:

- The first protocol (P_1, V_1) follows the straightforward idea that implements a succinct 2-round protocol with CRS for central relation after f, a, b is determined in the first two rounds, which gives the adversary potential ability to show he knows the witness of $f\|a\|b$ in central relation.

As the adversary can use the Fiat-Shamir hash function as the witness and can compute the history Π itself, it can convince the verifier that he knows the witness, which implies the insecurity after applying Fiat-Shamir. However, we do not know if such a 2-round protocol satisfies the soundness we need, so we are not sure about the security.

- The second protocol (P_2, V_2) goes from a different way. After the verifier determines f, the adversary can determine a, b_1, b_2 that $b_1 \neq b_2$ and try to prove he knows the witness for both instances (f, a, b_1) and (f, a, b_2) in central relation, with a 3-round secure protocol with CRS. As a serves as the commitment of the witness, it is hard for the adversary to find two different Cs, i.e., the witness for both (f, a, b_1) and (f, a, b_2), which indicates its security. But for this protocol, the insecurity after applying Fiat-Shamir is hard to argue.
- Then, we find the third protocol that resides between the first one and the second one. We can show the third protocol is secure if the second one is secure after applying Fiat-Shamir, and we can show the third protocol after applying Fiat-Shamir is insecure if the first one is insecure on its own.

With the existence of the third protocol, we can show at least one of the three protocols can be used as a protocol required in the theorem.

The whole proof follows the idea in [GK03], but some technical problems need to be addressed:

- We need to fit all cryptographic tools into a low-depth circuit class. Some of them such as the commitment scheme used in [GK03] are not known in low-depth circuit class currently.
- In the tree commitment scheme, we require the inner hash function to have polynomial stretch, which is used to keep the depth constant.
- Since the circuit size in central relation is bounded by $n^{\ln n}$ instead of a polynomial, super-constant depth may still occur even with polynomial-stretch hash function. We resolve this by requiring the prover to give all computation history to facilitate the verification. Due to the fact that honest prover does not need to do anything, and malicious prover can have depth depending on the Fiat-Shamir hash function, we can eventually achieve constant depth.

To start, we first introduce the 2-round protocol that is used in the first protocol but not guaranteed to be secure, and the 3-round protocol used in the second protocol, as mentioned above. Here, we require the protocol to be succinct, thus we cannot directly send the circuit as the former construction. [Kil92] gives a 3-round argument with CRS for NEXP language that satisfies knowledge extraction, which can be used for proving our central relation. To make it fit in $\mathsf{AC}^0[2]$, we need to use a specific PCP system.

Definition 6.4 (Probabilistic verifier in $\mathsf{AC}^0[2]$). A probabilistic verifier for an NEXP language L is an $\mathsf{AC}^0[2]$ circuit V_{pcp} that is given a input x, and oracle access to a proof π which we will denote by V_{pcp}^{π}. We say that V_{pcp} has randomness complexity $r(n)$ and query complexity $q(n)$ with respect to polynomials $r(n)$ and $q(n)$ if for any input x such that $|x| = l_x(n)$,

1. $V_{pcp}^{\pi}(x)$ uses at most $r(n)$ random bits.
2. $V_{pcp}^{\pi}(x)$ queries at most $q(n)$ times to the oracle.

As a special note, V_{pcp} should only make non-adaptive queries, that is, there is an probabilistic polynomial time algorithm Q_{pcp} upon receiving the input x and the random tape γ for V_{pcp}, outputs queries $(\phi_1, \phi_2, \cdots, \phi_{q(n)})$. $\diamondsuit$

Theorem 6.5. For the central relation $\mathcal{R}$, there exists a PCP system (P_{pcp}, V_{pcp}) where V_{pcp} is a probabilistic verifier in $\mathsf{AC}^0[2]$ for $\mathcal{R}$ with randomness complexity $O(\log n)$ and query complexity $O(1)$. while P_{pcp} is an $\mathsf{AC}^0[2]$ circuit. This system satisfies:

- **Completeness**: For all $(x, w) \in \mathcal{R}$,

$$\Pr\left[V_{pcp}^{\pi}(x) = 1 \;\middle|\; \pi \leftarrow P_{pcp}(x, w)\right] = 1.$$

  This implies that there exists a polynomial $p(\cdot)$ such that the proof π generated is of length $p(|x| + |w|)$.

- **Soundness**: For all x such that there does not exists w that $(x, w) \in \mathcal{R}$, for any proof π^*,

$$\Pr\left[V_{pcp}^{\pi^*}(x) = 1\right] < \frac{1}{2},$$

  where the randomness is over the internal randomness of V_{pcp}.

- **Knowledge extraction**: there exists an extractor E_{pcp} in $\mathsf{AC}^0[2]$ such that for any input x and proof $\tilde{\pi}$ if

$$\Pr\left[V_{pcp}^{\tilde{\pi}}(x) = 1\right] \geq 1 - \frac{1}{\log^5 n},$$

  then

$$\Pr\left[(x, w) \in \mathcal{R} \mid w \leftarrow E_{pcp}(x, \tilde{\pi})\right] = 1. \qquad \diamondsuit$$

Theorem 6.5 essentially follows from the PCP construction in [Par21; CWY23], but we include a proof in Appendix C that the prover can be implemented in $\mathsf{AC}^0[2]$.

Recall that if we want to use a PCP system to prove the central relation we have to address the problem that the witness is of super-polynomial size. But the above PCP system guarantees that there exists a polynomial $p(\cdot)$ such that the proof π generated is of length $p(|x| + |w|)$. In other words, if the witness $\hat{C} \| \Pi$ in the central relation is of some fixed polynomial length (which is the case in later applications), the proof π is also fixed polynomial size.

Later on we consider running $n^{0.1}$ PCPs in parallel to make the probability of accepting a wrong statement to negligible. When we say V_{pcp} later, we refer to the verifier that runs $n^{0.1}$ PCPs in parallel and accepts if and only if all verifications are passed.

In later protocols we use PCP together with the tree commitment technique. We first let $\pi \leftarrow P_{pcp}$ be the proof, then use a tree commitment to compress the proof into a length n message $\beta = \mathsf{TC}_{f^{TC}}(\pi)$. Instead of outputting the queried bit later, we output the authentication path on tree commitment. Since the tree is of constant depth with $O(n^{0.2})$-length authentication path and we run $n^{0.1}$ PCP verification in parallel, all queries and answers can be expressed by strings $\gamma, \delta \in \{0,1\}^n$, respectively.

6.1 Interactive Argument for Central Relation

According to [Kil92], there is an argument (P_{Kil}, V_{Kil}) for central relation $\mathcal{R}$ based on (P_{pcp}, V_{pcp}) for $\mathcal{R}$. Recall $\mathcal{F}$ is the CRHF used for tree commitment. The protocol is shown below:

- Common input: $f \| a \| b$.

- Auxiliary input for the prover: $w = \hat{C} \| \Pi$.

1. V_{Kil} : Output $crs = f^{TC} \leftarrow \mathcal{F}_n$.
2. P_{Kil} : On input $(f\|a\|b, w, crs)$:
   (a) Compute PCP proof $\pi \leftarrow P_{pcp}(f\|a\|b, w)$.
   (b) Compute $\beta \leftarrow \mathsf{TC}_{f^{TC}}(\pi)$ and output β.
3. V_{Kil} : On input $(f\|a\|b, crs, \beta)$, select random tape γ for V_{pcp} and output γ.
4. P_{Kil}: On input $(f\|a\|b, w, crs, \beta, \gamma)$, simulate V_{pcp} on random tape γ, record its queries and let the authentication paths along with the answers to these queries be δ. Output δ.
5. V_{Kil}: On input $(f\|a\|b, crs, \beta, \gamma, \delta)$, accept iff:
   (a) The authentication paths in δ are consistent under the tree commitment scheme.
   (b) V_{pcp} accepts after receiving answers in δ.

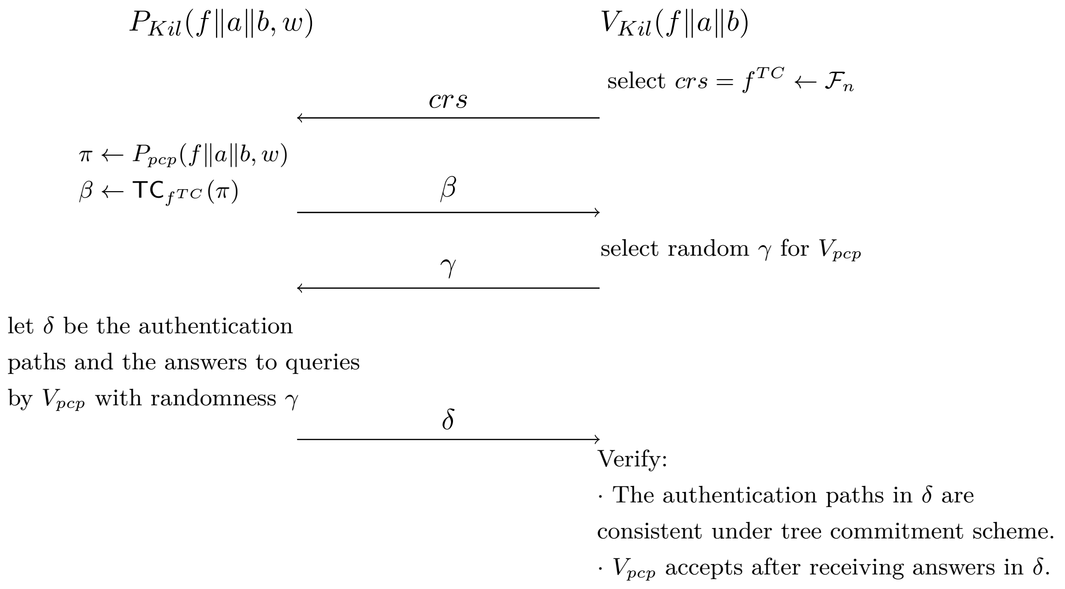

$P_{Kil}(f\|a\|b, w)$
$V_{Kil}(f\|a\|b)$
select $crs = f^{TC} \leftarrow \mathcal{F}_n$
crs
$\pi \leftarrow P_{pcp}(f\|a\|b, w)$
$\beta \leftarrow \mathsf{TC}_{f^{TC}}(\pi)$
β
select random γ for V_{pcp}
γ
let δ be the authentication paths and the answers to queries by V_{pcp} with randomness γ
δ
Verify:
· The authentication paths in δ are consistent under tree commitment scheme.
· V_{pcp} accepts after receiving answers in δ.

Figure 6: Interactive Argument for the Central Relation

Lemma 6.6. (P_{Kil}, V_{Kil}) is a 3-round argument in $\mathcal{C}$ with CRS for language $\mathcal{R}$ against $\mathcal{C}$, that satisfies knowledge extraction in $\mathsf{AC}^0[2]$. $\Diamond$

Proof. We can obtain completeness directly from the completeness of probabilistic verification. For soundness and knowledge extraction we prove them together below.

We try to prove that there exists an extractor $E^{\tilde{P}}$ such that for any prover $\tilde{P}$ and a polynomial $p(n)$, if for an $x \in \{0,1\}^{l_x(n)}$,

$$\Pr\left[(\tilde{P}, V)(x) = 1\right] \geq \frac{1}{p(n)}$$

then with probability $\frac{1}{2p(n)}$ over the generation of proof $\tilde{\pi} \leftarrow E^{\tilde{P}}$, the proof $\tilde{\pi}$ satisfies

$$\Pr[V_{pcp}^{\tilde{\pi}}(x) = 1] \geq \frac{\mathrm{e} - 2}{4\mathrm{e}} \cdot \frac{1}{(p(n))^2}.$$

Note that the probability here is the probability that $\tilde{\pi}$ passes all $n^{0.1}$ parallel verification. Since all verifications are independent, the probability that $\tilde{\pi}$ passes a single verification is greater than $1 - \frac{1}{\log^5 n}$. If such extractor exists, then we prove the knowledge extraction property. The extractor calls $E^{\tilde{P}}$ at first to obtain the proof $\tilde{\pi}$ and then calls the PCP extractor to obtain the witness $w \leftarrow E_{pcp}(\tilde{\pi})$ with probability 1. For soundness, if $\neg \exists w$ s.t. $(x, w) \in \mathcal{R}$, then there is no way to extract such w and thus no prover $\tilde{P}$ exists. To construct such extractor $E^{\tilde{P}}$, a key observation is that the distributions of queries $\xi = Q_{pcp}(f \| a \| b, \gamma)$ are the same when the prover $\tilde{P}$ interacts with the verifier V_{Kil} and when the PCP verifier queries its oracle. The proof idea is to repeatedly use $\tilde{P}$ and find out all positions in π that would be queried with considerable probability. $E^{\tilde{P}}$ on input $(f \| a \| b)$ works as follows:

1. Compute $crs \leftarrow V_{Kil}$.
2. Run $\tilde{P}(f \| a \| b, crs)$ for the first round to obtain β.
3. Start with $\tilde{\pi} = 0^{\ell}$ where ℓ is the length of PCP proof. Let S be a empty set and $j = k = 0$ be two counters. Repeat the following steps with different random bits:
4. Choose $\gamma \leftarrow \{0,1\}^n$.
5. Run $\delta \leftarrow \tilde{P}(f \| a \| b, crs, \beta, \gamma)$.
6. If $V_{Kil}(f \| a \| b, crs, \beta, \gamma, \delta) = 1$,
    (a) Increase the counter $k \leftarrow k + 1$, $j \leftarrow j + 1$.
    (b) Record the queried positions $\xi_k \leftarrow Q_{pcp}(f \| a \| b, \gamma)$.
    (c) Update $\tilde{\pi}$ to match δ on positions ξ_k. If this update contradict any previous update, abort and output 0^{ℓ}.
    (d) Update the set of positions that are updated $S \leftarrow S \cup \xi_k$.
    (e) Halt if $k = \ell$ and output $\tilde{\pi}$.
    (f) If $j > (\ell p(n))^2$, abort and output 0^{ℓ}.

This algorithm can be implemented in $\mathsf{AC}^0[2]$. We can run the loop simultaneously since they are independent.
Suppose that the prover $\tilde{P}$ convinces V_{Kil} on input $(f \| a \| b)$ with probability $\frac{1}{p(n)}$. Define the state after the first two steps to be good if

$$\Pr\left[V_{Kil}(f \| a \| b, crs, \beta, \gamma, \delta) = 1 \;\middle|\; \begin{array}{l} \gamma \leftarrow \{0,1\}^n \\ \delta \leftarrow \tilde{P}(f \| a \| b, crs, \beta, \gamma) \end{array} \right] \geq \frac{1}{2p(n)},$$

where crs, β are generated in the first two steps.

By Markov inequality, the probability that the state is good is at least $\frac{1}{2p(n)}$. Note that if the state is good then the abort probability of $E^{\tilde{P}}$ is negligible since:

- We find a collision for the tree commitment when we abort in 6(c).
- We abort in 6(f) with negligible probability by Chernoff bound.

Also, if the state is good then we have (ignore some negligible probability where $E^{\tilde{P}}$ aborts),

$$\begin{aligned}
&\Pr\left[V_{pcp}^{\tilde{\pi}}(f\|a\|b)=1\right]\\
\geq &\Pr\left[Q_{pcp}(f\|a\|b,\gamma)\subseteq S \,\middle|\, \begin{array}{l}\gamma\leftarrow\{0,1\}^n\\ \delta\leftarrow\tilde{P}(f\|a\|b,crs,\beta,\gamma)\\ V_{Kil}(f\|a\|b,crs,\beta,\gamma,\delta)=1\end{array}\right]\\
\cdot &\Pr\left[V_{Kil}(f\|a\|b,crs,\beta,\gamma,\delta)=1 \,\middle|\, \begin{array}{l}\gamma\leftarrow\{0,1\}^n\\ \delta\leftarrow\tilde{P}(f\|a\|b,crs,\beta,\gamma)\end{array}\right]\\
\geq &\frac{1}{2p(n)}\cdot\Pr\left[Q_{pcp}(f\|a\|b,\gamma)\subseteq S \,\middle|\, \begin{array}{l}\gamma\leftarrow\{0,1\}^n\\ \delta\leftarrow\tilde{P}(f\|a\|b,crs,\beta,\gamma)\\ V_{Kil}(f\|a\|b,crs,\beta,\gamma,\delta)=1\end{array}\right]\\
\geq &\frac{1}{2p(n)}\cdot\left(1-\sum_{i=1}^{\ell}\prod_{k=1}^{\ell}\Pr\left[i\notin\xi_k\right]\cdot\Pr\left[i\in Q_{pcp}(f\|a\|b,\gamma) \,\middle|\, \begin{array}{l}\gamma\leftarrow\{0,1\}^n\\ \delta\leftarrow\tilde{P}(f\|a\|b,crs,\beta,\gamma)\\ V_{Kil}(f\|a\|b,crs,\beta,\gamma,\delta)=1\end{array}\right]\right)
\end{aligned}$$

The first inequality is by condition probability. Plug in the result that *state* is good we get the second inequality. The last inequality is by union bound. Since we independently sample γ each round in $E^{\tilde{P}}$, the probabilities that the i-th bit is queried in each round and by $V_{pcp}^{\tilde{\pi}}$ at last are the same. That is,

$$\Pr\left[i\in\xi_k\right]=\Pr\left[i\in Q_{pcp}(f\|a\|b,\gamma) \,\middle|\, \begin{array}{l}\gamma\leftarrow\{0,1\}^n\\ \delta\leftarrow\tilde{P}(f\|a\|b,crs,\beta,\gamma)\\ V_{Kil}(f\|a\|b,crs,\beta,\gamma,\delta)=1\end{array}\right].$$

Then we have,

$$\begin{aligned}
&\Pr\left[V_{pcp}^{\tilde{\pi}}(f\|a\|b)=1\right]\\
\geq &\frac{1}{2p(n)}\cdot\left(1-\sum_{i=1}^{\ell}\prod_{k=1}^{\ell}\Pr\left[i\notin\xi_k\right]\cdot\Pr\left[i\in Q_{pcp}(f\|a\|b,\gamma) \,\middle|\, \begin{array}{l}\gamma\leftarrow\{0,1\}^n\\ \delta\leftarrow\tilde{P}(f\|a\|b,crs,\beta,\gamma)\\ V_{Kil}(f\|a\|b,crs,\beta,\gamma,\delta)=1\end{array}\right]\right)\\
\geq &\frac{1}{2p(n)}\cdot\left(1-\ell\cdot\frac{1}{\mathrm{e}\ell}\right)=\frac{\mathrm{e}-1}{2\mathrm{e}}\cdot\frac{1}{p(n)}. \quad \text{(e is the natural constant)}
\end{aligned}$$

Overall, we have

$$\Pr\left[V_{pcp}^{\tilde{\pi}}(x)=1 \,\middle|\, \tilde{\pi}\leftarrow E^{\tilde{P}}\right]\geq\frac{\mathrm{e}-2}{4\mathrm{e}}\cdot\frac{1}{(p(n))^2}.$$ □

6.2 Reduced Argument for Central Relation

Now we introduce a reduced argument (P_{Mic}, V_{Mic}) for central relation that is similar to the previous argument. This is a 2-round protocol with CRS. One can imagine it to be some kind of compressed version of the previous argument by Fiat-Shamir-like transformation but there are some nuances:

- Two Fiat-Shamir transformations may be applied on the original protocol in parallel. That means, in the reduced protocol, the prover will generate the verifier's public coin message twice with two different Fiat-Shamir hash functions, and need to pass the verification for both messages.

- Instead of sending β in the first round, we send the commitment of β, denoted as $\hat{\beta}$. This is employed to address some details in later proofs, especially the insecurity of $({P_2}^{FS}, {V_2}^{FS}), ({P_3}^{FS}, {V_3}^{FS})$, which will be introduced later.

- We separate the message from verifier as the CRS and the first message. The CRS can be generated by a private coin generator, and the first message must be generated with public coins. This distinction is made to facilitate the integration of the reduced protocol into later 3-round protocols. Specifically, our aim is to incorporate the CRS from this protocol into the CRS of the 3-round protocol, while placing the first message from this protocol into the second message (the one sent by the verifier) in the 3-round protocol.

Let $\mathcal{F}, \mathsf{Comm}$ be two CRHFs as mentioned in the beginning of this section. We will give two versions of this argument (P^1_{Mic}, V^1_{Mic}) and (P^2_{Mic}, V^2_{Mic}). (P^1_{Mic}, V^1_{Mic}) only uses one Fiat-Shamir hash function, while (P^2_{Mic}, V^2_{Mic}) uses two.
For the first version, let $\mathcal{G}$ be a Fiat-Shamir hash function which will be clear in the content when we analyze or utilize the protocol. (P^1_{Mic}, V^1_{Mic}) works as follows:

- Common input: $f\|a\|b$.

- Auxiliary input for the prover: $w = \hat{C}\|\Pi$.

1. V^1_{Mic}: Select $f^{TC} \leftarrow \mathcal{F}_n$, $\mathsf{comm} \leftarrow \mathsf{Comm}_n$, $g \leftarrow \mathcal{G}$ and output $crs = (f^{TC}, g, \mathsf{comm})$. Later, we will denote this process as $crs \leftarrow \mathsf{Gen}^{\mathcal{G}}_{Mic}$.
2. V^1_{Mic}: Select the randomness for comm, $r_c \leftarrow \{0,1\}^n$ and output r_c.
3. P^1_{Mic}: On input $(f\|a\|b, w, crs, r_c)$:
   (a) Compute PCP proof $\pi \leftarrow P_{pcp}(f\|a\|b, w)$.
   (b) Compute $\beta \leftarrow \mathsf{TC}_{f^{TC}}(\pi)$.
   (c) Compute $\hat{\beta} \leftarrow \mathsf{comm}(\beta\|r_c)$.
   (d) Compute $\gamma \leftarrow g(\hat{\beta})$.
   (e) Simulate V_{pcp} on γ and let the authentication paths along with answers to queries from V_{pcp} be δ.
   (f) Output $ans = (\beta, \hat{\beta}, \gamma, \delta)$.
4. V^1_{Mic}: On input $(f\|a\|b, crs, ans)$, accept iff:
   (a) $\hat{\beta} = \mathsf{comm}(\beta\|r_c)$.
   (b) $\gamma = g(\hat{\beta})$.
   (c) V_{pcp} accept with random tape γ and query responses δ.

For the second version, let $\mathcal{G}_1, \mathcal{G}_2$ be two Fiat-Shamir hash functions which will be clear in the content. (P^2_{Mic}, V^2_{Mic}) works as follows:

- Common input: $f\|a\|b$.

- Auxiliary input for the prover: $w = \hat{C}\|\Pi$.

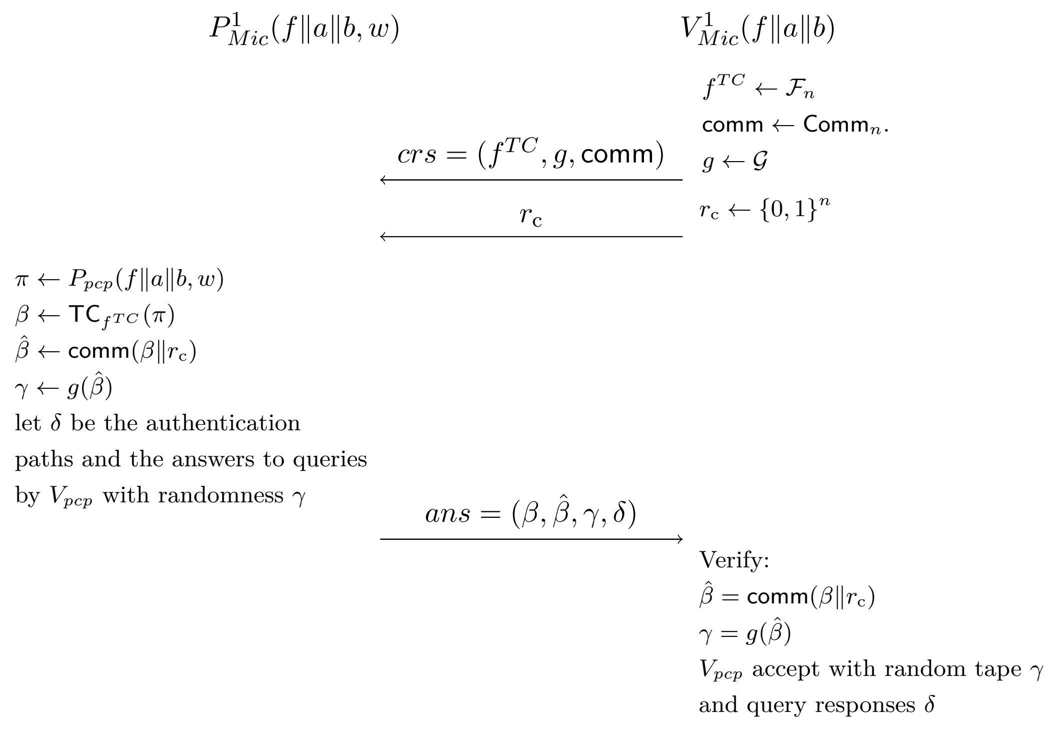
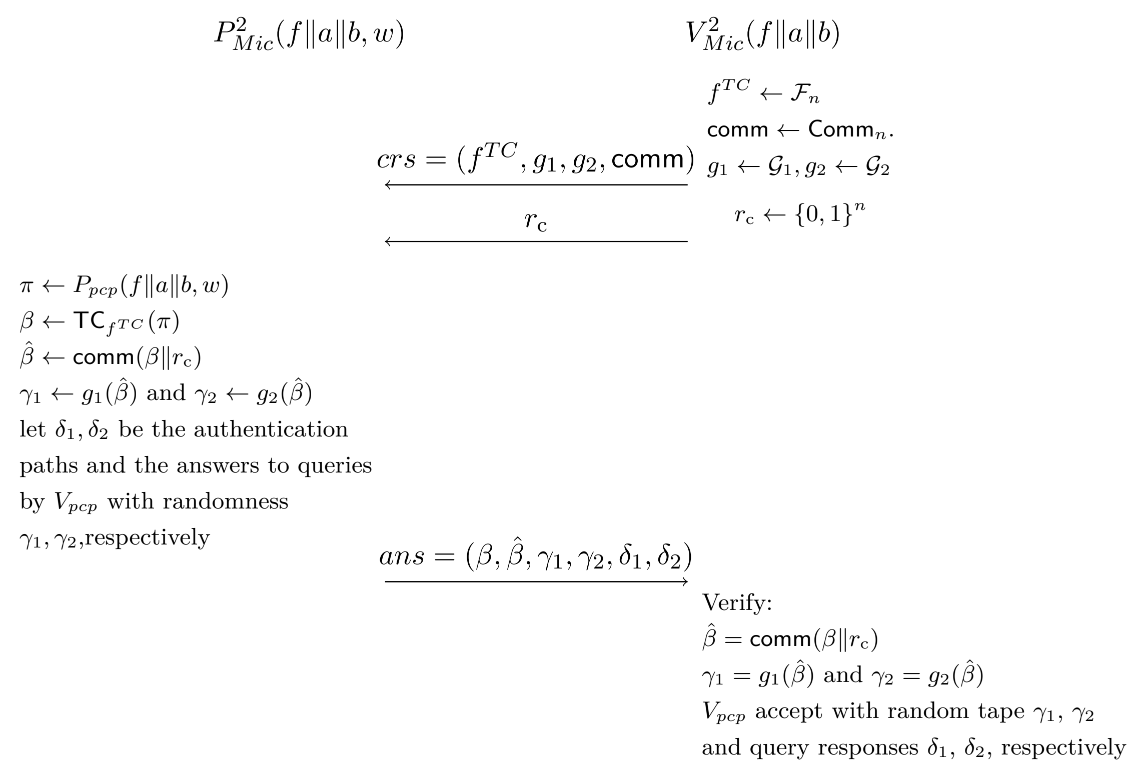

$P^1_{Mic}(f\|a\|b, w)$
$V^1_{Mic}(f\|a\|b)$
$f^{TC} \leftarrow \mathcal{F}_n$
$\mathsf{comm} \leftarrow \mathsf{Comm}_n.$
$g \leftarrow \mathcal{G}$
$crs = (f^{TC}, g, \mathsf{comm})$
$r_c \leftarrow \{0,1\}^n$
r_c
$\pi \leftarrow P_{pcp}(f\|a\|b, w)$
$\beta \leftarrow \mathsf{TC}_{f^{TC}}(\pi)$
$\hat{\beta} \leftarrow \mathsf{comm}(\beta\|r_c)$
$\gamma \leftarrow g(\hat{\beta})$
let δ be the authentication paths and the answers to queries by V_{pcp} with randomness γ
$ans = (\beta, \hat{\beta}, \gamma, \delta)$
Verify:
$\hat{\beta} = \mathsf{comm}(\beta\|r_c)$
$\gamma = g(\hat{\beta})$
V_{pcp} accept with random tape γ and query responses δ

Figure 7: Reduced Argument for the Central Relation (P^1_{Mic}, V^1_{Mic})

1. V^2_{Mic}: Select $f^{TC} \leftarrow \mathcal{F}_n$, $\mathsf{comm} \leftarrow \mathsf{Comm}_n$, $g_1 \leftarrow \mathcal{G}_1$, $g_2 \leftarrow \mathcal{G}_2$ and output $crs = (f^{TC}, g_1, g_2, \mathsf{comm})$. Later, we will denote this process as $crs \leftarrow \mathsf{Gen}^{\mathcal{G}_1,\mathcal{G}_2}_{Mic}$.
2. V^2_{Mic}: Select the randomness for comm, $r_c \leftarrow \{0,1\}^n$ and output r_c.
3. P^2_{Mic}: On input $(f\|a\|b, w, crs, r_c)$:
   (a) Compute PCP proof $\pi \leftarrow P_{pcp}(f\|a\|b, w)$.
   (b) Compute $\beta \leftarrow \mathsf{TC}_{f^{TC}}(\pi)$.
   (c) Compute $\hat{\beta} \leftarrow \mathsf{comm}(\beta\|r_c)$.
   (d) Compute $\gamma_1 \leftarrow g_1(\hat{\beta})$ and $\gamma_2 \leftarrow g_2(\hat{\beta})$.
   (e) Simulate V_{pcp} on γ_1, γ_2 and let the authentication paths along with answers to queries from V_{pcp} be δ_1, δ_2, respectively.
   (f) Output $ans = (\beta, \hat{\beta}, \gamma_1, \gamma_2, \delta_1, \delta_2)$.
4. V^2_{Mic}: On input $(f\|a\|b, crs, ans)$, accept iff:
   (a) $\hat{\beta} = \mathsf{comm}(\beta\|r_c)$.
   (b) $\gamma_1 = g_1(\hat{\beta})$ and $\gamma_2 = g_2(\hat{\beta})$.
   (c) V_{pcp} accept with random tape γ_1, γ_2 and query responses δ_1, δ_2, respectively.

Note that those two reduced protocols maintains the completeness of the previous protocol (P_{Kil}, V_{Kil}), but there is no guarantee about the soundness. Our following proof will consider both cases whether the protocol has soundness.


$P^2_{Mic}(f\|a\|b, w)$ $\quad$ $V^2_{Mic}(f\|a\|b)$

V^2_{Mic}: $f^{TC} \leftarrow \mathcal{F}_n$; $\mathsf{comm} \leftarrow \mathsf{Comm}_n$. $g_1 \leftarrow \mathcal{G}_1, g_2 \leftarrow \mathcal{G}_2$

$$\xleftarrow{crs = (f^{TC}, g_1, g_2, \mathsf{comm})}$$

$r_c \leftarrow \{0,1\}^n$

$$\xleftarrow{r_c}$$

P^2_{Mic}:
$\pi \leftarrow P_{pcp}(f\|a\|b, w)$
$\beta \leftarrow \mathsf{TC}_{f^{TC}}(\pi)$
$\hat{\beta} \leftarrow \mathsf{comm}(\beta\|r_c)$
$\gamma_1 \leftarrow g_1(\hat{\beta})$ and $\gamma_2 \leftarrow g_2(\hat{\beta})$
let δ_1, δ_2 be the authentication paths and the answers to queries by V_{pcp} with randomness γ_1, γ_2,respectively

$$\xrightarrow{ans = (\beta, \hat{\beta}, \gamma_1, \gamma_2, \delta_1, \delta_2)}$$

Verify:
$\hat{\beta} = \mathsf{comm}(\beta\|r_c)$
$\gamma_1 = g_1(\hat{\beta})$ and $\gamma_2 = g_2(\hat{\beta})$
V_{pcp} accept with random tape γ_1, γ_2 and query responses δ_1, δ_2, respectively

Figure 8: Reduced Argument for the Central Relation (P^2_{Mic}, V^2_{Mic})

6.3 Protocol 1

In the first protocol (P_1, V_1), the reduced protocol is applied to enable the prediction of the next message of the verifier after the Fiat Shamir transformation. In the CRS we generate the function f for central relation, then we let the prover send a and the verifier send b, to determine the instance of central relation. After the instance is determined, we invoke (P^2_{Mic}, V^2_{Mic}) to prove it. Note that although interaction of (P^2_{Mic}, V^2_{Mic}) starts after f, a, b is determined, the CRS of (P^2_{Mic}, V^2_{Mic}) should be incorporated into the CRS of (P_1, V_1), because it may be generated with private coins. Since the adversary can choose a after he learns the CRS of (P^2_{Mic}, V^2_{Mic}), it seems like we need (P^2_{Mic}, V^2_{Mic}) to satisfy some strong soundness to keep (P_1, V_1) itself secure. Actually we will not prove the exact soundness of (P^2_{Mic}, V^2_{Mic}), because we use the third protocol (P_3, V_3) to deal with the situation that (P_1, V_1) is not secure.

With function ensemble $\mathcal{G}_1, \mathcal{G}_2$, which we determine later, protocol (P_1, V_1) is constructed as follows:

- Common input: $x \in \{0,1\}^{l_x(n)}$.
- Auxiliary input to the prover: none.

1. V_1:
   (a) Compute $f \leftarrow F_n$.
   (b) Compute $crs' = (f^{TC}, g_1, g_2, \mathsf{comm}) \leftarrow \mathsf{Gen}^{\mathcal{G}_1, \mathcal{G}_2}_{Mic}$.
   (c) Let $crs = (f, crs')$ and output it.

2. P_1: On input (x, crs), output an arbitrary $a \in \{0,1\}^n$.
3. V_1: On input (x, crs, a), output random $b, r_c \leftarrow \{0,1\}^n$.
4. P_1: On input $(x, crs, a, b\|r_c)$, outputs ans.
5. V_1: On input $(x, crs, a, b\|r_c, ans)$:
   (a) Parse crs as f, crs'.
   (b) Accept if and only if $V^2_{Mic}(f\|a\|b, crs', r_c, ans) = 1$.

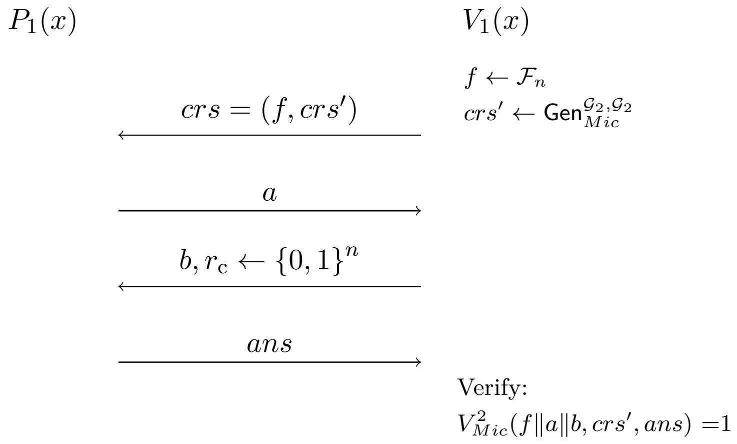

$P_1(x)$
$V_1(x)$
$f \leftarrow \mathcal{F}_n$
$crs' \leftarrow \mathsf{Gen}^{\mathcal{G}_2,\mathcal{G}_2}_{Mic}$
$crs = (f, crs')$
a
$b, r_c \leftarrow \{0,1\}^n$
ans
Verify:
$V^2_{Mic}(f\|a\|b, crs', ans) = 1$

Figure 9: Protocol 1

Lemma 6.7. For all function ensemble $\mathcal{H} \in \mathcal{C}$, apply the Fiat-Shamir transform on (P_1, V_1) with respect to $\mathcal{H}$ gives $({P_1}^{FS}, {V_1}^{FS})$ that is not a non-interactive public coin argument for empty language against $\mathcal{C}$ adversary. $\diamondsuit$

The insecurity of $({P_1}^{FS}, {V_1}^{FS})$ is relatively straight-forward. As the adversary knows the witness for the central relation.

Proof. Here we construct an adversary P^* for $({P_1}^{FS}, {V_1}^{FS})$ as below:
On input $(x, h\|crs)$:

1. Parse crs as f, crs'.
2. Hard-code x into the input of h and truncate the output to only contain b, to get C. Compute the encoding $\tilde{C}$.
3. Compute $a \leftarrow \mathsf{TC}_f(\tilde{C}), b \leftarrow C(a)$, and record the computation history Π.
4. Compute $(b, r_c) \leftarrow h(x\|a)$.
5. Emulate $(P_{Mic}, V_{Mic}|_{crs'})$ with input $f\|a\|b$ and witness $\hat{C}\|\Pi$, to produce ans.
6. Output a, ans.

Note that P^* is in $\mathcal{C}$ and it can always succeed due to the completeness of (P^2_{Mic}, V^2_{Mic}). $\square$

The completeness of (P_1, V_1) is also trivial, as the language is empty and the honest prover can just output some fixed strings. However, we cannot prove the security of (P_1, V_1) directly. Here, we consider 2 cases:

- (P_1, V_1) is secure. In this case, (P_1, V_1) is the protocol that the theorem states.
- (P_1, V_1) is not secure. In this case, we want to prove the insecurity of $({P_3}^{FS}, {V_3}^{FS})$, which will be introduced later.

Instead of directly discussing the security of (P_1, V_1), we use another notation for this:

Definition 6.8 (IMPERSONATOR). For any Fiat-Shamir hash function family $\mathcal{G}_1, \mathcal{G}_2 \in \mathcal{C}$, define the statement $\exists$IMPERSONATOR to be the following. There exists a polynomial $p(n)$ and $\mathcal{C}$ circuit families $P^{IMP} = \left\{P_n^{IMP}\right\}_{n\in\mathbb{N}}$ and $F^{IMP} = \left\{F_n^{IMP}\right\}_{n\in\mathbb{N}}$ such that for infinite many $n \in \mathbb{N}$,

$$\Pr\left[(P^{IMP}(aux), V_{Mic}^2|_{crs})(f\|a\|b) = 1 \;\middle|\; \begin{array}{l} f \leftarrow \mathcal{F}_n, b \leftarrow \{0,1\}^n \\ crs \leftarrow \mathsf{Gen}_{Mic}^{\mathcal{G}_1,\mathcal{G}_2} \\ a, aux \leftarrow F^{IMP}(f, crs) \end{array}\right] \geq \frac{1}{p(n)}. \qquad \diamondsuit$$

We can observe that this is equivalent to the security of (P_1, V_1), as the IMPERSONATOR P^{IMP}, F^{IMP} can be used as the adversary.

Lemma 6.9. If there exists $\mathcal{G}_1, \mathcal{G}_2 \in \mathcal{C}$ such that $\neg\exists$IMPERSONATOR then with respect to $\mathcal{G}_1, \mathcal{G}_2$, (P^1, V^1) is a 3-round public coin argument for empty language against adversary in $\mathcal{C}$. $\diamondsuit$

Proof. It follows directly from the fact that the definition of $\neg\exists$IMPERSONATOR is exactly the soundness condition. $\square$

6.4 Protocol 2

In the second protocol (P_2, V_2), we are not aiming to construct some protocol that is easy to break after Fiat-Shamir transformation, but some protocol that is definitely secure. In the protocol, we first generate function f for central relation in the CRS, then let the prover decide a, b_1, b_2 that $b_1 \neq b_2$ and use two parallel (P_{Kil}, V_{Kil}) to prove the knowledge of witness of $f\|a\|b_1$ and $f\|a\|b_2$. It is impossible for the adversary to know the witnesses for both $f\|a\|b_1$ and $f\|a\|b_2$, due to the collision resistance of the tree commitment, so it is definitely secure. In the protocol, although $a, b_1, b_2, \beta_1, \beta_2$ are determined before sending the first message, we will not send all of them in the first message. Instead, a special version of commitments of them will be sent, due to some complicated details in the proof.

Protocol (P_2, V_2) is constructed as follows:

- Common input: $x \in \{0,1\}^{l_x(n)}$.
- Auxiliary input to the prover: none.

1. V_2:
    (a) Compute $f, f^{TC} \leftarrow \mathcal{F}_n$.
    (b) Compute $\mathsf{comm} \leftarrow \mathsf{Comm}_n$ and $r_c, r' \leftarrow \{0,1\}^n$.

(c) Compute $\gamma_1' \leftarrow \{0,1\}^n$.

(d) Let $crs = (f, f^{TC}, \mathsf{comm}, r_c, r', \gamma_1')$ and output it.

2. P_2: On input (x, crs):

   (a) Decide $a, b_1, b_2 \in \{0,1\}^n$.

   (b) Decide $\beta_1, \beta_2 \in \{0,1\}^n$.

   (c) Compute $\hat{\beta}_2 = \mathsf{comm}(\beta_2 \| (\mathsf{comm}(\beta_1 \| a \| b_1 \| b_2 \| r_c) \oplus r'))$, the commitment of $a, b_1, b_2, \beta_1, \beta_2$. Recall that shorter inputs can be padded to the correct length by padding 0s.

   (d) Output $\hat{\beta}_2$.

3. V_2: On input $(x, crs, \hat{\beta}_2)$, output random $\gamma_1'', \gamma_2 \in \{0,1\}^n$.

4. P_2: On input $(x, crs, \hat{\beta}_2, \gamma_1'' \| \gamma_2)$:

   (a) Decide $\delta_1, \delta_2 \in \{0,1\}^n$.

   (b) Output $a, b_1, b_2, \beta_1, \beta_2, \delta_1, \delta_2$.

5. V_2: On input $(x, crs, \hat{\beta}_2, \gamma_1'' \| \gamma_2, a \| b_1 \| b_2 \| \beta_1 \| \beta_2 \| \delta_1 \| \delta_2)$,

   (a) Parse crs as $f, f^{TC}, \mathsf{comm}, r_c, r', \gamma_1'$.

   (b) Check $b_1 \neq b_2$.

   (c) Check $\hat{\beta}_2 = \mathsf{comm}(\beta_2 \| (\mathsf{comm}(\beta_1 \| a \| b_1 \| b_2 \| r_c) \oplus r'))$.

   (d) Check $V_{Kil}(f \| a \| b_1, f^{TC}, \beta_1, \gamma_1' \oplus \gamma_1'', \delta_1) = 1$.

   (e) Check $V_{Kil}(f \| a \| b_2, f^{TC}, \beta_2, \gamma_2, \delta_2) = 1$.

   (f) Accept if and only if all checks are passed.

Here we first prove the security of (P_2, V_2). The intuition is to reduce the security to the security of CRHF. If there is an adversary that breaks (P_2, V_2), due to the knowledge extraction property of (P_{Kil}, V_{Kil}), we can actually get f, a, b_1, b_2, C_1, C_2 such as $(f \| a \| b_1, \hat{C_1} \| \Pi) \in \mathcal{R}$ and $(f \| a \| b_2, \hat{C_2} \| \Pi) \in \mathcal{R}$. Because $C_1(a) = b_1 \wedge C_2(a) = b_2 \wedge b_1 \neq b_2$, we know C_1, C_2 must be two different circuits, which breaks the collision resistant property of $\mathsf{TC}_{f^{TC}}$.

Lemma 6.10. (P_2, V_2) is a 3-round public coin argument for empty language against adversary in $\mathcal{C}$. ♢

Proof. Completeness is trivially satisfied. Now we prove soundness by contradiction. Suppose there exists an adversary circuit family P^* in $\mathcal{C}$ and a polynomial $p(n)$ along with instance x such that

$$\Pr[(P^*, V_2)(x) = 1] \geq \frac{1}{p(n)}.$$

Then we can construct circuit families $\tilde{F}$, $\tilde{P}_1$ and $\tilde{P}_2$ in $\mathcal{C}$ such that there exists a polynomial $p'(n)$ and

$$\Pr\left[\begin{matrix}(\tilde{P}_1(aux), V_{Kil}|_{crs})(f \| a \| b_1) = 1 \wedge \\ (\tilde{P}_2(aux), V_{Kil}|_{crs})(f \| a \| b_2) = 1\end{matrix} \;\middle|\; (crs, a, b_1, b_2, aux) \leftarrow \tilde{F}(f, f^{TC})\right] \geq \frac{1}{p'(n)}$$

where the randomness is over $f, f^{TC} \leftarrow \mathcal{F}_n$ and the randomness of all these circuits.

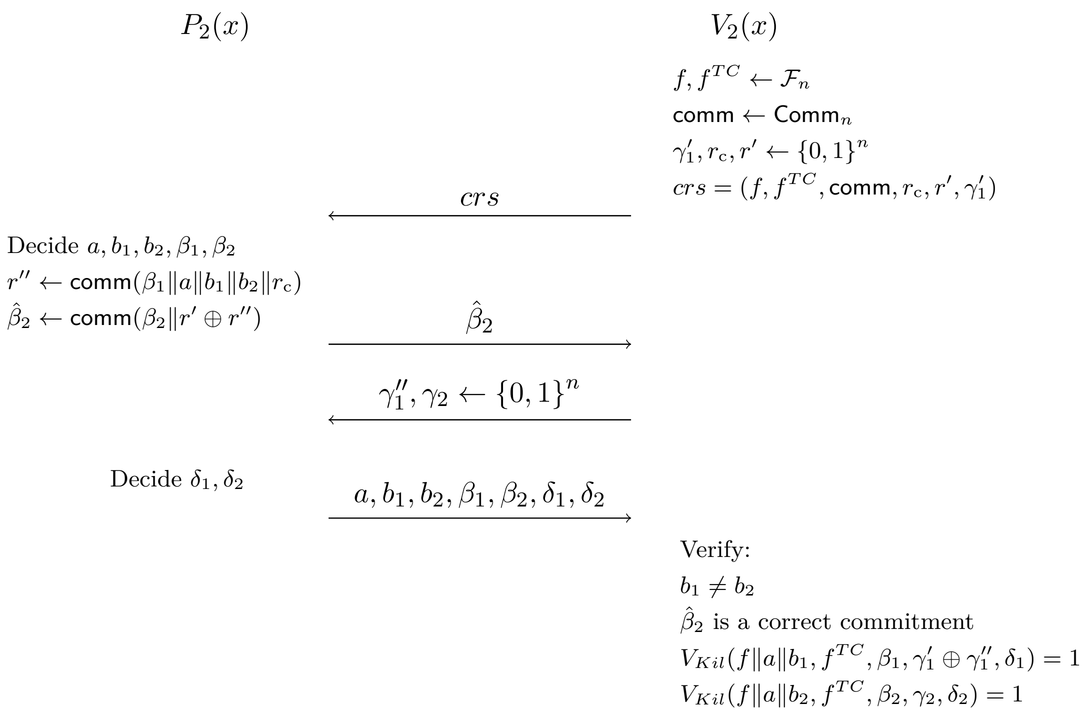

$P_2(x)$
$V_2(x)$
$f, f^{TC} \leftarrow \mathcal{F}_n$
$\mathsf{comm} \leftarrow \mathsf{Comm}_n$
$\gamma_1', r_c, r' \leftarrow \{0,1\}^n$
$crs = (f, f^{TC}, \mathsf{comm}, r_c, r', \gamma_1')$
crs
Decide $a, b_1, b_2, \beta_1, \beta_2$
$r'' \leftarrow \mathsf{comm}(\beta_1 \| a \| b_1 \| b_2 \| r_c)$
$\hat{\beta}_2 \leftarrow \mathsf{comm}(\beta_2 \| r' \oplus r'')$
$\hat{\beta}_2$
$\gamma_1'', \gamma_2 \leftarrow \{0,1\}^n$
Decide δ_1, δ_2
$a, b_1, b_2, \beta_1, \beta_2, \delta_1, \delta_2$
Verify:
$b_1 \neq b_2$
$\hat{\beta}_2$ is a correct commitment
$V_{Kil}(f \| a \| b_1, f^{TC}, \beta_1, \gamma_1' \oplus \gamma_1'', \delta_1) = 1$
$V_{Kil}(f \| a \| b_2, f^{TC}, \beta_2, \gamma_2, \delta_2) = 1$

Figure 10: Protocol 2

- $\tilde{F}$ works as follows:
  1. Select $\mathsf{comm} \leftarrow \mathsf{Comm}_n$ and $r_c, r', \gamma_1' \leftarrow \{0,1\}^n$.
  2. Let $crs' = (f, f^{TC}, \mathsf{comm}, r_c, r', \gamma_1')$.
  3. Select a random tape r_p for P^*. Emulate the interaction

     $$(\hat{\beta}_2, (\gamma_1'', \gamma_2), (a, b_1, b_2, \beta_1, \beta_2, \delta_1, \delta_2)) \leftarrow (P^*, V_2|_{crs'})(x)$$

     where P^* uses r_p as its randomness.
  4. Let $crs = f^{TC}, aux = (\beta_1, \beta_2, r_p, crs')$ and output (crs, a, b_1, b_2, aux).
- $\tilde{P}_1(aux)$ works as follows:
  1. crs from $V_{Kil}|_{crs}$ and parse it as f^{TC}.
  2. Parse aux as $(\beta_1, \beta_2, r_p, crs')$.
  3. Output β_1.
  4. Receive γ_1^1 from $V_{Kil}|_{crs}$.
  5. Select $\gamma_2^1 \leftarrow \{0,1\}^n$ and emulate the interaction

     $$(\hat{\beta}_2, (\gamma_1' \oplus \gamma_1^1, \gamma_2^1), (a', b_1', b_2', \beta_1', \beta_2', \delta_1', \delta_2')) \leftarrow (P^*, V_2|_{crs', (\gamma_1' \oplus \gamma_1^1, \gamma_2^1)})(x)$$

     where P^* is invoked using the same random tape r_p as the emulation in $\tilde{F}$.
  6. Output δ_1'.

- $\tilde{P}_2(aux)$ works as follows:
    1. crs from $V_{Kil}|_{crs}$ and parse it as f^{TC}.
    2. Parse aux as $(\beta_1, \beta_2, r_p, crs')$.
    3. Output β_2.
    4. Receive γ_2^2 from $V_{Kil}|_{crs}$.
    5. Select $\gamma_1^2 \leftarrow \{0,1\}^n$ and emulate the interaction
    $$(\hat{\beta}_2, (\gamma_1^2, \gamma_2^2), (a'', b_1'', b_2'', \beta_1'', \beta_2'', \delta_1'', \delta_2'')) \leftarrow (P^*, V_2|_{crs', (\gamma_1^2, \gamma_2^2)})(x)$$
    where P^* is invoked using the same random tape r_p as the emulation in $\tilde{F}$.
    6. Output δ_2''.

To see that $\tilde{F}, \tilde{P}_1$ and $\tilde{P}_2$ satisfy the conditions above. Notice that

1. $\gamma_1' \oplus \gamma_1^1, \gamma_2^1, \gamma_1^2$ and γ_2^2 are uniform random strings thus we can emulate these interaction correctly.

2. Since we invoke P^* with the same randomness r_p every time, the first output $\hat{\beta}_2$ will be the same every time. And since Comm is collision resistant, we have
$$(a, b_1, b_2, \beta_1, \beta_2) = (a', b_1', b_2', \beta_1', \beta_2') = (a'', b_1'', b_2'', \beta_1'', \beta_2'').$$

3. $V_{Kil}|_{crs}$ will accept the transcript $(\beta_1, \gamma_1^1, \delta_1')$ since it is exactly one of the accept condition for V_2. For the same reason $V_{Kil}|_{crs}$ will also accept the transcript $(\beta_2, \gamma_2^2, \delta_2'')$.

Given $\tilde{F}$, $\tilde{P}_1$ and $\tilde{P}_2$ we can find a collision for f and thus contradict to our assumption that $\mathcal{F}$ is collision resistant. By the proof of knowledge property, given oracle access to $\tilde{F}, \tilde{P}_1$ and $\tilde{P}_2$, there exists a extractor circuit family $E = \{E_n\}_{n \in \mathbb{N}}$ such that:
$$\Pr\left[\begin{matrix}((f\|a\|b_1), w_1) \in \mathcal{R} \wedge \\ ((f\|a\|b_2), w_2) \in \mathcal{R}\end{matrix} \;\middle|\; \begin{matrix} w_1 \leftarrow E^{\tilde{P}_1(aux)}(f\|a\|b_1) \\ w_2 \leftarrow E^{\tilde{P}_2(aux)}(f\|a\|b_2)\end{matrix}\right] = 1.$$
Note that with non-negligible probability we obtain w_1 and w_2 simultaneously and we have $TC_f(w_1) = TC_f(w_2)$. Note that we use the encoding for the circuit with high minimum distance. To find a collision of f, we select a random position i, the probability that the i-th position of w_1 and w_2 are different is non-negligible. This is because the history Π is of polynomial length of the circuit encoding $\hat{C}$ and the encoding has high minimum distance (which means the witness itself has high minimum distance). Thus by traversing the path from position i to the tree root on both tree commitments we can obtain a collision for f. □

For $({P_2}^{FS}, {V_2}^{FS})$, we do not know how to prove the insecurity. So, we also separate this situation into 2 cases:

- (P_3, V_3), the protocol we will introduce later, is not secure. In this case, we can prove $({P_2}^{FS}, {V_2}^{FS})$ is also not secure.

- (P_3, V_3) is secure.

Here, we use strong-IMPERSONATOR to characterize the security of (P_3, V_3).

Definition 6.11 (strong-IMPERSONATOR). For any Fiat-Shamir hash function family $\mathcal{G} \in \mathcal{C}$, define statement $\exists$strong-IMPERSONATOR to be the following. There exists a polynomial $p(n)$ and $\mathcal{C}$ circuit families P_1^{sIMP}, P_2^{sIMP} and F^{sIMP} such that for infinite many $n \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l|l} (P_1^{sIMP}(aux), V_{Kil}|_{f^{TC}})(f\|a\|b_1) = 1 \wedge & f \leftarrow \mathcal{F}_n, b_2 \leftarrow \{0,1\}^n \\ (P_2^{sIMP}(aux), V_{Mic}^1|_{crs})(f\|a\|b_2) = 1 & crs = (f^{TC}, g, \mathsf{comm}) \leftarrow \mathsf{Gen}_{Mic}^{\mathcal{G}} \\ & a, b_1, aux \leftarrow F^{sIMP}(f, crs) \end{array}\right] \geq \frac{1}{p(n)} \qquad \diamondsuit$$

Lemma 6.12. If for any function ensemble $\mathcal{H} \in \mathcal{C}$ there exists strong-IMPERSONATOR, then applying the Fiat-Shamir transform on (P_2, V_2) with respect to $\mathcal{H}$ gives $({P_2}^{FS}, {V_2}^{FS})$ that is not a non-interactive public coin argument for empty language against $\mathcal{C}$ adversary. $\diamondsuit$

Proof. Assume $h \leftarrow \mathcal{H}$ is the transformation function. The adversary P^* works as follows:

On input $(x, h\|crs)$:

1. Parse crs as $f, f^{TC}, \mathsf{comm}, r_c, r', \gamma_1'$.
2. Hard-code x into the input of h, and truncate the output to only contain γ_2, to get h'. Let $crs' = (f^{TC}, h', \mathsf{comm})$.
3. Emulate
$$(a, b_1, aux) \leftarrow F_n^{sIMP}(f, crs')$$
4. Emulate
$$(f^{TC}, \beta_1, *, *) \leftarrow (P_1^{sIMP}(aux), V_{Kil}|_{f^{TC}})(f\|a\|b_1)$$
5. Select $b_2 \leftarrow \{0,1\}^n$ and compute $r'' \leftarrow \mathsf{Comm}(\beta_1\|a\|b_1\|b_2\|r_c)$
6. Emulate
$$(crs, r' \oplus r'', (\beta_2, \hat{\beta}_2, \gamma_2, \delta_2)) \leftarrow (P_2^{sIMP}(aux), V_{Mic}^1|_{crs', r' \oplus r''})(f\|a\|b_2)$$
7. $(\gamma_1'', *) = h(\hat{\beta}_2)$ and emulate
$$(\beta_1, *, \delta_1) \leftarrow (P_1^{sIMP}(aux), V_{Kil}|_{f^{TC}, \gamma_1' \oplus \gamma_1''})$$
8. Output $(\hat{\beta}_2, (a, b_1, b_2, \beta_1, \beta_2, \delta_1, \delta_2))$.

Consider the conditions that V_2 accepts:

1. Since $b_2 \leftarrow \{0,1\}^n$ is an independent random string, the condition $b_1 \neq b_2$ holds with probability $1 - 2^{-n}$.
2. γ_1' is uniformly random thus the call to $V_{Kil}|_{f^{TC}, \gamma_1' \oplus \gamma_1''}$ is a valid call.
3. $r' \oplus r''$ is uniform random since r' is uniform random, thus we can call $V_{Mic}^1|_{crs', r' \oplus r''}$ correctly and obtain $\hat{\beta}_2 = \mathsf{Comm}(\beta_2, r_c')$. Together with $r_c' = r' \oplus \mathsf{Comm}(\beta_1\|a\|b_1\|b_2\|r_c)$, we pass $\hat{\beta}_2 = \mathsf{comm}(\beta_2\|(\mathsf{comm}(\beta_1\|a\|b_1\|b_2\|r_c) \oplus r'))$.
4. $V_{Kil}(f\|a\|b_1, f^{TC}, \beta_1, \gamma_1' \oplus \gamma_1'', \delta_1) = 1$ and $V_{Kil}(f\|a\|b_2, f^{TC}, \beta_2, \gamma_2, \delta_2) = 1$ are satisfied by the definition of $\exists$strong-IMPERSONATOR.

Since P^* above is in $\mathcal{C}$, $({P_2}^{FS}, {V_2}^{FS})$ is not a non-interactive public coin argument for empty language against $\mathcal{C}$ adversary. $\square$

6.5 Protocol 3

In the third protocol (P_3, V_3), we first generate function f for central relation in the CRS, then let the prover decide a, b_1 while letting the verifier choose b_2. One (P_{Kil}, V_{Kil}) is invoked on instance $f\|a\|b_1$, while one (P^1_{Mic}, V^1_{Mic}) is invoked on instance $f\|a\|b_2$. Components used in (P_{Kil}, V_{Kil}) are labeled with subscript 1, and components used in (P^1_{Mic}, V^1_{Mic}) are labeled with subscript 2. Again, we use a commitment of a, b_1, β_1, instead of sending them in the first message, because it is necessary if we want to use IMPERSONATOR to break $({P_3}^{FS}, {V_3}^{FS})$. All the $\oplus$ here is used to make the view of IMPERSONATOR same as an uniform distribution after the Fiat-Shamir transformation.

With function ensemble $\mathcal{G}$, which we determine later, protocol (P_3, V_3) is constructed as follows:

- Common input: $x \in \{0,1\}^{l_x(n)}$.
- Auxiliary input to the prover: none.

1. V_3:
   (a) Compute $f \leftarrow F_n$.
   (b) Compute $crs' = (f^{TC}, g, \mathsf{comm}) \leftarrow \mathsf{Gen}^{\mathcal{G}}_{Mic}$.
   (c) Compute $b'_2, r_{\mathrm{c},1}, r'_{\mathrm{c},2}, r' \leftarrow \{0,1\}^n$.
   (d) Let $crs = (f, crs', b'_2, r_{\mathrm{c},1}, r'_{\mathrm{c},2}, r')$ and output it.
2. P_3: On input (x, crs):
   (a) Parse crs as $f, f^{TC}, g, \mathsf{comm}, b'_2, r_{\mathrm{c},1}, r'_{\mathrm{c},2}, r'$.
   (b) Decide $a, b_1 \in \{0,1\}^n$.
   (c) Decide $\beta_1 \in \{0,1\}^n$.
   (d) Compute $\hat{\beta}_1 \leftarrow \mathsf{comm}(\beta_1\|(\mathsf{comm}(a\|b_1\|r_{\mathrm{c},1}) \oplus r'))$, the commitment of a, b_1, β_1.
   (e) Output $\hat{\beta}_1$.
3. V_3: On input $(x, crs, \hat{\beta}_1)$, output random $b''_2, r''_{\mathrm{c},2}, \gamma_1 \leftarrow \{0,1\}^n$.
4. P_3: On input $(x, crs, \hat{\beta}_1, b''_2\|r''_{\mathrm{c},2}\|\gamma_1)$:
   (a) Decide δ_1 and ans.
   (b) Output $a, b_1, \beta_1, \delta_1, ans$.
5. V_3: On input $(x, crs, \hat{\beta}_1, b''_2\|r''_{\mathrm{c},2}\|\gamma_1, a\|b_1\|\beta_1\|\delta_1\|ans)$:
   (a) Parse crs as $f, crs', b'_2, r_{\mathrm{c},1}, r'_{\mathrm{c},2}, r'$.
   (b) Parse crs' as f^{TC}, g, comm.
   (c) Check $\hat{\beta}_1 = \mathsf{comm}(\beta_1\|(\mathsf{comm}(a\|b_1\|r_{\mathrm{c},1}) \oplus r'))$.
   (d) Check $V_{Kil}(f\|a\|b_1, f^{TC}, \beta_1, \gamma_1, \delta_1) = 1$.
   (e) Check $V^1_{Mic}(f\|a\|(b'_2 \oplus b''_2), crs', r'_{\mathrm{c},2} \oplus r''_{\mathrm{c},2}, ans) = 1$.

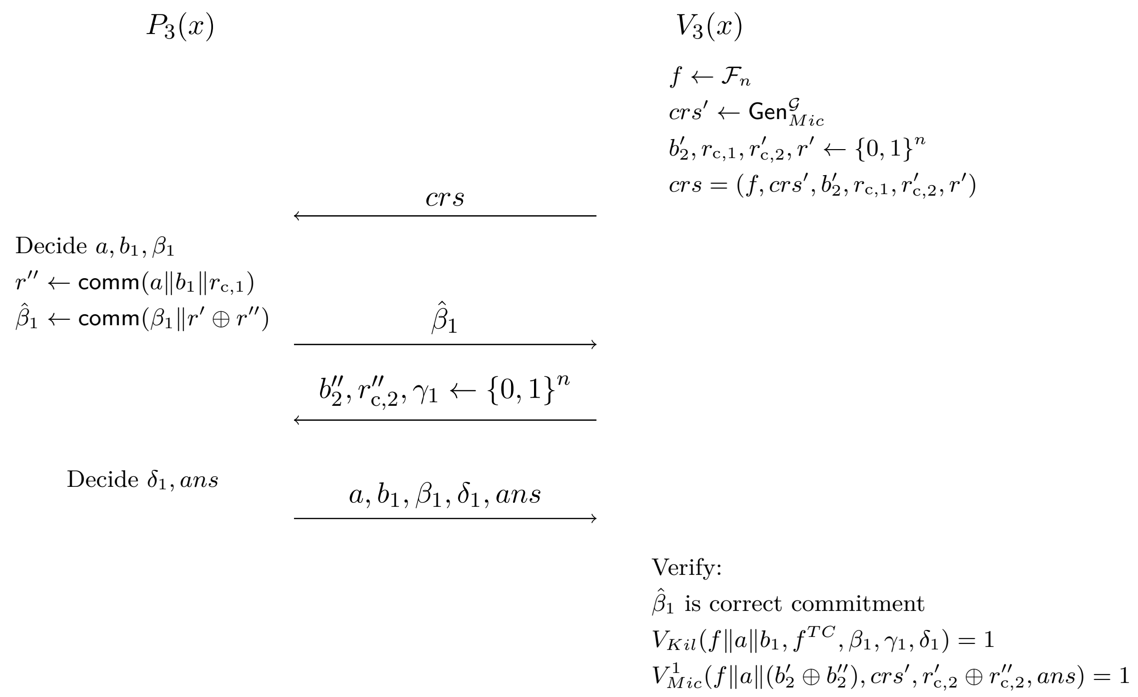

$P_3(x)$
$V_3(x)$
$f \leftarrow \mathcal{F}_n$
$crs' \leftarrow \mathsf{Gen}^{\mathcal{G}}_{Mic}$
$b'_2, r_{c,1}, r'_{c,2}, r' \leftarrow \{0,1\}^n$
$crs = (f, crs', b'_2, r_{c,1}, r'_{c,2}, r')$
crs
Decide a, b_1, β_1
$r'' \leftarrow \mathsf{comm}(a\|b_1\|r_{c,1})$
$\hat{\beta}_1 \leftarrow \mathsf{comm}(\beta_1\|r' \oplus r'')$
$\hat{\beta}_1$
$b''_2, r''_{c,2}, \gamma_1 \leftarrow \{0,1\}^n$
Decide δ_1, ans
$a, b_1, \beta_1, \delta_1, ans$
Verify:
$\hat{\beta}_1$ is correct commitment
$V_{Kil}(f\|a\|b_1, f^{TC}, \beta_1, \gamma_1, \delta_1) = 1$
$V^1_{Mic}(f\|a\|(b'_2 \oplus b''_2), crs', r'_{c,2} \oplus r''_{c,2}, ans) = 1$

Figure 11: Protocol 3

In the second protocol, we have defined the strong-IMPERSONATOR, which is actually the adversary for (P_3, V_3).

Lemma 6.13. If $\exists \mathcal{G} \in \mathcal{C}$, $\neg\exists$strong-IMPERSONATOR, then there exists a function ensemble $\mathcal{G} \in \mathcal{C}$ such that (P_3, V_3) is a 3-round public coin argument for empty language against adversary in $\mathcal{C}$. ♢

Proof. It is trivial, since ($\neg(\forall\mathcal{G}\ \exists$strong-IMPERSONATOR)) is exact the soundness condition. Note that b'_2, r_c, r'_c, r' do not help in breaking the protocol. □

After the Fiat-Shamir transformation, $({P_3}^{FS}, {V_3}^{FS})$ is similar to a protocol that first runs (P^1_{Mic}, V^1_{Mic}) with $\mathcal{H}$ for $f\|a\|b_1$ and then runs (P^1_{Mic}, V^1_{Mic}) with $\mathcal{G}$ for $f\|a\|b_2$. The IMPERSONATOR can break (P^2_{Mic}, V^2_{Mic}), and we can use it to break two (P^1_{Mic}, V^1_{Mic}) with different hash functions.

Lemma 6.14. If $\forall\mathcal{G}_1, \mathcal{G}_2 \in \mathcal{C}$, there exists IMPERSONATOR, then for all function ensemble $\mathcal{G}, \mathcal{HC}$, applying the Fiat-Shamir transformation on (P_3, V_3) with respect to $\mathcal{H}$ gives $({P_3}^{FS}, {V_3}^{FS})$ that is not a non-interactive public coin argument for empty language against $\mathcal{C}$ adversary. ♢

Proof. For any $\mathcal{H}$ and input x, let $\mathcal{H}'$ be the function ensemble obtained by hard-coding x into input of all $h \in \mathcal{H}$ and truncating all $h \in \mathcal{H}$ to only output last n bits, γ_1.

By assumption, we know for $\mathcal{H}', \mathcal{G}$, there exists IMPERSONATOR, denoted as P^{IMP} and F^{IMP}. Without loss of generality, we assume that crs is included in aux where $a, aux \leftarrow F^{IMP}(f, crs)$.

Claim 6.15. From P^{IMP}, F^{IMP}, we can construct $P_{\mathcal{H}'}^{IMP}$ and $P_{\mathcal{G}}^{IMP}$ in $\mathcal{C}$ with polynomial $p(n)$, that for infinite many $n \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l|l} & f \leftarrow \mathcal{F}_n \\ (P_{\mathcal{H}'}^{IMP}(aux), V_{Mic}^1|_{crs_1})(f\|a\|b_1) = 1 \wedge & b_1, b_2 \leftarrow \{0,1\}^n \\ (P_{\mathcal{G}}^{IMP}(aux), V_{Mic}^1|_{crs_2})(f\|a\|b_2) = 1 & crs \leftarrow \mathsf{Gen}_{Mic}^{\mathcal{H}',\mathcal{G}} \\ & a, aux \leftarrow F^{IMP}(f, crs) \end{array}\right] > 1/p(n),$$

Here, crs_1, crs_2 are the modified $crs = (f^{TC}, h', g, \mathsf{comm})$ that g is removed and h' is removed respectively. ♢

Proof. We show the construction of $P_{\mathcal{H}'}^{IMP}$ below, $P_{\mathcal{G}}^{IMP}$ can be constructed similarly:

On input $(aux, f\|a\|b, crs_1, r_c)$:

1. Get crs from aux.
2. Parse crs as $f^{TC}, h', g, \mathsf{comm}$.
3. Emulate
$$(crs, r_c, (\hat{\beta}, \beta, \gamma_1, \gamma_2, \delta_1, \delta_2)) \leftarrow (P^{IMP}, V_{Mic}^2|_{crs, r_c})(f\|a\|b)$$
4. Output $ans = (\beta, \hat{\beta}, \gamma_1, \delta_1)$.

From the definition of P^{IMP}, F^{IMP}, we know there exists a polynomial $q(n)$, with non-negligible probability over choice of f, crs and $a, aux \leftarrow F(f, crs)$,

$$\Pr\left[(P^{IMP}(aux), V_{Mic}^2|_{crs})(f\|a\|b) = 1 \;\middle|\; b \leftarrow \{0,1\}^n\right] > \frac{1}{q(n)}.$$

From the construction above, we know with non-negligible probality over the choice of f, crs and a, aux,

$$\begin{aligned} &\Pr\left[\begin{array}{l|l} (P_{\mathcal{H}'}^{IMP}(aux), V_{Mic}^1|_{crs_1})(f\|a\|b_1) = 1) \wedge & \\ (P_{\mathcal{G}}^{IMP}(aux), V_{Mic}^1|_{crs_2})(f\|a\|b_2) = 1) & b_1, b_2 \leftarrow \{0,1\}^n \end{array}\right] \\ =&\Pr\left[(P^{IMP}(aux), V_{Mic}^2|_{crs})(f\|a\|b) = 1 \;\middle|\; b \leftarrow \{0,1\}^n\right]^2 \\ \geq&\frac{1}{q^2(n)}, \end{aligned}$$

which proves the claim. □

Here, we use $P_{\mathcal{H}'}^{IMP}, P_{\mathcal{G}}^{IMP}, F^{IMP}$ to construct an adversary P^* for $({P_3}^{FS}, {V_3}^{FS})$ as below:

On input $(x, h\|crs)$:

1. Parse crs as $f, crs', b_2', r_{c,1}, r_{c,2}', r'$.
2. Parse crs' as f^{TC}, g, comm.
3. Compute $a, aux \leftarrow F^{IMP}(f, crs')$.
4. Compute $b_1 \leftarrow \{0,1\}^n$ and $r'' \leftarrow \mathsf{comm}(a\|b_1\|r_{c,1})$.

5. Let h' be the corresponding function of h in $\mathcal{H}'$, and emulate

$$(f^{TC}\|h'\|\mathsf{comm}, r' \oplus r'', (\beta_1, \hat{\beta}_1, \gamma, \delta)) \leftarrow (P^{IMP}_{\mathcal{H}'}(aux), V^1_{Mic}|_{f^{TC}\|h'\|\mathsf{comm}, r'\oplus r''})(f\|a\|b_1)$$

6. Compute $b''_2, r''_{\mathrm{c},2}, \gamma_1 \leftarrow h(x\|\hat{\beta}_1)$ and $b_2 = b'_2 \oplus b''_2$.
7. Emulate

$$(f^{TC}\|g\|\mathsf{comm}, r'_{\mathrm{c},2} \oplus r''_{\mathrm{c},2}, ans) \leftarrow (P^{IMP}_{\mathcal{G}}(aux), V^1_{Mic}|_{f^{TC}\|g\|\mathsf{comm}, r'_{\mathrm{c},2}\oplus r''_{\mathrm{c},2}})(f\|a\|b_2)$$

8. Output $\hat{\beta}_1, (a, b_1, \beta_1, \delta_1, ans)$.

Claim 6.16. There exists a polynomial $p(n)$, that for infinite many n, for all $x \in \{0,1\}^n$,

$$\Pr[(P^*, V_3)(x) = 1] \geq \frac{1}{p(n)},$$

where the probability is over P^* and V_3. ♢

Proof. According to claim 6.15, we know for infinite many $n \in \mathbb{N}$,

$$\Pr\left[\begin{array}{l} (P^{IMP}_{\mathcal{H}'}(aux), V^1_{Mic}|_{crs_1})(f\|a\|b_1) = 1 \wedge \\ (P^{IMP}_{\mathcal{G}}(aux), V^1_{Mic}|_{crs_2})(f\|a\|b_2) = 1 \end{array} \middle| \begin{array}{l} f \leftarrow \mathcal{F}_n, b_1, b_2 \leftarrow \{0,1\}^n \\ crs \leftarrow \mathsf{Gen}^{\mathcal{H}',\mathcal{G}}_{Mic} \\ a, aux \leftarrow F^{IMP}(f, crs) \end{array}\right] > 1/p(n).$$

Now we claim similarly that for infinite many n, for all $x \in \{0,1\}^n$,

$$\Pr\left[\begin{array}{l} (P^{IMP}_{\mathcal{H}'}(aux), V^1_{Mic}|_{crs_1, r'\oplus r''})(f\|a\|b_1) = 1 \wedge \\ (P^{IMP}_{\mathcal{G}}(aux), V^1_{Mic}|_{crs_2, r'_{\mathrm{c},2}\oplus r''_{\mathrm{c},2}})(f\|a\|b'_2 \oplus b''_2) = 1 \end{array} \middle| \begin{array}{l} crs = (f, crs', b'_2, r_{\mathrm{c},1}, r'_{\mathrm{c},2}, r') \leftarrow V_3 \\ a, aux \leftarrow F^{IMP}(f, crs') \\ r''_{\mathrm{c},2}, b''_2 \leftarrow P^*(x, crs) \end{array}\right] > 1/p(n),$$

which is because $b'_2, r'_{\mathrm{c},2}, r'$ is selected uniformly and independently in $\{0,1\}^n$. Here, crs_1, crs_2 is the CRS that g is removed from crs' and h' is removed from crs' respectively. As a result, with non-negligible probability over interaction $(P^*, V_3)(x)$, the verifier accepts because the following conditions hold:

- $V^1_{Mic}(f\|a\|b_1, crs_1, r' \oplus r'', ans_1)$ accepts, which indicates:
  - $(*, *, \gamma_1) = h(x\|\hat{\beta}_1)$.
  - $V_{Kil}(f^{TC}, \beta_1, \gamma_1, \delta_1)$ accepts.
  - $\hat{\beta}_1 = \mathsf{comm}(\beta_1\|(\mathsf{comm}(a\|b_1\|r_{\mathrm{c},1}) \oplus r'))$
- $V^1_{Mic}(f\|a\|(b'_2 \oplus b''_2), crs_2, r'_{\mathrm{c}} \oplus r''_{\mathrm{c}}, ans_2)$ accepts. □

Finally, we conclude that one of these three protocols is the required protocol in Theorem 6.1. There are three cases:

1. If $\exists \mathcal{G}_1, \mathcal{G}_2 \in \mathcal{C}$ that $\neg\exists$IMPERSONATOR), then

- (P_1, V_1) is a 3-round public coin argument for empty language in $\mathcal{C}$ secure against any adversary in $\mathcal{C}$.
- Applying the Fiat-Shamir transformation on (P_1, V_1) with any $\mathcal{H} \in \mathcal{C}$ results in a protocol $({P_1}^{FS}, {V_1}^{FS})$ that is insecure against some adversary in $\mathcal{C}$.

2. If $\forall \mathcal{G} \in \mathcal{C}$ that $\exists$strong-IMPERSONATOR, then
   - (P_2, V_2) is a 3-round public coin argument for empty language in $\mathcal{C}$ secure against any adversary in $\mathcal{C}$.
   - Applying the Fiat-Shamir transformation on (P_2, V_2) with any $\mathcal{H} \in \mathcal{C}$ results in a protocol $({P_2}^{FS}, {V_2}^{FS})$ that is insecure against some adversary in $\mathcal{C}$.
3. If $\exists \mathcal{G} \in \mathcal{C}$ that $\neg\exists$strong-IMPERSONATOR and $\forall \mathcal{G}_1, \mathcal{G}_2 \in \mathcal{C}$ that $\exists$IMPERSONATOR, then with respect to this $\mathcal{G}$,
   - (P_3, V_3) is a 3-round public coin argument for empty language in $\mathcal{C}$ secure against any adversary in $\mathcal{C}$.
   - Applying the Fiat-Shamir transformation on (P_3, V_3) with any $\mathcal{H} \in \mathcal{C}$ results in a protocol $({P_3}^{FS}, {V_3}^{FS})$ that is insecure against some adversary in $\mathcal{C}$.

Acknowledgements

We thank Ran Canetti for reading earlier versions of this paper and giving helpful comments. We also thank Leonid Reyzin for his help in finding the private-coin version of Goldwasser-Kalai'03.

References

[Ajt96] Miklós Ajtai. "Generating Hard Instances of Lattice Problems (Extended Abstract)". In: *STOC*. 1996, pp. 99–108 (cit. on pp. 5, 57).

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. "Cryptography in NC^0". In: *SIAM J. Comput.* 36.4 (2006), pp. 845–888 (cit. on pp. 3, 54).

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach.* Cambridge University Press, 2009 (cit. on p. 11).

[Bar01] Boaz Barak. "How to Go Beyond the Black-Box Simulation Barrier". In: *FOCS*. IEEE Computer Society, 2001, pp. 106–115 (cit. on p. 2).

[BBHMR19] James Bartusek, Liron Bronfman, Justin Holmgren, Fermi Ma, and Ron D. Rothblum. "On the (In)security of Kilian-Based SNARGs". In: *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part II*. Ed. by Dennis Hofheinz and Alon Rosen. Vol. 11892. Lecture Notes in Computer Science. Springer, 2019, pp. 522–551. DOI: 10.1007/978-3-030-36033-7_20. URL: https://doi.org/10.1007/978-3-030-36033-7%5C_20 (cit. on p. 6).

[BKP18] Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. "Multi-collision resistance: a paradigm for keyless hash functions". In: *STOC*. ACM, 2018, pp. 671–684 (cit. on p. 2).

[BKM20] Zvika Brakerski, Venkata Koppula, and Tamer Mour. "NIZK from LPN and Trapdoor Hash via Correlation Intractability for Approximable Relations". In: *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*. Ed. by Daniele Micciancio and Thomas Ristenpart. Vol. 12172. Lecture Notes in Computer Science. Springer, 2020, pp. 738–767. DOI: 10.1007/978-3-030-56877-1_26. URL: https://doi.org/10.1007/978-3-030-56877-1%5C_26 (cit. on p. 2).

[CG18] Matteo Campanelli and Rosario Gennaro. "Fine-Grained Secure Computation". In: *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*. Ed. by Amos Beimel and Stefan Dziembowski. Vol. 11240. Lecture Notes in Computer Science. Springer, 2018, pp. 66–97. DOI: 10.1007/978-3-030-03810-6_3. URL: https://doi.org/10.1007/978-3-030-03810-6%5C_3 (cit. on p. 3).

[Can+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: *STOC*. ACM, 2019, pp. 1082–1090 (cit. on pp. 2, 6).

[CCR16] Ran Canetti, Yilei Chen, and Leonid Reyzin. "On the Correlation Intractability of Obfuscated Pseudorandom Functions". In: *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*. Ed. by Eyal Kushilevitz and Tal Malkin. Vol. 9562. Lecture Notes in Computer Science. Springer, 2016, pp. 389–415. DOI: 10.1007/978-3-662-49096-9_17. URL: https://doi.org/10.1007/978-3-662-49096-9%5C_17 (cit. on p. 1).

[CCRR18] Ran Canetti, Yilei Chen, Leonid Reyzin, and Ron D. Rothblum. "Fiat-Shamir and Correlation Intractability from Strong KDM-Secure Encryption". In: *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10820. Lecture Notes in Computer Science. Springer, 2018, pp. 91–122. DOI: 10.1007/978-3-319-78381-9_4. URL: https://doi.org/10.1007/978-3-319-78381-9%5C_4 (cit. on pp. 1, 4).

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. "The random oracle methodology, revisited". In: *J. ACM* 51.4 (2004), pp. 557–594 (cit. on p. 8).

[CWY23] Lijie Chen, Ryan Williams, and Tianqi Yang. "Black-Box Constructive Proofs Are Unavoidable". In: *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*. Ed. by Yael Tauman Kalai. Vol. 251. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, 35:1–35:24. ISBN: 978-3-95977-263-1. DOI: 10.4230/LIPIcs.ITCS.2023.35. URL: https://drops.dagstuhl.de/opus/volltexte/2023/17538 (cit. on pp. 32, 58, 59).

[CLZ22] Yilei Chen, Qipeng Liu, and Mark Zhandry. "Quantum Algorithms for Variants of Average-Case Lattice Problems via Filtering". In: *EUROCRYPT (3)*. Vol. 13277. Lecture Notes in Computer Science. Springer, 2022, pp. 372–401 (cit. on p. 57).

[CLMQ21] Yilei Chen, Alex Lombardi, Fermi Ma, and Willy Quach. “Does Fiat-Shamir Require a Cryptographic Hash Function?” In: *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*. Ed. by Tal Malkin and Chris Peikert. Vol. 12828. Lecture Notes in Computer Science. Springer, 2021, pp. 334–363. DOI: 10.1007/978-3-030-84259-8_12. URL: https://doi.org/10.1007/978-3-030-84259-8%5C_12 (cit. on p. 2).

[CGJJZ23] Arka Rai Choudhuri, Sanjam Garg, Abhishek Jain, Zhengzhong Jin, and Jiaheng Zhang. “Correlation Intractability and SNARGs from Sub-exponential DDH”. In: *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part IV*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14084. Lecture Notes in Computer Science. Springer, 2023, pp. 635–668. DOI: 10.1007/978-3-031-38551-3_20. URL: https://doi.org/10.1007/978-3-031-38551-3%5C_20 (cit. on p. 2).

[Cho+19] Arka Rai Choudhuri, Pavel Hubácek, Chethan Kamath, Krzysztof Pietrzak, Alon Rosen, and Guy N. Rothblum. “Finding a Nash equilibrium is no easier than breaking Fiat-Shamir”. In: *STOC*. ACM, 2019, pp. 1103–1114 (cit. on p. 2).

[CJJ21a] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. “Non-interactive Batch Arguments for NP from Standard Assumptions”. In: *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*. Ed. by Tal Malkin and Chris Peikert. Vol. 12828. Lecture Notes in Computer Science. Springer, 2021, pp. 394–423. DOI: 10.1007/978-3-030-84259-8_14. URL: https://doi.org/10.1007/978-3-030-84259-8%5C_14 (cit. on p. 2).

[CJJ21b] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. “SNARGs for P from LWE”. In: *FOCS*. IEEE, 2021, pp. 68–79 (cit. on p. 2).

[DVV16] Akshay Degwekar, Vinod Vaikuntanathan, and Prashant Nalini Vasudevan. “Fine-Grained Cryptography”. In: *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9816. Lecture Notes in Computer Science. Springer, 2016, pp. 533–562. DOI: 10.1007/978-3-662-53015-3_19. URL: https://doi.org/10.1007/978-3-662-53015-3%5C_19 (cit. on pp. 3, 4, 5, 19, 20, 55).

[DNRS99] Cynthia Dwork, Moni Naor, Omer Reingold, and Larry J. Stockmeyer. “Magic Functions”. In: *FOCS*. IEEE Computer Society, 1999, pp. 523–534 (cit. on p. 2).

[EWT21] Shohei Egashira, Yuyu Wang, and Keisuke Tanaka. “Fine-Grained Cryptography Revisited”. In: *J. Cryptol.* 34.3 (2021), p. 23. DOI: 10.1007/s00145-021-09390-3. URL: https://doi.org/10.1007/s00145-021-09390-3 (cit. on pp. 3, 7, 21).

[FS86] Amos Fiat and Adi Shamir. “How to Prove Yourself: Practical Solutions to Identification and Signature Problems”. In: *Advances in Cryptology - CRYPTO ’86, Santa Barbara, California, USA, 1986, Proceedings*. Ed. by Andrew M. Odlyzko. Vol. 263. Lecture Notes in Computer Science. Springer, 1986, pp. 186–194. DOI: 10.1007/3-540-47721-7_12. URL: https://doi.org/10.1007/3-540-47721-7%5C_12 (cit. on p. 1).

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. "Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based". In: *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. Lecture Notes in Computer Science. Springer, 2013, pp. 75–92. DOI: 10.1007/978-3-642-40041-4_5. URL: https://doi.org/10.1007/978-3-642-40041-4%5C_5 (cit. on pp. 4, 6).

[GW11] Craig Gentry and Daniel Wichs. "Separating succinct non-interactive arguments from all falsifiable assumptions". In: *STOC*. ACM, 2011, pp. 99–108 (cit. on p. 6).

[GK03] Shafi Goldwasser and Yael Tauman Kalai. "On the (In)security of the Fiat-Shamir Paradigm". In: *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*. IEEE Computer Society, 2003, pp. 102–113. DOI: 10.1109/SFCS.2003.1238185. URL: https://doi.org/10.1109/SFCS.2003.1238185 (cit. on pp. 2, 3, 4, 5, 7, 9, 10, 26, 29, 31).

[GK] Shafi Goldwasser and Yael Tauman Kalai. "On the (In)security of the Fiat-Shamir Paradigm". URL: https://web.archive.org/web/20060823172040id_/http://www.mit.edu:80/~tauman/fiatshamirlong.pdf (cit. on p. 10).

[HV06] Alexander Healy and Emanuele Viola. "Constant-Depth Circuits for Arithmetic in Finite Fields of Characteristic Two". In: *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006, Proceedings*. Ed. by Bruno Durand and Wolfgang Thomas. Vol. 3884. Lecture Notes in Computer Science. Springer, 2006, pp. 672–683. DOI: 10.1007/11672142_55. URL: https://doi.org/10.1007/11672142%5C_55 (cit. on p. 59).

[HL18] Justin Holmgren and Alex Lombardi. "Cryptographic Hashing from Strong One-Way Functions (Or: One-Way Product Functions and Their Applications)". In: *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*. Ed. by Mikkel Thorup. IEEE Computer Society, 2018, pp. 850–858. DOI: 10.1109/FOCS.2018.00085. URL: https://doi.org/10.1109/FOCS.2018.00085 (cit. on p. 2).

[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: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 750–760. DOI: 10.1145/3406325.3451116. URL: https://doi.org/10.1145/3406325.3451116 (cit. on p. 2).

[HJKS22] James Hulett, Ruta Jawale, Dakshita Khurana, and Akshayaram Srinivasan. "SNARGs for P from Sub-exponential DDH and QR". In: *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II*. Ed. by Orr Dunkelman and Stefan Dziembowski. Vol. 13276. Lecture Notes in Computer Science. Springer, 2022, pp. 520–549. DOI: 10.1007/978-3-031-07085-3_18. URL: https://doi.org/10.1007/978-3-031-07085-3%5C_18 (cit. on p. 2).

[IK00] Yuval Ishai and Eyal Kushilevitz. “Randomizing Polynomials: A New Representation with Applications to Round-Efficient Secure Computation”. In: *FOCS*. IEEE Computer Society, 2000, pp. 294–304 (cit. on p. 54).

[JJ21] Abhishek Jain and Zhengzhong Jin. “Non-interactive Zero Knowledge from Subexponential DDH”. In: *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*. Ed. by Anne Canteaut and François-Xavier Standaert. Vol. 12696. Lecture Notes in Computer Science. Springer, 2021, pp. 3–32. DOI: 10.1007/978-3-030-77870-5_1. URL: https://doi.org/10.1007/978-3-030-77870-5%5C_1 (cit. on p. 2).

[JKKZ21] Ruta Jawale, Yael Tauman Kalai, Dakshita Khurana, and Rachel Yun Zhang. “SNARGs for bounded depth computations and PPAD hardness from sub-exponential LWE”. In: *STOC ’21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 708–721. DOI: 10.1145/3406325.3451055. URL: https://doi.org/10.1145/3406325.3451055 (cit. on p. 2).

[KLV23] Yael Tauman Kalai, Alex Lombardi, and Vinod Vaikuntanathan. “SNARGs and PPAD Hardness from the Decisional Diffie-Hellman Assumption”. In: *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part II*. Ed. by Carmit Hazay and Martijn Stam. Vol. 14005. Lecture Notes in Computer Science. Springer, 2023, pp. 470–498. DOI: 10.1007/978-3-031-30617-4_16. URL: https://doi.org/10.1007/978-3-031-30617-4%5C_16 (cit. on p. 2).

[KRR17] Yael Tauman Kalai, Guy N. Rothblum, and Ron D. Rothblum. “From Obfuscation to the Security of Fiat-Shamir for Proofs”. In: *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*. Ed. by Jonathan Katz and Hovav Shacham. Vol. 10402. Lecture Notes in Computer Science. Springer, 2017, pp. 224–251. DOI: 10.1007/978-3-319-63715-0_8. URL: https://doi.org/10.1007/978-3-319-63715-0%5C_8 (cit. on p. 1).

[KVZ21] Yael Tauman Kalai, Vinod Vaikuntanathan, and Rachel Yun Zhang. “Somewhere Statistical Soundness, Post-Quantum Security, and SNARGs”. In: *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part I*. Ed. by Kobbi Nissim and Brent Waters. Vol. 13042. Lecture Notes in Computer Science. Springer, 2021, pp. 330–368. DOI: 10.1007/978-3-030-90459-3_12. URL: https://doi.org/10.1007/978-3-030-90459-3%5C_12 (cit. on p. 2).

[Kil92] Joe Kilian. “A Note on Efficient Zero-Knowledge Proofs and Arguments (Extended Abstract)”. In: *STOC*. ACM, 1992, pp. 723–732 (cit. on pp. 6, 9, 31, 32).

[Kiy22] Susumu Kiyoshima. “Public-Coin 3-Round Zero-Knowledge from Learning with Errors and Keyless Multi-Collision-Resistant Hash”. In: *CRYPTO (1)*. Vol. 13507. Lecture Notes in Computer Science. Springer, 2022, pp. 444–473 (cit. on p. 2).

[LV22] Alex Lombardi and Vinod Vaikuntanathan. “Correlation-Intractable Hash Functions via Shift-Hiding”. In: *ITCS*. Vol. 215. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 102:1–102:16 (cit. on p. 2).

[Mic00] Silvio Micali. “Computationally Sound Proofs”. In: *SIAM J. Comput.* 30.4 (Oct. 2000), pp. 1253–1298. ISSN: 0097-5397. DOI: 10.1137/S0097539795284959. URL: https://doi.org/10.1137/S0097539795284959 (cit. on p. 6).

[MP13] Daniele Micciancio and Chris Peikert. “Hardness of SIS and LWE with small parameters”. In: *Advances in Cryptology–CRYPTO 2013*. Springer, 2013, pp. 21–39 (cit. on p. 57).

[MR07] Daniele Micciancio and Oded Regev. “Worst-case to average-case reductions based on Gaussian measure”. In: *SIAM Journal on Computing* 37.1 (2007), pp. 267–302 (cit. on p. 57).

[Par21] Orr Paradise. “Smooth and Strong PCPs”. In: *Comput. Complex.* 30.1 (2021), p. 1. DOI: 10.1007/s00037-020-00199-3. URL: https://doi.org/10.1007/s00037-020-00199-3 (cit. on pp. 32, 58).

[PS19] Chris Peikert and Sina Shiehian. “Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors”. In: *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Vol. 11692. Lecture Notes in Computer Science. Springer, 2019, pp. 89–114. DOI: 10.1007/978-3-030-26948-7_4. URL: https://doi.org/10.1007/978-3-030-26948-7%5C_4 (cit. on pp. 2, 3, 4, 6, 7, 15, 21, 22, 23).

[PS96] David Pointcheval and Jacques Stern. “Security Proofs for Signature Schemes”. In: *Advances in Cryptology - EUROCRYPT ’96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*. Ed. by Ueli M. Maurer. Vol. 1070. Lecture Notes in Computer Science. Springer, 1996, pp. 387–398. DOI: 10.1007/3-540-68339-9_33. URL: https://doi.org/10.1007/3-540-68339-9%5C_33 (cit. on p. 1).

[WP22] Yuyu Wang and Jiaxin Pan. “Non-Interactive Zero-Knowledge Proofs with Fine-Grained Security”. In: *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II*. Ed. by Orr Dunkelman and Stefan Dziembowski. Vol. 13276. Lecture Notes in Computer Science. Springer, 2022, pp. 305–335. DOI: 10.1007/978-3-031-07085-3_11. URL: https://doi.org/10.1007/978-3-031-07085-3%5C_11 (cit. on pp. 3, 4, 6, 20).

A Detailed Proofs of Almost-Everywhere Security

To begin with, we state several properties of $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$ that is implicit in [IK00] or [AIK06].

Lemma A.1. For any function family $\{f_\lambda\} \in \oplus\mathsf{L}/\mathsf{poly}$, there exists a randomized circuit family $\{\mathcal{F}_\lambda\} \in \mathsf{AC}^0[2]$ and a polynomial $n = n(\lambda)$ such that for any $x \in \{0,1\}^\lambda$,

- $\{\mathcal{F}_\lambda(x)\} \equiv \mathsf{ZeroSamp}(n(\lambda))$, if $f_\lambda(x) = 0$;

- $\{\mathcal{F}_\lambda(x)\} \equiv \mathsf{OneSamp}(n(\lambda))$, if $f_\lambda(x) = 1$. $\diamondsuit$

Lemma A.2. The following statements about matrix $A \in \mathbb{Z}_2^{n\times n}$ are equivalent, and they are still equivalent if all get modified by the texts in brackets:

- A is the output of $\mathsf{ZeroSamp}$ ($\mathsf{OneSamp}$) using any randomness.
- A is in the form of
$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n-1} & a_{1,n} \\ 1 & a_{2,2} & a_{2,3} & \cdots & a_{2,n-1} & a_{2,n} \\ & 1 & a_{3,3} & \cdots & a_{3,n-1} & a_{3,n} \\ & & 1 & \cdots & a_{4,n-1} & a_{4,n} \\ & & & \ddots & \vdots & \vdots \\ & & & & 1 & a_{n,n} \end{pmatrix},$$
and $\det A = 0$ ($\det A = 1$).
- The two distributions are equivalent:
$$\{L(r^{(1)})AR(r^{(2)}) : r^{(1)} \leftarrow \{0,1\}^*, r^{(2)} \leftarrow \{0,1\}^*\} \equiv \mathsf{ZeroSamp}(n)$$
$$\left(\{L(r^{(1)})AR(r^{(2)}) : r^{(1)} \leftarrow \{0,1\}^*, r^{(2)} \leftarrow \{0,1\}^*\} \equiv \mathsf{OneSamp}(n)\right). \quad \diamondsuit$$

With all these properties, we restate lemma 4.3 of [DVV16] after extending the adversary class.

Lemma A.3 (Minor extension of lemma 4.3 of [DVV16]). For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, assuming Assumption 4.8 holds for $\mathcal{C}$, there is a polynomial $n = n(\lambda)$ and a negligible function $\mathsf{negl}(\cdot)$ such that for any family $\mathcal{F} = \{f_\lambda\}$ in $\mathcal{C}$, for infinitely many λ,

$$\left| \Pr_{M \leftarrow \mathsf{ZeroSamp}(n(\lambda))}[f_\lambda(M) = 1] - \Pr_{M \leftarrow \mathsf{OneSamp}(n(\lambda))}[f_\lambda(M) = 1] \right| < \mathsf{negl}(\lambda). \quad \diamondsuit$$

The difference between this lemma and lemma 4.10 is the existence of the polynomial $n = n(\lambda)$. In fact this lemma trivially implies lemma 4.10, as for any infinitely long sequence $\lambda_1, \lambda_2, \ldots \in \mathbb{N}$, $n(\lambda_1), n(\lambda_2), \ldots$ is also an infinitely long sequence.

Proof. The idea is by lemma A.1, any circuit that distinguishes $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$ can be used to compute any language in $\oplus\mathsf{L}/\mathsf{poly}$.

Assume for any polynomial $n = n(\lambda)$, there exists a constant $d > 0$ and an adversary $\mathcal{A} = \{\mathcal{A}_\lambda\} \in \mathcal{C}$ s.t. for all but finitely many λ, $\mathcal{A}_\lambda$ distinguishes $\mathsf{ZeroSamp}(n(\lambda))$ and $\mathsf{OneSamp}(n(\lambda))$ by probability λ^{-d}. By lemma A.1, we can directly link the output of the circuit $\mathcal{F}_\lambda$ (that computes $\mathsf{ZeroSamp}$ or $\mathsf{OneSamp}$) to $\mathcal{A}_\lambda$ to compute any language in $\oplus\mathsf{L}/\mathsf{poly}$. As $\Pr[\mathcal{A}_\lambda(\mathcal{F}_\lambda(x)) = 1 | f(x) = 0]$ and $\Pr[\mathcal{A}_\lambda(\mathcal{F}_\lambda(x)) = 1 | f(x) = 1]$ only differs by λ^{-d}, we need to do error reduction to enlarge this gap to $1 - \mathsf{poly}(\lambda)^{-1}$.

Let $p = \Pr[\mathcal{A}_\lambda(\mathsf{ZeroSamp}) = 1]$ and $q = \Pr[\mathcal{A}_\lambda(\mathsf{OneSamp}) = 1]$, then the error reduction is done by taking many samples and checking whether $\mathcal{A}_\lambda$ outputs 1 on more than $\frac{p+q}{2}$ of the samples. If $\mathcal{C}$ can compute majority function, then the entire reduction is done in $\mathcal{C}$; otherwise, for any $c > 0$, we can complete the reduction in $\widetilde{\mathsf{Sum}}_{1/\lambda^c} \circ \mathcal{C}$, as taking polynomially many samples allows us to approximate $\Pr[\mathcal{A}_\lambda(\mathcal{F}_\lambda) = 1]$ by error λ^{-c} with exponentially small failure probability for any $c > 0$. $\square$

Although we can prove almost everywhere security under the strengthened assumption 4.11, we cannot get rid of the polynomial $n = n(\lambda)$ by directly using the same proof idea.

Lemma A.4. For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, assuming Assumption 4.11 holds for $\mathcal{C}$, there is a polynomial $n = n(\lambda)$ and a negligible function $\mathsf{negl}(\cdot)$ such that for any family $\mathcal{F} = \{f_\lambda\}$ in $\mathcal{C}$, for all but infinitely many λ,

$$\left| \Pr_{M \leftarrow \mathsf{ZeroSamp}(n(\lambda))}[f_\lambda(M) = 1] - \Pr_{M \leftarrow \mathsf{OneSamp}(n(\lambda))}[f_\lambda(M) = 1] \right| < \mathsf{negl}(\lambda). \qquad \diamondsuit$$

In order to get rid of the polynomial $n = n(\lambda)$, we use a simple observation that for any $n' > n$ and any circuit that distinguishes $\mathsf{ZeroSamp}(n')$ and $\mathsf{OneSamp}(n')$, we can use the same circuit to distinguish $\mathsf{ZeroSamp}(n)$ and $\mathsf{OneSamp}(n)$.

Lemma A.5. There exists a constant c, such that for any $n' > n$ and circuit C' with circuit size s and depth d that distinguishes $\mathsf{ZeroSamp}(n')$ and $\mathsf{OneSamp}(n')$ by some probability, there exists a circuit C that distinguishes $\mathsf{ZeroSamp}(n)$ and $\mathsf{OneSamp}(n)$ with the same probability, with circuit size at most cs and depth at most cd. $\diamondsuit$

Proof. For any matrix $M \in \mathbb{Z}_2^{n \times n}$ obtained from $\mathsf{ZeroSamp}(n)$ (or $\mathsf{OneSamp}(n)$), we define

$$M' = \begin{bmatrix} & & & 0 & 0 & \cdots & 0 & 0 \\ & M & & 0 & 0 & \cdots & 0 & 0 \\ & & & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \implies \det M' = \det M.$$

Therefore, by lemma A.2, since M' has the correct form and $\det M' = \det M$, M' is also an output of $\mathsf{ZeroSamp}(n')$ (or $\mathsf{OneSamp}(n')$); furthermore, by the same lemma, take $r^{(1)} \leftarrow \{0,1\}^*, r^{(2)} \leftarrow \{0,1\}^*$, the distribution of $\{L(r^{(1)})M'R(r^{(2)})\}$ is exactly $\mathsf{ZeroSamp}(n')$ (or $\mathsf{OneSamp}(n')$).

The circuit C is constructed by expanding M to M' and computing $L(r^{(1)})M'R(r^{(2)})$ before invoking C'. The overhead is minor. $\square$

Proof of lemma 4.12. If we have an adversary $\{\mathcal{A}_n\} \in \mathcal{C}$ that distinguishes $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$ with non-negligible probability for infinitely many n, then for any polynomial $p(n)$, we can construct $\{\mathcal{B}_n\} \in \mathcal{C}$ to break lemma A.4, i.e. $\exists$ a polynomial $q(n)$, for infinitely many n,

$$\left| \Pr_{M \leftarrow \mathsf{ZeroSamp}(p(n))}[\mathcal{B}_n(M) = 1] - \Pr_{M \leftarrow \mathsf{OneSamp}(p(n))}[\mathcal{B}_n(M) = 1] \right| > \frac{1}{q(n)}.$$

By lemma A.5, if $\{\mathcal{A}_n\}$ distinguishes $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$ for $\{n_i\}_{i \in \mathbb{N}}$, we construct $\{\mathcal{B}_n\}$ as:

- If $n = \max\{n : p(n) \leq n_i\}$ for some i, $\mathcal{B}_n$ is the distinguisher constructed from $\mathcal{A}_{n_i}$.
- Otherwise, $\mathcal{B}_n$ outputs 0.

Then $\{\mathcal{B}_n\} \in \mathcal{C}$ and we can easily see that for all $n = \max\{n : p(n) \leq n_i\}$, $\mathcal{B}_n$ succeeds by non-negligible probability. $\square$

B Low Complexity CRHF

In this section we propose candidates for collision resistant hash functions with arbitrary polynomial shrinkage computable in $\mathsf{AC}^0[c]$ for some constant $c \geq 2$ against p.p.t. adversaries.

Definition B.1 (SIS^∞). For any $n, m, q, B \in \mathbb{N}^+$, $q \geq 4$ define the short integer solution problem $\mathsf{SIS}^\infty_{n,m,q,B}$ as follows: Given $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, find a non-zero vector $\mathbf{x} \in \{-B, -B+1, ..., B-1, B\}^m$ such that

$$\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q.$$

We are mostly interested in the case where $B = 1$ and $q = 2^k$ for some constant $k \geq 2$. When $q = 2^k$ for some constant $k \geq 2$, the function $f : \{0,1\}^m \to \mathbb{Z}_q^n, \mathbf{x} \mapsto \mathbf{A}\cdot\mathbf{x} \bmod q$ is computable in $\mathsf{AC}^0[2]$. Note that when $m \geq (n+1)^k$, there is a polynomial time classical algorithm that solves $\mathsf{SIS}^\infty_{n,m,2^k,1}$, see [CLZ22, Appendix]. However, when $m < n^k$, we do not know any polynomial time algorithm. We thus make the following conjecture:

Conjecture B.2. For any constant $\varepsilon > 0$, for any $2 \leq k \in O(1)$, $q = 2^k$, for any $n, m \in \mathbb{N}^+$ such that $m \leq n^{k-\varepsilon}$. There is no $\mathsf{poly}(n)$ time algorithm that solves $\mathsf{SIS}^\infty_{n,m,q,1}$. $\diamondsuit$

In fact, as long as for some constant $\delta > 0$, for some constant $k \geq 2$, $\mathsf{SIS}^\infty_{n,n^{1+\delta},2^k,1}$ is hard, then there exists CRHFs with arbitrary polynomial shrinkage computable in $\mathsf{AC}^0[2]$, by stacking constant levels of CRHFs with compression rates of $n^{1+\delta}$ to n.

Let us remark that the hardness of the variant of $\mathsf{SIS}^\infty_{n,m,q,1}$ we conjectured in Conjecture B.2 is not known to be as hard as worst-case lattice problems, as oppose to the "standard SIS" problem (cf. Def 4.1) where the solution is bounded in its ℓ_2 norm, for which we know worst-case to average case reductions [Ajt96; MR07; MP13]. We recall the precise statement from [MP13].

Lemma B.3 (Theorem 3.8 of [MP13]). Let $n, m \in \mathbb{N}^+$, $S = \{\mathbf{z} \in \mathbb{Z}^m \setminus \{\mathbf{0}\} \mid \|\mathbf{z}\| < \beta \wedge \|\mathbf{z}\|_\infty < \beta_\infty\}$ for some real $\beta \geq \beta_\infty > 0$, and $q \geq \beta \cdot n^{\Omega(1)}$ be an integer modulus with at most $\mathsf{poly}(n)$ integer divisors less than β_∞. Then for some $\gamma = \max\{1, \beta\beta_\infty/q\} \cdot O(\beta\sqrt{n})$, there is an efficient quantum reduction from $\mathsf{SIVP}_{\gamma\cdot\omega_n}$ on n-dimensional lattices to solving $\mathsf{SIS}_{n,m,q}$ within set S with non-negligible probability. $\diamondsuit$

We observe that to apply Lemma B.3, we must use $q \geq \sqrt{n}$, but then we do not know how to compute the SIS function in $\mathsf{AC}^0[c]$ for any constant c.

Although not used in our paper, we would also like to make a conjecture about the hardness of $\mathsf{SIS}^\infty_{n,m,q,1}$ for possibly non-power-of-two moduli:

Conjecture B.4. For any constant $\varepsilon \in (0, 0.99)$, for any $4 \leq q \in O(1)$, for any $n, m \in \mathbb{N}^+$ such that $m \in O(n^{1+\varepsilon})$. There is no $\mathsf{poly}(n)$ time algorithm that solves $\mathsf{SIS}^\infty_{n,m,q,1}$. $\diamondsuit$

Note that if Conj. B.4 is true, then we have a CRHF family with arbitrary polynomial compression computable in $\mathsf{AC}^0[q]$. The best classical and quantum algorithms against Conj. B.4 requires m to be at least n^2. Readers are referred to [CLZ22] for the details.

C $\mathsf{AC}^0[2]$-computable and $\mathsf{AC}^0[2]$-proof-of-knowledge PCP

In this section, we give a short sketch of how the PCP proof system in Theorem 6.5 is constructed. We begin by recalling the theorem.

Theorem 6.5. For the central relation $\mathcal{R}$, there exists a PCP system (P_{pcp}, V_{pcp}) where V_{pcp} is a probabilistic verifier in $\mathsf{AC}^0[2]$ for $\mathcal{R}$ with randomness complexity $O(\log n)$ and query complexity $O(1)$. while P_{pcp} is an $\mathsf{AC}^0[2]$ circuit. This system satisfies:

- **Completeness**: For all $(x, w) \in \mathcal{R}$,
$$\Pr\left[V_{pcp}^{\pi}(x) = 1 \mid \pi \leftarrow P_{pcp}(x, w)\right] = 1.$$
This implies that there exists a polynomial $p(\cdot)$ such that the proof π generated is of length $p(|x| + |w|)$.
- **Soundness**: For all x such that there does not exists w that $(x, w) \in \mathcal{R}$, for any proof π^*,
$$\Pr\left[V_{pcp}^{\pi^*}(x) = 1\right] < \frac{1}{2},$$
where the randomness is over the internal randomness of V_{pcp}.
- **Knowledge extraction**: there exists an extractor E_{pcp} in $\mathsf{AC}^0[2]$ such that for any input x and proof $\tilde{\pi}$ if
$$\Pr\left[V_{pcp}^{\tilde{\pi}}(x) = 1\right] \geq 1 - \frac{1}{\log^5 n},$$
then
$$\Pr\left[(x, w) \in \mathcal{R} \mid w \leftarrow E_{pcp}(x, \tilde{\pi})\right] = 1. \qquad \diamondsuit$$

Our PCP system follows from the canonical, smooth, and strong construction by Paradise [Par21]. Chen, Williams, and Yang [CWY23] additionally proved that it is $\mathsf{AC}^0[2]$-locally decodable. Note that knowledge extraction in $\mathsf{AC}^0[2]$ follows directly from strongness and $\mathsf{AC}^0[2]$-local-decodability. So we only show that there is a prover in $\mathsf{AC}^0[2]$ outputting the PCP proof π given a witness w to the original relation $\mathcal{R}$.

Below, we give a brief sketch of why the prover can be implemented in $\mathsf{AC}^0[2]$, assuming the familiarity of [CWY23]. In their terminology, we show that if the good predicate $V(x, y)$ (which is $\mathcal{R}$ in our paper) is an $\mathsf{AC}^0[2]$ circuit, then the canonical mapping Π_V (which is P_{pcp} in our paper) can also be computed by an $\mathsf{AC}^0[2]$ circuit.

Proof sketch of Theorem 6.5 (AC^0[2]-computability). Suppose that the relation $\mathcal{R}$ is computable by an $\mathsf{AC}^0[2]$ circuit of wire complexity s.

Our first step is to convert the relation $\mathcal{R}$ by a 3-CNF formula $\varphi(x, w')$ of size $O(s)$, such that for each $x \in \{0, 1\}^n$,
$$\exists w, (x, w) \in \mathcal{R} \iff \exists w', \varphi(x, w'),$$
where a satisfying w' can be computed by $\mathsf{AC}^0[2]$ circuits given x and w where $(x, w) \in \mathcal{R}$.

For $\mathsf{P/poly}$ circuits, this is the standard reduction from CktSAT to $\mathsf{3SAT}$, but we need to be very careful for $\mathsf{AC}^0[2]$ circuits to ensure that w' can be computed in $\mathsf{AC}^0[2]$. To do so, we define ℓ new variables for each gate of fan-in ℓ. Suppose, without loss of generality, that the gate is $H(g_1, \ldots, g_\ell) = g_1 \wedge g_2 \wedge \cdots \wedge g_\ell$, where $g_1, \ldots, g_\ell$ are outputs of lower gates (with possible negations). Then we define
$$v_i = \bigwedge_{j=1}^{i} g_j.$$

Then the 3-CNF contains clauses that describe $v_1 = g_1$ and $v_i = v_{i-1} \wedge g_i$. We then use v_ℓ as the output as the gate H for further uses. Note that for H being an $\oplus$ gate, we can still use a constant number of clauses to describe $v_i = v_{i-1} \oplus g_i$. In addition, we have a clause asserting that the output gate of the circuit $\mathcal{R}$ evaluates to 1. This results in a 3-CNF of size $O(s)$.

We still need to show that each v_i can be computed in $\mathsf{AC}^0[2]$. This is true since we can evaluate each v_i by a single gate given $g_1, \dots, g_\ell$. For example, if H is an $\wedge$-gate, then $v_i = g_1 \wedge g_2 \wedge \cdots \wedge g_i$, which is a single $\wedge$-gate. Since the original circuit for $\mathcal{R}$ is of constant depth, we can also compute each v_i in $\mathsf{AC}^0[2]$.

We then show that the canonical proof of the PCP in [CWY23] is computable in $\mathsf{AC}^0[2]$ if the predicate is a 3-CNF. Note that the PCP in [CWY23] is constructed by composing for several times a robust PCP of proximity (PCPP) from low-degree extension, with a Hadamard PCP at the end. The proof of each part is repeated for a polynomial number of times to ensure smoothness. The proof of the Hadamard PCP is trivially computable in $\mathsf{AC}^0[2]$ since each bit is just an inner-product[7].

For the PCP based on low-degree extension, the proof π is constructed by first computing an low-degree extension of w over a field $\mathbb{F}$ with $|\mathbb{F}| = 2^{2 \cdot 3^\ell}$ for some $\ell \in \mathbb{N}$ and $|\mathbb{F}| \leq \mathsf{polylog}(s)$, then encoding each value in the field $\mathbb{F}$ by a Boolean string of length $b = O(\log|\mathbb{F}|)$ by a constant-rate error-correcting code. The latter ECC can be easily computed by $\mathsf{AC}^0[2]$ circuits since $b = O(\log \log s)$, so the ECC encoding can be implemented by a look-up table of size $2^b = \mathsf{polylog}(s)$.

We now describe the low-degree extension in more details. Suppose, w.l.o.g., that $|w| = s$. Let $m = \lceil \log s / \log \log s \rceil$, and h be the smallest power of 2 such that $h^m \geq s$.[8]

We view the string w as a function $[h]^m \to \{0,1\}$ (we pad the string w with 0 when necessary). Then we identify $[h]$ with h distinct elements in $\mathbb{F}$ such that the 0 and 1 in $[h]$ are mapped to the 0 and 1 in $\mathbb{F}$, so w can be viewed as a partial function from $\mathbb{F}^m$ to $\mathbb{F}$. We then compute the unique polynomial $\widehat{A} : \mathbb{F}^m \to \mathbb{F}$ with degree at most $h-1$ in each variable, which agrees with w on all inputs from $[h]^m$. We will output (the ECC encoding of) the truth table of $\widehat{A}$ as the proof π, which is $\widehat{A}(x)$ for all $x \in \mathbb{F}^m$.

The translation of w to the function $[h]^m \to \{0,1\}$ can be easily done in $\mathsf{AC}^0[2]$ given h as a power of 2. We can also translate easily this into $\mathbb{F}^m$ since the mapping from $[h]$ to $\mathbb{F}$ can be arbitrary. The only hard part is the low-degree extension, where we need to evaluate a multi-variate polynomial interpolation. Fortunately, the points given are the product set $[h]^m$, so we can use the Lagrange interpolation on each variable respectively. In particular, suppose that $[h]$ is mapped to $a_0, a_1, \dots, a_{h-1}$ in $\mathbb{F}$ respectively, and by viewing w as a function $[h]^m \to \{0,1\}$, $w(y)$ is the function value of $y \in [h]^m$, then

$$\widehat{A}(x) = \sum_{y \in [h]^m} w(y) \prod_{i=1}^{m} \prod_{j \in [h], j \neq y_i} \frac{x_i - a_j}{a_{y_i} - a_j}.$$

The denominator does not depend on x, so it can be hardwired into the circuit. Each enumerator is an iterated multiplication of $m \cdot (h-1) \leq \mathsf{polylog}(s)$ terms, so it can be computed in $\mathsf{AC}^0[2]$ of $\mathsf{poly}(s)$ size by [HV06]. So the polynomial $\widehat{A}(x)$ can be evaluated by $\mathsf{AC}^0[2]$ circuits. □

[7]Technically speaking, it is not an inner-product since the PCP proof is the Hadamard code of the all constant-degree monomials of the witness w. However, this is still computable in $\mathsf{AC}^0[2]$ since all computations are on $\mathbb{F}_2$.

[8]Here we modify the construction in [CWY23] a little bit by taking h to be a power of 2. This is to ensure that any integer in $[s]$ can be easily encoded into $[h]^m$. This will not change the complexity too much since $h^m \leq 2^m \cdot s \leq s^2$.