

A Fiat-Shamir Transformation for Private-Coin Protocols

Shany Ben-David	Agni Datta	Eylon Yogev
shany.ben-david@biu.ac.il	agnidatta.org@gmail.com	eylon.yogev@biu.ac.il
Bar-Ilan University	University of Edinburgh	Bar-Ilan University

Abstract

The Fiat–Shamir transformation is a fundamental paradigm in cryptography for compiling interactive public-coin protocols into non-interactive arguments. While this transformation has been extensively studied for public-coin protocols, the setting of *private-coin* protocols, where the verifier’s messages depend on secret randomness, remains largely unexplored. To date, there are no candidate transformations for private-coin protocols, neither in the standard model nor in idealized settings.

In this work, we present a generic transformation that compiles any 3-message private-coin interactive proof into a non-interactive argument in the standard model, while preserving zero-knowledge. Our construction relies on sub-exponentially secure indistinguishability obfuscation, pseudorandom functions, and input-hiding obfuscation (IHO). As a corollary, this rules out the existence of 3-message weak zero-knowledge proofs with small enough soundness error.

We complement this positive result by showing that the reliance on IHO is inherent: any secure Fiat–Shamir transformation for general private-coin protocols necessarily implies the existence of IHO.

Finally, we rule out Fiat–Shamir transformations for multi-round private-coin protocols. Specifically, assuming the existence of collision-resistant hash functions, we prove that no such transformation exists in the multi-round setting.

Contents

1	Introduction	1
1.1	Our Results	2
1.2	Discussion: A Unified View of Fiat–Shamir	3
1.3	Fiat-Shamir without IHO	4
	Paper Organization and AI Disclosure	5
2	Technical Overview	6
2.1	Constructing Fiat–Shamir for Private-Coin Protocols	6
2.2	Warm-Up: Analysis for GNI	7
2.3	Analysis for Private-Coin Protocols	11
2.4	On The Necessity of Input-Hiding Obfuscation	15
2.5	Impossibility for Protocols Beyond 3 Messages	16
2.6	On The Existence of 3-Message Zero-Knowledge Proofs for NP	17
3	Preliminaries	19
3.1	Distributions and Sampling	19
3.2	Pseudo-Random Function	19
3.3	Indistinguishability Obfuscation	20
3.4	Input-Hiding Obfuscation	20
3.5	Interactive Proof Systems	21
4	A Fiat–Shamir Transformation for Private-Coin Protocols	23
4.1	The Transformation	23
4.2	Adaptive Soundness	24
4.3	Zero-Knowledge in the Honest-Verifier Setting	35
5	On The Necessity of Input-Hiding Obfuscation	39
6	Beyond 3-Message Protocols	47
7	On The Existence of 3-Message Zero-Knowledge Proofs for NP	49
8	Statistical Difference Problem	51
8.1	Protocol	51
8.2	Analysis	52
8.3	Statistical Zero-Knowledge in the Honest-Verifier Setting	59
9	A Fiat–Shamir Transformation for Public-Coin Protocols	61
9.1	Protocol	61
9.2	Adaptive Soundness	62
	Acknowledgments	67
	References	67

1 Introduction

The *Fiat-Shamir transformation*, introduced in the seminal work of Fiat and Shamir [FS87], is a fundamental paradigm in modern cryptography. It provides a generic method to compile interactive public-coin protocols into non-interactive arguments. This transformation plays a pivotal role in the construction of numerous cryptographic primitives, ranging from efficient digital signature schemes (e.g., Schnorr signatures [Sch91]) to succinct non-interactive arguments of knowledge (SNARKs) [Mic00; BCS16; CY21a; CY21b; ACY23; AHIV17; BBHR18; ZCF24; ACFY24; ACFY25; BCKP14], verifiable delay functions [BBBF18] and more.

Given its centrality, the security of the Fiat-Shamir transformation has been extensively analyzed. Initial analyses relied on the *random oracle model* (ROM) [BR93; PS96], treating the hash function as an ideal random function. While this heuristic provides strong efficiency benefits, it does not guarantee security when instantiated with a concrete hash function [CGH98; Bar01; GK03; KRS25; AY25]. Consequently, a long line of research has focused on establishing the security of the transformation in the *standard model*, typically relying on *correlation-intractable* hash functions (e.g., [CCR16; HL18; CCH+19; PS19; BKM20; JJ21; HLR21; CJJ21; HJKS22; KLV23]). However, most of these works successfully instantiate the transformation only for specific families of protocols. To date, the only constructions suitable for *all* public-coin protocols are those of Kalai et al. [KRR17], which rely on indistinguishability obfuscation and input-hiding obfuscation, and Canetti et al. [CCRR18], which rely on strong key-dependent message (KDM) security with optimal parameters.

Crucially, the Fiat-Shamir transformation is inherently designed for *public-coin* protocols, where the verifier’s messages consist solely of fresh random coins. In this setting, the transformation replaces the verifier’s random challenges with the output of a cryptographic hash function applied to the protocol’s partial transcript. Since the verifier simply outputs random bits, a random oracle (or a sufficiently complex hash function) serves as a natural proxy. However, for *private-coin* protocols, where the verifier may send messages that arbitrarily depend on secret randomness, no analogous transformation is known. To date, there is not even a candidate construction for a “private-coin Fiat-Shamir” transformation, neither in idealized models nor under standard cryptographic assumptions.

Private-coin protocols. Private-coin protocols form a rich and important class of interactive proof systems. In many settings, they offer distinct advantages over their public-coin counterparts, achieving significantly lower communication complexity and improved verifier efficiency.

A prominent example of a class of private-coin protocols is Statistical Zero-Knowledge (SZK). This class consists of languages that admit interactive proofs in which the verifier’s view can be efficiently simulated so as to be negligibly close—in statistical distance—to its view in a real execution.

SZK plays a central role in complexity theory and cryptography, as it captures forms of computational hardness that do not rely on unproven computational assumptions. SZK contains the well-known *Graph Non-Isomorphism* (GNI) [GMW86], and the canonical complete problem is the *Statistical Difference* problem (SD), shown to be SZK-complete by Sahai and Vadhan [SV00].

A major challenge is that the most natural and efficient protocols for these problems are *private-coin* protocols, where the verifier’s secret randomness is essential for soundness. While the celebrated result of Goldwasser and Sipser [GS86] proves that any private-coin protocol can be transformed into a public-coin protocol, this transformation incurs a significant blowup in the prover’s

computational complexity and communication overhead [AR21]. Consequently, developing an “interactive to non-interactive” transformation for private-coin protocols remains a major open goal, as standard Fiat–Shamir techniques are inapplicable in this setting. This state of affairs motivates a natural and fundamental question:

Is there a Fiat–Shamir transformation for private-coin protocols?

1.1 Our Results

In this work, we provide the first affirmative answer to this question. We present a general transformation for private-coin protocols, along with a specialized instantiation for SZK. We complement these results by showing the necessity of cryptographic primitives needed to apply the Fiat–Shamir transformation to private-coin protocols.

We begin by describing our most general transformation.

Fiat–Shamir for private-coin protocols. We introduce a generic transformation that compiles 3-message private-coin interactive proofs into non-interactive arguments in the standard model. Our construction relies on strong, yet standard assumptions in the plain model. We assume the existence of sub-exponentially secure *indistinguishability obfuscation* (iO), sub-exponentially secure *PRFs* (PRF), and *input-hiding obfuscation* (IHO). IHO is a notion of obfuscation guaranteeing that for a circuit C drawn from an evasive family, it is hard to find an input x such that $C(x) = 1$ given only the obfuscated circuit $\hat{C}$.

Theorem 1 (Informal). *Assume the existence of sub-exponentially secure pseudorandom functions, indistinguishability obfuscation and input-hiding obfuscation. Then, there is a Fiat–Shamir transformation for private-coin protocols with the following parameters:*

Any 3-message private-coin protocol with error ϵ , is compiled to a non-interactive scheme with (adaptive) soundness error $\epsilon^{\frac{1}{2}} \cdot 2^n \cdot \text{poly}(\lambda) + \text{negl}(\lambda)$, where n is a bound on the instance size and the first message of the protocol. The transformation also preserves zero-knowledge.

Any application of the Fiat–Shamir transformation inherently incurs a security loss, as the adversary has greater power than in the interactive setting. Recall that the soundness error ϵ can be exponentially reduced via *parallel repetition*. Crucially, by performing this repetition without replicating the first message (thereby keeping the length n constant), one can amplify any protocol, even those with constant error, to satisfy the conditions of our theorem.

Our theorem has tight parameters for adversaries that run in sub-exponential time. Nevertheless, constructing a Fiat–Shamir transformation for private-coin protocols with only *polynomial* security loss (against polynomial adversaries) remains an open problem.

IHO is inherent for Fiat–Shamir for private-coin. Our construction relies on iO and IHO, which is not too surprising. Indeed, even for public-coin protocols, the work of [KRR17] relies on the same assumptions (albeit with an IHO for a specific family of functions).

We prove a complementary lower bound showing that the reliance on input-hiding obfuscation is not an artifact of our proof technique but an inherent requirement. Specifically, we show that *any* secure Fiat–Shamir transformation for private-coin protocols implies the existence of secure IHO scheme for any family of evasive circuits. While we have established the necessity of IHO, it remains an open question whether iO is similarly inherent.

Theorem 2 (Informal). *If there exists a Fiat–Shamir transformation for private-coin 3-message protocols (even with subexponential security loss) and a secure puncturable PRF scheme, then there exists an input-hiding obfuscation scheme for any evasive circuit family.*

Recall that the work of [KRR17] used an IHO for a very specific family of evasive functions. In contrast, our theorem yields an IHO for any evasive circuit family. This broader applicability suggests that IHOs are unlikely to be inherent to the Fiat–Shamir transform for public-coin protocols.

Limits for multi-round protocols. We have discussed that our Fiat–Shamir transformation for private-coin protocols applies successfully to three-message protocols. A natural question is whether this approach can be extended to multi-round protocols. Unfortunately, we show that this is not the case. In particular, assuming the existence of collision-resistant hash (CRH) functions, we prove that no Fiat–Shamir transformation for private-coin protocols can exist for protocols that contain more than a single verifier message. Our result is established via a (contrived) counterexample: a protocol that cannot be rendered non-interactive without violating the assumed security of the CRH.

Theorem 3 (Informal). *Assuming the existence collision-resistant hash function, then there is no Fiat–Shamir transformation for private-coin protocols for 5-message protocols (or more).*

Impossibility of 3-message weak zero-knowledge. There is a well established connection between the Fiat–Shamir transformation and the impossibility of constructing certain zero-knowledge protocols with negligible soundness error [DNRS99; HT98; KRR17; FGJ18]. At a high level, these works show that if sufficiently strong zero-knowledge protocols exist—particularly constant-round protocols for languages outside of BPP—then the Fiat–Shamir paradigm cannot be secure when applied to them. In particular, [FGJ18] established a strong impossibility result ruling out three-message zero-knowledge protocols in the presence of non-uniform verifiers and distinguishers, under the assumptions of subexponentially secure indistinguishability obfuscation (iO) and input-hiding obfuscation (IHO).

Our main result (Theorem 1) complements this line of work by showing that, even for three-message protocols, the Fiat–Shamir transformation itself rules out *weak* zero-knowledge [DNRS99] (see also [BKP19; GO94]). Weak zero-knowledge is a relaxation of standard zero-knowledge obtained by reversing the order of quantifiers. Standard zero-knowledge requires that, for every verifier, there exists a simulator whose output is indistinguishable from the verifier’s view in a real interaction against every distinguisher. By contrast, weak zero-knowledge requires only that, for every verifier and every distinguisher, there exists a simulator that fools that particular distinguisher.¹ However, for our impossibility result to hold, the protocol must have a sufficiently small soundness error.

Theorem 4. *Any protocols for which Theorem 1 yields a scheme with negligible soundness error, cannot satisfy weak zero-knowledge (see Definition 7.1), unless the language is in BPP.*

1.2 Discussion: A Unified View of Fiat–Shamir

Our results provide a unified perspective on the Fiat–Shamir transformation, framing it not merely as a method for removing interaction, but as a mechanism for *verifier obfuscation*.

¹The notion ruled out in [FGJ18] additionally allows the distinguisher to possess private information unavailable to the other parties. Our result requires no such distinguisher-only information.

In the standard public-coin setting, the verifier’s role is syntactically simple: it outputs fresh random coins. When analyzed in the ROM, the transformation effectively wraps this verifier in a “black box” (the oracle) and provides it to the prover. The prover has only query access to the box to obtain the verifier’s random challenge, but the rest of the randomness of the oracle remains hidden. When we move to the standard model, instantiating the random oracle with a concrete hash function is conceptually equivalent to providing the prover with an *obfuscated* program that computes these coins. Specifically, standard Fiat–Shamir instantiations can be viewed as heuristics for obfuscating a pseudorandom function (PRF) that derives the verifier’s messages from the transcript.

Our work makes this implicit connection explicit. In the private-coin setting, the verifier is no longer a simple source of randomness; it is an arbitrary function that depends on a secret internal state. Our construction demonstrates that the core logic of the transformation remains the same: we first derandomize the verifier by embedding a PRF, and then we obfuscate the resulting circuit.

This suggests that the public-coin and private-coin settings are two points on the same continuum. In both cases, the transformation follows the same paradigm:

1. **Derandomization:** The verifier is made deterministic by deriving its randomness from the protocol transcript via a PRF.
2. **Obfuscation:** The resulting deterministic circuit is obfuscated and sent to the prover.

In the public-coin case, the circuit being obfuscated is relatively simple (essentially just a PRF), which allows for heuristic instantiations or weaker primitives (like correlation-intractable hashing). In the private-coin case, the circuit is complex and “evasive,” necessitating stronger, principled tools like IHO. Thus, the challenge of performing a Fiat–Shamir transformation for private-coin protocols is not a fundamental change in the protocol’s structure, but rather a reflection of the increased complexity required to obfuscate a general verifier compared to a random one.

1.3 Fiat–Shamir without IHO

We describe two additional results that obtain Fiat–Shamir transformations without assuming the existence of IHO. The first is a Fiat–Shamir transformation for the subclass of statistical zero-knowledge protocols, SZK, and additionally assumes only polynomially secure iO. The second is an alternative Fiat–Shamir transformation for public-coin protocols that avoids the use of IHO, but leverages subexponential security.

Fiat–Shamir for SZK without IHO. Our main theorem constructs a Fiat–Shamir transformation for general 3-message private-coin protocols assuming sub-exponential indistinguishability obfuscation, pseudorandom functions, and input-hiding obfuscation. Furthermore, our lower bound demonstrates that IHO is in fact necessary for any transformation targeting the general class of private-coin protocols.

However, we show that this requirement can be relaxed for specific structured subclasses. In particular, for the statistical distance problem SD, which is complete for the class of statistical zero-knowledge (SZK), we construct a Fiat–Shamir transformation that relies *only* on iO (and puncturable PRFs), completely removing the need for IHO. Moreover, unlike our main result, this transformation requires only polynomially secure iO and PRFs, and has only a polynomial security loss.

Theorem 5 (Informal). *Assume the existence of secure pseudorandom functions and indistinguishability obfuscation. Then, there is a Fiat–Shamir transformation for private-coin protocols*

for the language SD with polynomial error loss. The transformation also preserves honest-verifier zero-knowledge.

Fiat–Shamir for public-coin protocols without IHO. Our techniques also yield new insights into the standard Fiat–Shamir transformation for public-coin protocols. A standard approach to realizing this transformation is via a correlation-intractable hash (CIH) family for sparse relations. As observed by Canetti et al. [CCRR18], the construction of Kalai et al. [KRR17] effectively instantiates such a CIH family; however, their security proof relies on sub-exponentially secure puncturable PRFs and iO , and secure input-hiding obfuscation for multi-point functions. The work of Canetti et al. [CCRR18] achieves a similar result but relies on a non-standard assumption regarding exponentially secure KDM-encryption.

In contrast, we provide a pure obfuscation-based approach that eliminates the need for input-hiding obfuscation, albeit for sufficiently sparse relations (the relation does not need to be efficiently computable). Specifically, we present a Fiat–Shamir transformation for public-coin protocols incurring a security loss exponential in the instance size and first message length. As discussed, the soundness of the protocol can always be amplified by parallel repetition to satisfy our requirement.

Theorem 6 (Informal). *Assuming the existence of sub-exponentially secure pseudorandom functions and indistinguishability obfuscation, there exists a Fiat–Shamir transformation for public-coin protocols with the following parameters:*

Any 3-message public-coin protocol with soundness error ϵ compiles into a non-interactive argument with (adaptive) soundness error $\epsilon \cdot 2^n \cdot \text{poly}(\lambda) + \text{negl}(\lambda)$, where n denotes the combined length of the instance and the first prover message.

A summary of our various Fiat–Shamir transformations is given in Table 1.

Setting	Assumptions
Private-Coin	subexponential iO , subexponential PPRF, IHO
SZK (SD)	polynomially secure iO , polynomially secure PPRF
Public-Coin	subexponential iO , subexponential PPRF

Table 1: Summary of Fiat–Shamir transformations

Paper Organization and AI Disclosure

In Section 2, we give a high-level overview of our main results. In Section 3, we present preliminaries. In Section 4, we describe our main transformation and analyze its properties. In Section 5 we prove the necessity of input-hiding obfuscation. In Section 6, we give our impossibility for multi-round protocols. In Section 7 we give our impossibility of 3-message zero-knowledge protocols. In Section 8, we give our specialized analysis for SZK. In Section 9, we give our Fiat–Shamir transformation for public-coin protocols.

The ideas and proofs in this paper were developed and written by the authors. AI was used only for text polishing and finding references.

2 Technical Overview

In this section, we provide a high-level overview of the techniques used in this paper. We begin by introducing our Fiat–Shamir transformation in Section 2.1. Then, as a warm-up, we begin by describing the ideas required to prove non-adaptive security for the graph non-isomorphism (GNI) protocol in Section 2.2. This proof can be extended to prove security for the SD protocol, which we introduce in Section 8. Finally, in Section 2.3, we explain how to achieve security for *general* private-coin protocols that also satisfy *adaptive security*.

Before discussing how to achieve the transformation, we first define the Fiat–Shamir transformation for private-coin protocols.

The Fiat–Shamir transformation for private-coin protocols. We focus on the case of 3-message protocols, and later discuss multi-round protocols. In 3-message protocols, the verifier’s role can be decomposed into two dependent steps $(\mathbf{V}_1, \mathbf{V}_2)$. The first is generating the challenge, and the second is verifying the proof. We define the Fiat–Shamir transformation for 3-message private-coin protocols as a PPT algorithm **Setup**. This algorithm takes the original verifier $\mathbf{V}$ and a PRF key k as input, and outputs two circuits, **Chal** and **Ver**, where **Chal** serves as the challenge generator, and **Ver** serves as the verifier. The crs is then $(\mathbf{Chal}, \mathbf{Ver})$.

For completeness, we require that for every instance x , first prover message m_1 , second prover message m_2 , and randomness $\rho \leftarrow \text{PRF}_k(m_1)$, the following conditions hold:

1. $\mathbf{Chal}(x, m_1) = \mathbf{V}_1(x, m_1; \rho)$.
2. $\mathbf{Ver}(x, m_1, m_2) = \mathbf{V}_2(x, m_1, m_2; \rho)$.

These conditions preserve consistency with the original protocol. We think of the prover in the interactive protocol as two algorithms, $(\mathbf{P}_1, \mathbf{P}_2)$, that share auxiliary information. $\mathbf{P}_1$ generates the first message m_1 , while $\mathbf{P}_2$ generates the second message m_2 . The non-interactive prover then does as follows: it emulates the original prover to obtain the first message m_1 , computes the challenge $a \leftarrow \mathbf{Chal}(x, m_1)$, and uses a in the emulation as the verifier’s response to generate m_2 . It then outputs the proof $\pi := (m_1, m_2)$. The verifier executes $\mathbf{Ver}(m_1, m_2)$ to decide whether to accept or reject.

Thus, if the original protocol is perfectly complete, the two conditions above imply that $\mathbf{Ver}(x, m_1, m_2) = 1$ with probability 1, which means the non-interactive protocol is perfectly complete as well. For soundness, we require that for a randomly chosen PRF, a PPT adversary given $(\mathbf{Chal}, \mathbf{Ver}) \leftarrow \mathbf{Setup}(\mathbf{V}, k)$ cannot convince the verifier of a false statement with high probability. That is, the adversary cannot output an instance $x \notin \mathcal{L}$ and a pair (m_1, m_2) such that $\mathbf{Ver}(x, m_1, m_2) = 1$. For the formal definition, see Definition 5.1.

Note that this framework generalizes the standard Fiat–Shamir transformation. In public-coin protocols, the CRS consists only of the **Chal** circuit (the hash function). Since the challenge consists of public random coins and the verifier maintains no internal state, the **Ver** circuit is redundant. The verification can be simulated by obtaining the random coins from **Chal** and running the original verifier.

2.1 Constructing Fiat–Shamir for Private-Coin Protocols

Relation to FGJ’s round-reduction compiler. FGJ [FGJ18] already constructs a two-message argument from a three-message private-coin proof: the verifier sends an obfuscated challenge generator with a hardwired PRF key, the prover locally emulates the interaction, and the verifier checks

the response using the retained key. Our Chal follows this construction. We additionally publish an obfuscation of Ver , so verification requires only the crs and the proof.

This additional program gives a cheating prover the ability to test polynomially many candidate responses for the same challenge. Soundness must therefore hold even with this extra access to the decision procedure. Our analysis handles it by isolating the residual verification circuit after fixing an instance, first message, and challenge, and applying IHO to the resulting evasive circuit family (Section 2.3). Together with the analysis of adaptive instance selection, this yields a publicly verifiable non-interactive argument under our assumptions and soundness bound. The guarantees come with different requirements: FGJ uses input-hiding obfuscation for multi-bit point functions, whereas our extension uses IHO for general evasive circuit families and requires the underlying soundness error to be small enough to absorb the loss in Theorem 1.

Our construction. In the following construction, we formally define the circuits Chal , Ver relative to a PRF key k . The Setup algorithm publishes obfuscations of both circuits.

$\text{Chal}[k](x, m_1)$:

1. Compute $\rho \leftarrow F_k(x \| m_1)$.
2. Output $a \leftarrow V_1(x, m_1; \rho)$.

$\text{Ver}[k](x, m_1, m_2)$:

1. Compute $\rho \leftarrow F_k(x \| m_1)$.
2. Output $V_2(x, m_1, m_2; \rho)$.

$\text{Setup}(V, k)$:

1. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k])$.
2. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k])$.
3. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

$\text{Prove}(\text{crs}, x)$:

1. Compute:
 - (a) $(m_1, \text{aux}) \leftarrow P_1(x)$.
 - (b) $a \leftarrow \widehat{\text{Chal}}(x, m_1)$.
 - (c) $m_2 \leftarrow P_2(x, \text{aux}, a)$.
2. Output (m_1, m_2) .

$\text{Verify}(\text{crs}, x, m_1, m_2)$:

1. Output $\widehat{\text{Ver}}(x, m_1, m_2)$.

We now turn to the security analysis. As a warm-up, we first apply the transformation to the Graph Non-Isomorphism (GNI) protocol and prove *non-adaptive* security for the resulting non-interactive argument.

2.2 Warm-Up: Analysis for GNI

We consider the λ -parallel repetition version of the GNI protocol. In this amplified setting, the verifier samples challenge bits $\vec{b} = (b_1, \dots, b_\lambda)$ and randomness $\vec{r} = (r_1, \dots, r_\lambda)$. Then, the verifier sends the permuted graphs $H_1, \dots, H_\lambda$ to the prover, where $H_i := \text{Perm}(G_{b_i}; r_i)$ is the permutation of G_{b_i} using randomness r_i . The prover then attempts to guess $\vec{b}$. If the input graphs are isomorphic,

the distribution of the challenges is independent of $\vec{b}$, bounding the success probability of guessing $\vec{b}$ to $2^{-\lambda}$. If the graphs are not isomorphic, the distributions of the challenges for $b_i = 0$ and $b_i = 1$ have disjoint supports, allowing an unbounded prover to find $\vec{b}$ with probability 1.

Below is the resulting non-interactive protocol. Note that since GNI is a 2-message protocol (lacking a first prover message), our transformation simplifies by making the challenge derived deterministically from the instance alone.

Non-interactive GNI	
Chal[k]($x = (G_0, G_1)$): 1. Compute $\rho \leftarrow F_k(x)$. 2. Parse ρ as $\vec{b}, \vec{r}$. 3. For $i \in [\lambda]$, set $H_i := \text{Perm}(G_{b_i}; r_i)$. 4. Output challenge $\vec{H} = (H_1, \dots, H_\lambda)$.	Ver[k](x, π): 1. Compute $\rho \leftarrow F_k(x)$. 2. Parse ρ as $\vec{b}, \vec{r}$. 3. Output 1 iff $\pi = \vec{b}$.
Setup(k): 1. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k])$ 2. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k])$. 3. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.	
Prove(crs, $x = (G_0, G_1)$): 1. Compute $(H_1, \dots, H_\lambda) := \widehat{\text{Chal}}(x)$. 2. For $i \in [\lambda]$: 1. If (H_i, G_0) are isomorphic, set $b_i := 0$ 2. Else, $b_i := 1$. 3. Output $\pi := (b_1, \dots, b_\lambda)$.	
Verify(crs, x, π): 1. Output $\widehat{\text{Ver}}(x, \pi)$.	

We now turn to the security analysis. Beyond the structural simplification of the 2-message protocol and the weaker (non-adaptive) security that we prove in this warm-up example, the satisfying proof in GNI is both unique and uniformly sampled by the verifier. We explicitly leverage these properties in the proofs.

In what follows, we define a sequence of computationally indistinguishable hybrids, starting with the real non-interactive protocol (Hybrid 0) and concluding with a hybrid where the prover cannot convince the verifier of a false statement except with negligible probability. This proof assumes the existence of secure puncturable PRFs, indistinguishability obfuscation (iO), and an injective one-way function.

Hybrid 1. Fix a false instance $x^* = (G_0^*, G_1^*) \notin \text{GNI}$. In Hybrid 1, we puncture the PRF key at x^* and replace the pseudorandom value with a hardcoded value ρ^* that was uniformly sampled by the setup algorithm.

Hybrid 1: puncturing the PRF key

$\text{Chal}[k\{x^*\}, \rho^*](x = (G_0, G_1)):$

1. If $(x = x^*)$, set $\rho := \rho^*$.
2. Else, set $\rho \leftarrow F_{k\{x^*\}}(x)$.
3. Parse ρ as $\vec{b}, \vec{r}$.
4. For $i \in [\lambda]$, set $H_i := \text{Perm}(G_{b_i}; r_i)$.
5. Output challenge $\vec{H} = (H_1, \dots, H_\lambda)$.

$\text{Ver}[k\{x^*\}, \rho^*](x, \pi):$

1. If $(x = x^*)$, set $\rho := \rho^*$.
2. Else, set $\rho \leftarrow F_{k\{x^*\}}(x)$.
3. Parse ρ as $\vec{b}, \vec{r}$.
4. Output 1 iff $\pi = \vec{b}$.

$\text{Setup}(k):$

1. $k\{x^*\} \leftarrow \text{PRF.Punc}(k, x^*)$.
2. Sample uniformly random ρ^* .
3. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k\{x^*\}, \rho^*])$.
4. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k\{x^*\}, \rho^*])$.
5. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

Indistinguishability follows by the security of the puncturable PRF and iO schemes.

Let $\rho^* = (\vec{b}^*, \vec{r}^*)$. Our goal in the subsequent hybrids is to “hide” the challenge bits $\vec{b}^*$ from the adversary. This hiding suffices for proving soundness, as $\vec{b}^*$ is the unique valid proof that the verifier accepts.

In *Hybrid 2*, we modify the circuits to directly output the values computed for x^* , rather than computing them from ρ^* . Specifically, we modify Chal to output the hardcoded challenge $\vec{H}^*$ on input x^* , removing the dependency on $\vec{r}^*$. Since $x^* = (G_0^*, G_1^*) \notin \text{GNI}$ (implying the graphs are isomorphic), the challenge $\vec{H}^*$, which consists of random permutations of the input graphs, is statistically independent of the choice of bits $\vec{b}^*$ (and therefore reveals no information about it on its own). Regarding the circuit Ver , we modify it to directly check if the proof equals $\vec{b}^*$.

Hybrid 2: hardcoding the challenge and verification

$\text{Chal}[k\{x^*\}, \vec{H}^*](x = (G_0, G_1)):$

1. If $(x = x^*)$
 - a. Output $\vec{H}^*$.
2. Else $(x \neq x^*)$
 - a. Set $\rho \leftarrow F_{k\{x^*\}}(x)$.
 - b. Parse ρ as $\vec{b}, \vec{r}$.
 - c. For $i \in [\lambda]$, set $H_i := \text{Perm}(G_{b_i}; r_i)$.
 - d. Output challenge $\vec{H} = (H_1, \dots, H_\lambda)$.

$\text{Ver}[k\{x^*\}, \vec{b}^*](x, \pi):$

1. If $(x = x^*)$
 - a. Output 1 iff $\pi = \vec{b}^*$.
2. Else $(x \neq x^*)$
 - a. Set $\rho \leftarrow F_{k\{x^*\}}(x)$.
 - b. Parse ρ as $\vec{b}, \vec{r}$.
 - c. Output 1 iff $\pi = \vec{b}$.

$\text{Setup}(k):$

1. $k\{x^*\} \leftarrow \text{PRF.Punc}(k, x^*)$.
2. Sample uniformly random ρ^* , and parse as $\vec{b}^*, \vec{r}^*$.

3. For $i \in [\lambda]$, set $H_i^* := \text{Perm}(G_{b_i^*}; r_i^*)$.
4. Set $\vec{H}^* = (H_1^*, \dots, H_\lambda^*)$.
5. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k\{x^*\}, \vec{H}^*])$.
6. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k\{x^*\}, \vec{b}^*])$.
7. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

Since these modifications merely replace the circuit's computation on x^* with its fixed output, the functionality of the circuits remains the same. Consequently, by the security of iO , the crs in Hybrid 2 is computationally indistinguishable from the crs in Hybrid 1.

In *Hybrid 3*, we decouple the hardcoded challenge $\vec{H}^*$ from the proof bits $\vec{b}^*$. We achieve this by modifying the **Setup** algorithm to generate the challenge by always permuting G_0^* , regardless of the value of $\vec{b}^*$ (i.e., we set $H_i^* := \text{Perm}(G_0^*; r_i^*)$).

Hybrid 3: decoupling the proof from the challenge

$\text{Chal}[k\{x^*\}, \vec{H}^*](x = (G_0, G_1))$:

1. If $(x = x^*)$
 - a. Output $\vec{H}^*$.
2. Else $(x \neq x^*)$
 - a. Set $\rho \leftarrow F_{k\{x^*\}}(x)$.
 - b. Parse ρ as $\vec{b}, \vec{r}$.
 - c. For $i \in [\lambda]$, set $H_i := \text{Perm}(G_{b_i}; r_i)$.
 - d. Output challenge $\vec{H} = (H_1, \dots, H_\lambda)$.

$\text{Ver}[k\{x^*\}, \vec{b}^*](x, \pi)$:

1. If $(x = x^*)$
 - a. Output 1 iff $\pi = \vec{b}^*$.
2. Else $(x \neq x^*)$
 - a. Set $\rho \leftarrow F_{k\{x^*\}}(x)$.
 - b. Parse ρ as $\vec{b}, \vec{r}$.
 - c. Output 1 iff $\pi = \vec{b}$.

Setup(k):

1. $k\{x^*\} \leftarrow \text{PRF.Punc}(k, x^*)$.
2. Sample uniformly random ρ^* , and parse as $\vec{b}^*, \vec{r}^*$.
3. For $i \in [\lambda]$, set $H_i^* := \text{Perm}(G_0; r_i^*)$.
4. Set $\vec{H}^* = (H_1^*, \dots, H_\lambda^*)$.
5. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k\{x^*\}, \vec{H}^*])$.
6. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k\{x^*\}, \vec{b}^*])$.
7. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

Recall that G_0^* and G_1^* are isomorphic. Therefore, the distribution of the challenge graphs is identical regardless of whether they are generated by permuting G_0^* or G_1^* , making the distribution of the hardcoded constant $\vec{H}^*$ identical to the previous hybrid and independent of $\vec{b}^*$. Consequently, the distributions of the crs in this hybrid and the previous hybrid are identical.

In *Hybrid 4*, we use an injective one-way function f . In the setup, we sample f and compute the image $v = f(\vec{b}^*)$ of the hardcoded proof's bits. We then modify the **Ver** circuit: on input x^* , instead of directly checking $\pi = \vec{b}^*$, it checks if $f(\pi) = v$.

Hybrid 4: replacing verification with OWF image check

$\text{Chal}[k\{x^*\}, \vec{H}^*](x = (G_0, G_1)):$

1. If $(x = x^*)$
 - a. Output $\vec{H}^*$.
2. Else $(x \neq x^*)$
 - a. Set $\rho \leftarrow F_{k\{x^*\}}(x)$.
 - b. Parse ρ as $\vec{b}, \vec{r}$.
 - c. For $i \in [\lambda]$, set $H_i := \text{Perm}(G_{b_i}; r_i)$.
 - d. Output challenge $\vec{H} = (H_1, \dots, H_\lambda)$.

$\text{Ver}[k\{x^*\}, f, v](x, \pi):$

1. If $(x = x^*)$
 - a. **Output 1 iff $f(\pi) = v$.**
2. Else $(x \neq x^*)$
 - a. Set $\rho \leftarrow F_{k\{x^*\}}(x)$.
 - b. Parse ρ as $\vec{b}, \vec{r}$.
 - c. Output 1 iff $\pi = \vec{b}$.

$\text{Setup}(k):$

1. $k\{x^*\} \leftarrow \text{PRF.Punc}(k, x^*)$.
2. Sample uniformly random ρ^* , and parse as $\vec{b}^*, \vec{r}^*$.
3. For $i \in [\lambda]$, set $H_i^* := \text{Perm}(G_0; r_i^*)$.
4. Set $\vec{H}^* = (H_1^*, \dots, H_\lambda^*)$.
5. **Sample $f \leftarrow \text{OWF.Gen}(1^\lambda)$.**
6. **Compute $v \leftarrow f(b_1^*, \dots, b_\lambda^*)$.**
7. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k\{x^*\}, \vec{H}^*])$.
8. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k\{x^*\}, f, v])$.
9. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

Note that the injectivity of the one-way function guarantees that $\vec{b}^*$ remains the unique accepting proof for x^* , thereby preserving the functionality of the circuits. Indistinguishability between the hybrids then follows from the security of iO .

Overall, we have shown that the distribution of the crs in Hybrid 0 is computationally indistinguishable from that in Hybrid 4. To finalize the proof, we now show that in the final hybrid, no PPT adversary can convince the verifier of the false instance x^* with more than negligible probability.

By the description of Hybrid 4, finding a valid proof for x^* is equivalent to inverting the one-way function on a random image v . Observe that the only information regarding $\vec{b}^*$ in the crs is its image v (as we removed the dependency of the circuit Chal in Hybrid 3). Therefore, by the security of the one-way function, the probability that an adversary can recover $\vec{b}^*$ is negligible. Since finding $\vec{b}^*$ is equivalent to finding a proof, this finalizes the proof.

2.3 Analysis for Private-Coin Protocols

We now turn to prove adaptive security for generic private-coin 3-message protocols. This generalization introduces some significant challenges compared to our warm-up example. One of the challenges is due to the first prover message that is added to the protocol. Unlike the non-adaptive GNI setting, we cannot trivially puncture the PRF at a pre-determined location, as the adversary chooses the first message after the crs generation. Another, more significant, challenge arises from the potential existence of multiple convincing proofs for a false statement, possibly even super-polynomial many such false proofs. In the GNI protocol, we relied on the uniqueness of the

satisfying proof to hardcode it into the circuit. Here, enumerating every valid proof within the circuit is infeasible.

As the first step in the proof, we describe how to choose where to puncture the PRF. Note that we want to achieve adaptive security, so both the instance and the first prover message are chosen after the crs was generated. To achieve the highest advantage in the reduction, we fix the pair (x^*, m_1^*) that maximizes the probability of the event where the adversary both outputs this pair and wins the game (i.e., $x^* \notin \mathcal{L}$ and the adversary successfully provides a convincing response m_2). We then continue the reduction while focusing on this specific attack point.

If the adversary wins the game with overall probability ν , then by an averaging argument, we show that the adversary succeeds on (x^*, m_1^*) with probability at least $\nu/2^{(n+\ell_1)}$, where n is the size of the instance and ℓ_1 is the size of the first prover message. To tolerate the exponential loss incurred by fixing this optimal pair, we assume sub-exponential security for the underlying interactive protocol, the PRF, and the iO schemes. We therefore instantiate the iO and PRF with a sufficiently large security parameter, polynomial in the instance size n , first message size ℓ_1 , and the original security parameter, to ensure security scales appropriately. Note that while this larger security parameter increases the size of the crs , it does not affect the proof, which simply remains (m_1, m_2) .

In what follows, we define a sequence of hybrids that are indistinguishable up to at most $2^{-c(n+\ell_1)}$ for some constant $c > 1$. We then prove that in the final hybrid, the adversary cannot find a second prover message m_2 such that the verifier accepts (x^*, m_1^*, m_2) with probability more than $\sqrt{\epsilon} \cdot \text{poly}(\lambda)$, where ϵ is the security of the underlying interactive protocol. Overall, by combining the hybrids, we get that $\nu/2^{(n+\ell_1)} - 2^{-c(n+\ell_1)} < \sqrt{\epsilon} \cdot \text{poly}(\lambda)$, which implies that ν is negligible assuming ϵ is small enough.

Let *Hybrid 0* be the non-interactive protocol generated by the transformation. In *Hybrid 1*, we puncture the PRF at location (x^*, m_1^*) , and replace the value of the PRF with a uniformly random string.

Hybrid 1: puncturing the PRF key	
$\text{Chal}[\mathbf{k}^*, \rho^*](x, m_1):$ 1. If $(x, m_1) = (x^*, m_1^*)$, a. Set $\rho := \rho^*$. 2. Else $(x \neq x^*)$: a. Set $\rho \leftarrow F_{\mathbf{k}^*}(x \ m_1)$. 3. Output $\mathbf{V}_1(x, m_1; \rho)$.	$\text{Ver}[\mathbf{k}^*, \rho^*](x, m_1, m_2):$ 1. If $(x, m_1) = (x^*, m_1^*)$, a. Set $\rho := \rho^*$. 2. Else $(x \neq x^*)$: a. Set $\rho \leftarrow F_{\mathbf{k}^*}(x \ m_1)$. 3. Output $\mathbf{V}_2(x, m_1, m_2; \rho)$.
$\text{Setup}(\mathbf{k}):$ 1. $\mathbf{k}^* \leftarrow \text{PRF.Punc}(\mathbf{k}, x^* \ m_1^*)$. 2. Sample uniformly random ρ^*. 3. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[\mathbf{k}\{x^*\}, \rho^*])$ 4. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[\mathbf{k}\{x^*\}, \rho^*])$. 5. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.	

Indistinguishability of the hybrids follows by the security of the iO and the PRF schemes.

In the subsequent steps, we aim to transition to a hybrid where we can prove that it is hard to find a response m_2 such that the verifier accepts (x^*, m_1^*, m_2) . Generally speaking, the security of private-coin protocols strictly relies on the verifier's random coins remaining hidden from the prover. Therefore, if we had access to Virtual Black Box (VBB) obfuscation, our proof would be immediate: the adversary would interact with the circuit essentially as a black box, replicating the interaction with the verifier in the interactive protocol. However, in the absence of VBB, security is not guaranteed if the obfuscated circuit leaks information about the verifier's internal random coins.

As the first step toward hiding the verifier's randomness ρ^* , in Hybrid 2, we modify the circuits to directly output the values for (x^*, m_1^*) , rather than computing them from ρ^* . Specifically, we modify Chal to output the hardcoded challenge $\mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho^*)$, removing the direct dependency on ρ^* . For the verification circuit, we encapsulate the verification functionality by defining a dedicated circuit $C_{\rho^*}(\cdot) := \mathbf{V}_2(x^*, m_1^*, \cdot; \rho^*)$, which is then embedded into Ver .

Hybrid 2: removing direct dependency on randomness

$\text{Chal}[\mathbf{k}^*, \mathbf{a}^*](x, m_1)$:

1. If $(x, m_1) = (x^*, m_1^*)$:
 - a. **Output $\mathbf{a}^*$.**
2. Else $(x \neq x^*)$:
 - a. Set $\rho \leftarrow \mathbf{F}_{\mathbf{k}^*}(x \| m_1)$.
 - b. Output $\mathbf{V}_1(x, m_1; \rho)$.

$\text{Ver}[\mathbf{k}^*, C_{\rho^*}](x, m_1, m_2)$:

1. If $(x, m_1) = (x^*, m_1^*)$:
 - a. **Output $C_{\rho^*}(m_2)$.**
2. Else $(x \neq x^*)$:
 - a. Set $\rho \leftarrow \mathbf{F}_{\mathbf{k}^*}(x \| m_1)$.
 - b. Output $\mathbf{V}_2(x, m_1, m_2; \rho)$.

$\text{Setup}(\mathbf{k})$:

1. $\mathbf{k}^* \leftarrow \text{PRF.Punc}(\mathbf{k}, x^* \| m_1^*)$.
2. Sample uniformly random ρ^* .
3. **Compute $\mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho^*)$.**
4. **Define $C_{\rho^*}(\cdot) := \mathbf{V}_2(x^*, m_1^*, \cdot; \rho^*)$.**
5. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[\mathbf{k}^*, \mathbf{a}^*])$.
6. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[\mathbf{k}^*, C_{\rho^*}])$.
7. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

Note that the challenge $\mathbf{a}^*$ itself should not provide the adversary with a significant advantage (due to the security of the underlying interactive protocol). The only location where we might leak information about ρ^* that can compromise security is the circuit C_{ρ^*} . At this point, in the GNI protocol, we relied on the specific structure of the satisfying proof: a unique, uniformly sampled vector. This allowed us to use an injective one-way function to hardcode the verification check while keeping the proof itself hidden. However, in the generic setting, C_{ρ^*} lacks this simple structure and may accept multiple (possibly super-polynomial many) valid proofs, making it impossible to enumerate or hardcode them directly. To address this, we use input-hiding obfuscation (IHO).

Using input-hiding obfuscation. An input-hiding obfuscator (IHO) is a PPT algorithm that takes as input a circuit C and outputs a new circuit $\widehat{C} \leftarrow \text{IHO}(C)$ that preserves the same functionality. We say that a circuit family $\mathcal{C}$ is s -sparse if for any fixed input x , the probability over

the choice of $C \leftarrow \mathcal{C}$ that $C(x) \neq 0$ is at most s (see Definition 3.8). The security of input-hiding obfuscation guarantees that given $\widehat{C} \leftarrow \text{IHO}(C)$, where $C \leftarrow \mathcal{C}$, a PPT adversary running in time T cannot find an input x such that $C(x) = 1$ with probability greater than $s \cdot T \cdot \text{poly}(\lambda)$, for security parameter λ is the security parameter. For the formal definition, see Definition 3.9.

In Hybrid 3, we replace the explicit circuit C_ρ embedded in Ver with an obfuscated version. Specifically, we generate $\widehat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*})$ and embed this obfuscated circuit into Ver .

Hybrid 3: obfuscating the verification	
$\text{Chal}[k^*, a^*](x, m_1):$ 1. If $(x, m_1) = (x^*, m_1^*)$: a. Output a^*. 2. Else $(x \neq x^*)$: a. Set $\rho \leftarrow F_{k^*}(x \ m_1)$. b. Output $V_1(x, m_1; \rho)$.	$\text{Ver}[k^*, \widehat{C}_{\rho^*}](x, m_1, m_2):$ 1. If $(x, m_1) = (x^*, m_1^*)$: a. Output $\widehat{C}_{\rho^*}(m_2)$. 2. Else $(x \neq x^*)$: a. Set $\rho \leftarrow F_{k^*}(x \ m_1)$. b. Output $V_2(x, m_1, m_2; \rho)$.
$\text{Setup}(k):$ 1. $k^* \leftarrow \text{PRF.Punc}(k, x^* \ m_1^*)$. 2. Sample uniformly random ρ^*. 3. Compute $a^* \leftarrow V_1(x^*, m_1^*; \rho^*)$. 4. Define $C_{\rho^*}(\cdot) := V_2(x^*, m_1^*, \cdot; \rho^*)$. 5. $\widehat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*})$. 6. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k^*, a^*])$. 7. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k^*, \widehat{C}_{\rho^*}])$. 8. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.	

We now turn to bound the adversary's success probability in Hybrid 3. To do so, we need to show that the adversary cannot find an input m_2 such that $C_{\rho^*}(m_2) = 1$ with more than negligible probability. Note that we cannot bound the success probability by simply invoking the security of the IHO scheme over the general circuit family $\mathcal{C} = \{C_\rho\}_{\rho \in \{0,1\}^r}$ (where r denotes the length of the verifier's randomness). While this family is indeed ϵ -sparse due to the soundness of the protocol, the adversary does not get the obfuscated circuit $\widehat{C}_{\rho^*}$ in isolation. We rather use the challenge $a^* \leftarrow V_1(x^*, m_1^*; \rho^*)$ in the crs , which leaks information about the obfuscated circuit, preventing us from using the IHO security.

Conditioning on the verifier's challenge also plays a role in FGJ's soundness analysis [FGJ18]. Here we use it to obtain an evasive family of decision circuits to which we can apply IHO, despite making verification public. To address this dependency, we define a new circuit family conditioned on the specific challenge $a^* \leftarrow V_1(x^*, m_1^*; \rho^*)$. We define S_{a^*} as the set of all randomnesses ρ that are consistent with this challenge:

$$S_{a^*} := \{\rho \in \{0,1\}^r \mid V_1(x^*, m_1^*; \rho) = a^*\}.$$

We then consider the conditional circuit family $\mathcal{C}_{a^*} := \{C_\rho\}_{\rho \in S_{a^*}}$. Each circuit in the family is

defined as before: $C_\rho(\cdot) := \mathbf{V}_2(x^*, m_1^*, \cdot; \rho)$, but the support of the family is restricted to a subset of circuits determined by S_{a^*} .

Regarding the sparsity of this circuit family, we prove that it is $\sqrt{\epsilon}$ -sparse with high probability (Claim 4.6). The sparsity of $\mathcal{C}_{a^*}$ corresponds to the success probability of the optimal strategy against the interactive protocol. Specifically, if for a fixed challenge a^* , the circuit family $\mathcal{C}_{a^*}$ is s -sparse, then an unbounded adversary can find a response m_2 for the challenge a^* which convinces the verifier with probability s . Therefore, by the ϵ soundness of the interactive protocol, the probability of sampling a challenge a^* such that the sparsity of $\mathcal{C}_{a^*}$ exceeds $\sqrt{\epsilon}$ is at most $\sqrt{\epsilon}$.

By the IHO security, we get that with high probability over the choice of a^* , a PPT adversary given $\hat{C}_{\rho^*}$ cannot find an input m_2 such that $C_{\rho^*}(m_2) = 1$ with probability greater than $\sqrt{\epsilon} \cdot \text{poly}(\lambda)$, which concludes the proof.

2.4 On The Necessity of Input-Hiding Obfuscation

We give an overview of the proof that IHO is inherent to the security analysis of the Fiat-Shamir transformation for private-coin protocols (Theorem 2). Specifically, we show that if a Fiat-Shamir transformation for private-coin protocols exists, then, assuming the existence of a puncturable PRF, there exists an IHO scheme for any evasive circuit family. See Theorem 5.2 for the formal theorem statement.

Fix any circuit family $\mathcal{C}$ for circuits of the form $C : \{0, 1\}^n \rightarrow \{0, 1\}$ that is $\text{negl}(\lambda)$ -sparse. We construct IHO for $\mathcal{C}$. For any security parameter $\lambda \in \mathbb{N}$, let $(\mathbf{P}, \mathbf{V})$ be the following 3-message private-coin protocol for the empty language. Given an instance $x \in \{0, 1\}^n$ the protocol is the following:

1. $\mathbf{P}(x)$: The prover sends the first message $m_1 := x$.
2. $\mathbf{V}(x, m_1)$: The verifier samples $\rho \leftarrow \{0, 1\}^{2\lambda}$, and sends $a := (\rho_1, \dots, \rho_\lambda)$ as the challenge.
3. $\mathbf{P}(x)$: The prover sends $m_2 := \perp$.
4. $\mathbf{V}(x, m_1, m_2)$: The verifier accepts if and only if $m_2 = \rho$.

Note that an adversary (even an unbounded one) cannot guess the last λ bits of ρ with probability greater than $2^{-\lambda}$. Therefore, the protocol has $2^{-\lambda}$ soundness error (and perfect completeness for the empty language).

Embedding $\mathcal{C}$ in the PRF. Let $\text{PRF}_{\mathcal{C}}$ be a pseudorandom function that takes as input n bits and outputs 2λ bits. The new key consists of a PRF key $k \leftarrow \text{PRF.Gen}(1^\lambda)$ and a circuit $C \leftarrow \mathcal{C}$. The PRF evaluator behaves the same as $\text{PRF.Eval}(k, \cdot)$, except on inputs $x \in \{0, 1\}^n$ where $C(x) = 1$ or $\text{PRF.Eval}(k, x) = \vec{1}$. The construction is defined as follows:

- $\text{PRF}_{\mathcal{C}}.\text{Gen}(1^\lambda)$:
 1. Sample $k \leftarrow \text{PRF.Gen}(1^\lambda)$.
 2. Sample $C \leftarrow \mathcal{C}$.
 3. Output $k_C := (k \| C)$.
- $\text{PRF}_{\mathcal{C}}.\text{Eval}(k_C, x)$:
 1. Parse k_C as $(k \| C)$.
 2. If $C(x) = 1$, output $\vec{1}$.
 3. Else,
 - (a) If $\text{PRF.Eval}(k, x) = \vec{1}$, then output $\vec{2}$.
 - (b) Else, output $\text{PRF.Eval}(k, x)$.

We argue that PRF_C is a secure puncturable PRF. By the evasiveness of the circuit family, an efficient adversary with oracle access to a circuit $C \leftarrow \mathcal{C}$ cannot find an input x such that $C(x) = 1$ with more than negligible probability. Therefore, an adversary to PRF_C will not “hit” a location where $C(x) = 1$ with more than negligible probability. Moreover, a PPT adversary with oracle access to a PRF with output domain of size 2λ , cannot find an input x that maps to a specific fixed value (such as $\vec{1}$) with probability more than $2^{-2\lambda} + \text{negl}(\lambda)$, as this would allow distinguishing the PRF from a random oracle. Therefore, with overwhelming probability, the view of a PPT adversary with oracle access to PRF_C is identical to its view if the oracle were replaced by the underlying PRF. Thus, the security of the underlying PRF implies the security of PRF_C . See Claim 5.3 for the formal claim and the proof. Puncturability is preserved by puncturing only the underlying PRF key and retaining C , since evasiveness and the negligible probability of remapping $\vec{1}$ ensure pseudorandomness at any fixed puncture point.

The IHO construction. In what follows, we construct the IHO scheme for $\mathcal{C}$ using the protocol $(\mathbf{P}, \mathbf{V})$ and PRF_C . In the construction, for an input circuit C , we first sample a PRF key k and set $k_C := (k || C)$. Next, we apply the Fiat-Shamir transformation using k_C to the protocol $(\mathbf{P}, \mathbf{V})$. Finally, we output the generated circuit Ver , with an arbitrary instance x and the second prover message m_2 hardcoded to $\vec{1}$ as the new obfuscated circuit.

- $\text{IHO}(C; r)$.
 1. Sample $k \leftarrow \text{PRF.Gen}(1^\lambda)$.
 2. Set $k_C := (k || C)$ as a PRF key for the PRF_C scheme.
 3. Apply Fiat-Shamir transformation, $(\text{Ver}, \text{Chal}) \leftarrow \text{FS.Setup}(\mathbf{V}, k_C)$, using PRF_C .
 4. Output $\hat{C} := \text{Ver}(x, \cdot, \vec{1})$, where x is an arbitrary instance.

We argue that this scheme is IHO-secure and preserves functionality. For the functionality, the intuition is as follows: before applying the Fiat-Shamir transformation, the underlying protocol accepts if and only if the verifier’s randomness matches the second prover message ($m_2 = \rho$). Since the transformation preserves the functionality of the underlying verifier, and we fix the second prover message to be $\vec{1}$, the circuit $\text{Ver}(x, m_1, \vec{1})$ outputs 1 if and only if the randomness generated by $\text{PRF}_C.\text{Eval}(k_C, m_1)$ equals $\vec{1}$. Recall that we constructed PRF_C specifically to output $\text{PRF}_C.\text{Eval}(k_C, m_1) = \vec{1}$ if and only if $C(m_1) = 1$. Therefore, it holds that $\text{Ver}(x, m_1, \vec{1}) = 1$ if and only if $C(m_1) = 1$, as required.

For the IHO security, we show that if there exists an adversary against the IHO scheme that can output a value m_1 satisfying $C(m_1) = 1$ given the obfuscated circuit $\hat{C}$ (which was set to $\text{Ver}(x, \cdot, \vec{1})$), then this adversary can be used to break the non-interactive protocol generated by the Fiat-Shamir transformation, contradicting its soundness guarantee. Let Adv be an adversary against the IHO scheme. The new adversary Adv' receives the pair $(\text{Chal}, \text{Ver})$ as the crs , and then emulates Adv on $\hat{C} := \text{Ver}(x, \cdot, \vec{1})$ to obtain m_1 . The adversary outputs the instance x and the proof $\pi := (m_1, \vec{1})$. If $C(m_1) = 1$, then $\text{Ver}(x, m_1, \vec{1}) = 1$, which breaks the soundness of the Fiat-Shamir transformation.

2.5 Impossibility for Protocols Beyond 3 Messages

We have seen that our Fiat-Shamir transformation for private-coin protocols can be successfully applied to three-message protocols. It is natural to ask whether a similar approach extends to multi-round protocols. Unfortunately, we show that this is not the case. Specifically, assuming the

existence of collision-resistant hash (CRH) functions, we prove that a Fiat–Shamir transformation for private-coin protocols cannot exist once the protocol includes more than a single verifier message. Our result is established via a contrived example of such a protocol that cannot be made non-interactive without contradicting the security of the CRH.

Our counterexample works as follows. Let λ be the security parameter and hk be a hash key for a collision-resistant hash function Hash that maps 2λ bits to λ bits. Consider the following 4-message interactive protocol for the empty language $\mathcal{L} = \emptyset$:

1. $\mathbf{V}_{\text{hk}}(x)$: The verifier samples $\rho \leftarrow \{0, 1\}^{2\lambda}$ and sends $y := \text{Hash}(\text{hk}, \rho)$ to the prover.
2. $\mathbf{P}_{\text{hk}}(x)$: The prover sends $m_1 := \perp$.
3. $\mathbf{V}_{\text{hk}}(x)$: The verifier sends ρ as the second verifier message.
4. $\mathbf{V}_{\text{hk}}(x)$: The verifier accepts if and only if the prover message m_1 satisfies $m_1 = \rho$.

Note that this protocol is statistically sound. Since ρ is sampled uniformly from $\{0, 1\}^{2\lambda}$, even an unbounded adversary cannot guess a pre-image of y under Hash with probability greater than $2^{-\lambda}$.

The transformation. A Fiat-Shamir transformation for the protocol above would yield a setup algorithm Setup outputting three circuits: Chal_1 , which generates the first verifier message y , Chal_2 , which generates the second verifier message ρ , and Ver , the final decision circuit. We show that if the transformation preserves soundness, then we can construct an adversary that breaks the collision-resistant hash function.

The CRH adversary. Fix an arbitrary instance x and prover message m_1 . We define a PPT adversary $\text{Adv}(\text{hk})$ against the collision resistance hash function Hash as follows.

1. Sample $(\text{Chal}_1, \text{Chal}_2, \text{Ver}) \leftarrow \text{Setup}(1^\lambda, \mathbf{V}_{\text{hk}})$.
2. Compute $\rho \leftarrow \text{Chal}_2(x, m_1)$, and set $m_1' = \rho$.
3. Compute $\rho' \leftarrow \text{Chal}_2(x, m_1')$.
4. Output (ρ, ρ') .

By the completeness of the transformation, we know that the transcript $(\text{Chal}_1(x), m_1, \text{Chal}_1(x, m_1))$ must be consistent with the transcript of the interactive protocol. Specifically, if $\rho = \text{Chal}_2(x, m_1)$, then it holds that $\text{Chal}_1(x) = \text{Hash}(\text{hk}, \rho)$. Similarly, if $\rho' = \text{Chal}_2(x, \rho)$, it must hold that $\text{Chal}_1(x) = \text{Hash}(\text{hk}, \rho')$. Thus, Adv always outputs a pair (ρ, ρ') such that $\text{Hash}(\text{hk}, \rho) = \text{Hash}(\text{hk}, \rho')$.

Next, we prove that $\rho \neq \rho'$ with overwhelming probability. Suppose, toward a contradiction, that $\rho = \rho'$. By the construction of Adv , this equality implies $m_1' = \text{Chal}_2(x, m_1')$. By the completeness of the transformation, if the prover message equals the verifier’s second message, then the verifier must accept: $\text{Ver}(x, m_1') = 1$. However, since $x \notin \mathcal{L}$, the soundness of the transformation implies that this can happen with at most negligible probability. Thus, Adv successfully finds a collision for $\text{Hash}(\text{hk}, \cdot)$ with overwhelming probability.

2.6 On The Existence of 3-Message Zero-Knowledge Proofs for NP

We show that our Fiat-Shamir transformation for private-coin protocols rules out 3-message protocols for languages outside BPP that satisfy zero-knowledge with bounded auxiliary input. In this setting, a single auxiliary input is shared among all parties - the distinguisher, simulator, and verifier - without giving the distinguisher any additional information (see Definition 7.1). While this notion of zero-knowledge is distinct from the weak zero-knowledge studied in [DNRS99; BKP19;

GO94], the latter implies our definition. Consequently, ruling out zero-knowledge with bounded auxiliary input effectively rules out weak zero-knowledge as well.

Our approach takes inspiration from Fleischhacker et al. [FGJ18], with the key difference that their transformation results in a 2-message protocol that requires private information for verification, whereas our transformation fully obfuscates the verifier, allowing verification to be public. This distinction is the key to ruling out the weaker notion of zero-knowledge.

In more detail, we define $\Pi = (\mathbf{P}, \mathbf{V})$ to be a 3-message proof system for some language $\mathcal{L}$, and $\Pi' = (\text{FS.Setup}, \text{FS.P}, \text{FS.V})$ to be the protocol after the transformation. Let $p(n) = \text{poly}(n)$ be the size of the crs generated by the setup. Assuming that Π satisfies zero-knowledge with p -bounded auxiliary input and that Π' has $\text{negl}(n)$ -soundness error, we show that $\mathcal{L} \in \text{BPP}$.

Let D be a distinguisher that, given $(m_1, m_2, \text{aux} = \text{crs})$, outputs 1 if and only if the verifier FS.V would have accepted the proof $\pi = (m_1, m_2)$. Let Sim be the simulator of the protocol relative to $\mathbf{V}^*$ and D . We can then decide the language $\mathcal{L}$ using the following PPT algorithm:

1. Sample $\text{crs} \leftarrow \text{FS.Setup}(1^n, \mathbf{V})$.
2. Emulate $(m_1, m_2) \leftarrow \text{Sim}(x, \text{crs})$.
3. Output 1 if and only if $\text{FS.V}(\text{crs}, x, \pi := (m_1, m_2))$ accepts.

For $x \in \mathcal{L}$, by the completeness of the transformation, the interaction between $\mathbf{P}$ and $\mathbf{V}^*$ with $\text{aux} = \text{crs}$ results in a transcript that makes FS.V accept. Since D has access to the same $\text{aux} = \text{crs}$, the zero-knowledge property ensures that $\text{Sim}(x, \text{crs})$ must also produce a transcript that makes FS.V accept.

For $x \notin \mathcal{L}$, suppose the algorithm outputs 1 with non-negligible probability. In this case, an adversary could use the simulator to generate a proof that satisfies $\text{FS.V}(\text{crs}, \cdot)$, which contradicts the soundness of the transformed protocol.

Since this PPT algorithm decides the language, it follows that $\mathcal{L} \in \text{BPP}$.

3 Preliminaries

Notation. For an integer $n \in \mathbb{N}$, we denote the set $\{1, \dots, n\}$ by $[n]$. For a vector v of size n , for every $i \in [n]$, we denote by v_i the i -th component of v . We denote the all-zero and all-one vectors by $\vec{0}$ and $\vec{1}$, respectively. For strings $x, y \in \{0, 1\}^*$, we denote their concatenation by $x\|y$. Let $\mathcal{D}_1, \mathcal{D}_2$ be two distributions. We write $\mathcal{D}_1 \equiv_\epsilon \mathcal{D}_2$ if any (unbounded) adversary can distinguish between a sample from $\mathcal{D}_1$ and a sample from $\mathcal{D}_2$ with only ϵ advantage. We write $\mathcal{D}_1 \approx_\epsilon \mathcal{D}_2$ if any PPT adversary can distinguish between a sample from $\mathcal{D}_1$ and a sample from $\mathcal{D}_2$ with only ϵ advantage.

3.1 Distributions and Sampling

Definition 3.1 (Statistical Distance). *The statistical distance between two random variables X, Y is defined by*

$$\Delta(X, Y) := \frac{1}{2} \cdot \sum_x |\Pr[X = x] - \Pr[Y = x]|$$

We say that X and Y are δ -close (resp. -far) if $\Delta(X, Y) \leq \delta$ (resp. $\Delta(X, Y) \geq \delta$).

3.2 Pseudo-Random Function

Definition 3.2 (T -secure PRF [GGM19]). *Let $T : \mathbb{N} \rightarrow \mathbb{N}$ be a function. A T -secure pseudorandom function (PRF) is a couple of algorithms $\Pi_{\text{PRF}} = (\text{Gen}, \text{Eval})$ satisfying the following.*

- **Correctness.** *Eval is a deterministic algorithm. For every security parameter λ , input length $\ell_{\text{in}} \in \mathbb{N}$, and output length $\ell_{\text{out}} \in \mathbb{N}$, and for every key k generated by $\text{Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$, the function $\text{Eval}(k, \cdot)$ maps $\{0, 1\}^{\ell_{\text{in}}}$ to $\{0, 1\}^{\ell_{\text{out}}}$.*
- **T -Security.** *The PRF is T -secure if for every security parameter λ , for $T := T(\lambda)$, for every $\text{poly}(T)$ -size algorithm D ,*

$$\left| \Pr \left[D^{\text{Eval}(k, \cdot)} = 1 \mid k \leftarrow \text{Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \right] - \Pr \left[D^f = 1 \mid f \leftarrow U_{\ell_{\text{in}}, \ell_{\text{out}}} \right] \right| \leq \frac{1}{T} ,$$

where $U_{\ell_{\text{in}}, \ell_{\text{out}}}$ is a uniform distribution over functions $f : \{0, 1\}^{\ell_{\text{in}}} \rightarrow \{0, 1\}^{\ell_{\text{out}}}$.

Definition 3.3 (T -secure puncturable PRF [SW14]). *A T -secure puncturable pseudorandom function (PPRF) is a T -secure PRF with an addition algorithm Punc that satisfies the following algorithms. For every $\ell_{\text{in}}, \ell_{\text{out}}$, the Π_{PPRF} scheme satisfies the following properties,*

- **Puncturing Correctness.** *For security parameter λ , for every key k generated by $\text{Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$, it holds that puncturing k at location $x \in \{0, 1\}^{\ell_{\text{in}}}$ using $\text{Punc}(k, x)$ to get the punctured key $k\{x\}$, can still be used to evaluate the PRF at any point $x' \neq x$,*

$$\forall x \in \{0, 1\}^{\ell_{\text{in}}}, x' \neq x, \Pr \left[\text{Eval}(k\{x\}, x') = \text{Eval}(k, x') \mid \begin{array}{l} k \leftarrow \text{Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ k\{x\} \leftarrow \text{Punc}(k, x) \end{array} \right] = 1 .$$

- *T-Puncturing Security.* The PPRF is *T-secure* if for every security parameter λ , for $T := T(\lambda)$, for every $\text{poly}(T)$ -size algorithm D , and for every $x \in \{0, 1\}^{\ell_{\text{in}}}$,

$$\left| \Pr \left[D(k\{x\}, x, \text{Eval}(k, x)) = 1 \mid \begin{array}{l} k \leftarrow \text{Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ k\{x\} \leftarrow \text{Punc}(k, x) \end{array} \right] - \Pr \left[D(k\{x\}, x, u) = 1 \mid \begin{array}{l} k \leftarrow \text{Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ k\{x\} \leftarrow \text{Punc}(k, x) \\ u \leftarrow \{0, 1\}^{\ell_{\text{out}}} \end{array} \right] \right| \leq \text{negl}(T)$$

We say that a PRF is secure if it is λ -secure.

Definition 3.4 (Sub-Exponentially Secure PPRF). A puncturable pseudorandom function (PPRF) scheme is said to be sub-exponentially secure if there exists a constant $0 < c < 1$ such that the scheme is *T-secure* for $T(\lambda) \geq 2^{\lambda^c}$.

Theorem 3.5 (OWF Implies PRF [GGM86; HILL99]). Assuming sub-exponentially hard one-way functions, then there exists a sub-exponentially secure puncturable pseudorandom function.

3.3 Indistinguishability Obfuscation

Definition 3.6 (*T-Secure Indistinguishability Obfuscation* [BGI+01]). Let $T : \mathbb{N} \rightarrow \mathbb{N}$ be a function, and let iO be a PPT algorithm that takes as input a security parameter $\lambda \in \mathbb{N}$, a circuit size s , and a circuit C of size at most s . The iO algorithm is a *T-secure indistinguishability obfuscator* that satisfies the following properties.

- *Preserving Functionality.* For every security parameter $\lambda \in \mathbb{N}$, a circuit size s , a circuit C of size at most s , and input x to the circuit,

$$\Pr \left[\widehat{C}(x) = C(x) \mid \widehat{C} \leftarrow iO(1^\lambda, 1^s, C) \right] = 1 .$$

- *T-Security.* We say that iO is *T-secure* if for every security parameter $\lambda \in \mathbb{N}$, every circuit size bound s , and every pair of circuits C_1, C_2 of size $|C_1|, |C_2| \leq s$ such that $C_1 \equiv C_2$, it holds that for every algorithm D of size $\text{poly}(T(\lambda))$,

$$\left| \Pr \left[D(\widehat{C}_1) = 1 \mid \widehat{C}_1 \leftarrow iO(1^\lambda, 1^s, C_1) \right] - \Pr \left[D(\widehat{C}_2) = 1 \mid \widehat{C}_2 \leftarrow iO(1^\lambda, 1^s, C_2) \right] \right| \leq \text{negl}(T) .$$

We say that an iO is secure if it is λ -secure.

Definition 3.7 (Sub-Exponentially Secure Indistinguishability Obfuscation). An indistinguishability obfuscator iO algorithm is a sub-exponentially secure if there exists a constant $0 < c < 1$ such that the scheme is *T-secure* for $T(\lambda) \geq 2^{\lambda^c}$.

3.4 Input-Hiding Obfuscation

We define Input-Hiding Obfuscation (IHO)

Definition 3.8 (*s*-Evasive Circuit Collections [BCKP14]). Let $\mathcal{C} = \{C_k : \{0, 1\}^\ell \rightarrow \{0, 1\}\}_{k \in \mathbb{N}}$ be a collection of circuits. We say that $\mathcal{C}$ is *s*-evasive if for all $x \in \{0, 1\}^\ell$,

$$\Pr_{C_k \leftarrow \mathcal{C}} [C_k(x) \neq 0] \leq s.$$

Definition 3.9 (*T*-Secure Input-Hiding Obfuscation for *s*-Evasive Circuit Collection [BCKP14]). A *T*-secure Input-Hiding Obfuscation (IHO) for an *s*-evasive circuit collection $\mathcal{C}$ is a PPT algorithm that satisfies the following.

- Preserving Functionality. For every security parameter $\lambda \in \mathbb{N}$, a circuit $C_k \in \mathcal{C}$, and input x to the circuit,

$$\Pr \left[\hat{C}_k(x) = C(x) \mid \hat{C}_k \leftarrow \text{IHO}(1^\lambda, C) \right] = 1.$$

- *T*-Security. For every *T*-size adversary $\mathcal{A}$ and security parameter $\lambda \in \mathbb{N}$,

$$\Pr \left[\mathcal{A}(\hat{C}_k, z) \mid \begin{array}{l} C_k \leftarrow \mathcal{C} \\ \hat{C}_k \leftarrow \text{IHO}(1^\lambda, C_k) \end{array} \right] = s \cdot T \cdot \text{poly}(\lambda).$$

Note that we say that input-hiding obfuscation is secure if it is *T*-secure for any $T \leq \text{poly}(\lambda)$.

3.5 Interactive Proof Systems

Interactive proof systems. Following the framework of [GMR89], an interactive proof system for a language $\mathcal{L}$ is defined as a two-party protocol involving a prover and a verifier. Given a shared input x , the parties engage in a message exchange where the prover aims to prove the validity of the statement $x \in \mathcal{L}$ to the verifier. Formally an interactive proof is defined as follows.

Definition 3.10 (Private-Coin Interactive Proof System). A *k*-message public-coin interactive proof system for language $\mathcal{L}$ is a *k*-message protocol between the prover $\mathbf{P}$ and the verifier $\mathbf{V}$. The verifier $\mathbf{V}$ is computationally bounded and uses private randomness, and the prover $\mathbf{P}$ is unbounded. The protocol satisfies:

1. Completeness: For every $x \in \mathcal{L}$, the verifier $\mathbf{V}$ accepts with probability 1 after interacting with $\mathbf{P}$ on common input x :

$$\Pr_r[(\mathbf{P}, \mathbf{V}(r))(x) = 1] = 1.$$

2. ϵ -Soundness: For every $x \notin \mathcal{L}$ and every (unbounded) prover $\hat{\mathbf{P}}$,

$$\Pr_r[(\hat{\mathbf{P}}, \mathbf{V}(r))(x) = 1] \leq \epsilon.$$

In a 3-message private-coin protocol, we model the prover and the verifier as pairs of algorithms: $(\mathbf{P}_1, \mathbf{P}_2)$ and $(\mathbf{V}_1, \mathbf{V}_2)$. The algorithm $\mathbf{P}_1$ generates the first prover message and shares state (auxiliary information) with $\mathbf{P}_2$ that generates the second prover message. The algorithm $\mathbf{V}_1$ generates the challenge, while $\mathbf{V}_2$ decides whether to accept or reject the proof. Both verifier algorithms share the private randomness ρ . The protocol then follows the following steps:

1. $(m_1, \text{aux}) \leftarrow \mathbf{P}_1(x)$.

2. $\mathbf{a} \leftarrow \mathbf{V}_1(x, m_1; \rho)$.
3. $m_2 \leftarrow \mathbf{P}_2(x, \mathbf{aux}, \mathbf{a})$.
4. $b \leftarrow \mathbf{V}_2(x, m_1, m_2; \rho)$.

Definition 3.11 (Honest-Verifier Statistical Zero-Knowledge). *An interactive proof system $(\mathbf{P}, \mathbf{V})$ for language $\mathcal{L}$ is honest-verifier statistical zero-knowledge if there exists a PPT simulator $\text{Sim}(\cdot)$ such that*

$$\{\text{View}_{\mathbf{V}}(\mathbf{P}, \mathbf{V})(x)\}_{x \in \mathcal{L}} \equiv_{\epsilon} \{\text{Sim}^{\mathbf{V}}(x)\}_{x \in \mathcal{L}}$$

where $\equiv_{\epsilon}$ denotes ϵ statistical indistinguishability.

Definition 3.12 ((T, ϵ) -secure Non-Interactive Argument). *A (T, ϵ) -secure non-interactive argument (NARG) for a language $\mathcal{L}$ consists of a tuple of PPT algorithms $(\text{Setup}, \text{Prove}, \text{Verify})$ satisfying the following properties.*

- Completeness. *For every security parameter $\lambda \in \mathbb{N}$, every instance size $n \in \mathbb{N}$, and every instance $x \in \mathcal{L}$,*

$$\Pr \left[\text{Verify}(\text{crs}, x, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^{\lambda}, 1^n) \\ \pi \leftarrow \text{Prove}(\text{crs}, x) \end{array} \right] \geq 1 - \text{negl}(\lambda) .$$

- ϵ -Adaptive Soundness. *For every T -size adversary Adv , and for every security parameter $\lambda \in \mathbb{N}$,*

$$\Pr \left[\begin{array}{l} x \notin \mathcal{L} \\ \text{Verify}(\text{crs}, x, \pi) = 1 \end{array} \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^{\lambda}, 1^n) \\ (x, \pi) \leftarrow \text{Adv}(\text{crs}) \end{array} \right] \leq \epsilon(T, \lambda) .$$

We say that the scheme is polynomially secure if for every T -size adversary, where $T \leq \text{poly}(\lambda)$, there exists some negligible function such that the scheme is $\epsilon(T, \lambda) = \text{negl}(\lambda)$ secure.

Definition 3.13 (Honest-Verifier Statistical Zero-Knowledge Non-Interactive Argument). *A Zero-Knowledge Non-Interactive Argument (zkNARG) for a language $\mathcal{L}$ is a NARG (Definition 3.12) that satisfies the following additional property.*

- Statistical Zero-Knowledge. *There exists a PPT simulator Sim such that for every security parameter $\lambda \in \mathbb{N}$, instance size $n \in \mathbb{N}$, instance $x \in \mathcal{L} \cap \{0, 1\}^n$, and every distinguisher D (even unbounded), there exists a negligible function negl such that:*

$$\left| \Pr \left[\text{D}(\text{crs}, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^{\lambda}, 1^n) \\ \pi \leftarrow \mathbf{P}(\text{crs}, x) \end{array} \right] - \Pr \left[\text{D}(\text{crs}, \pi) = 1 \mid (\text{crs}, \pi) \leftarrow \text{Sim}(1^{\lambda}, x) \right] \right| \leq \text{negl}(\lambda) .$$

Definition 3.14 (Honest-Verifier Computational Zero-Knowledge Non-Interactive Argument). *A Zero-Knowledge Non-Interactive Argument (zkNARG) for a language $\mathcal{L}$ is a NARG (Definition 3.12) that satisfies the following additional property.*

- Computational Zero-Knowledge. *There exists a PPT simulator Sim such that for every security parameter $\lambda \in \mathbb{N}$, instance size $n \in \mathbb{N}$, instance $x \in \mathcal{L} \cap \{0, 1\}^n$, and every PPT distinguisher D , there exists a negligible function negl such that:*

$$\left| \Pr \left[\text{D}(\text{crs}, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^{\lambda}, 1^n) \\ \pi \leftarrow \mathbf{P}(\text{crs}, x) \end{array} \right] - \Pr \left[\text{D}(\text{crs}, \pi) = 1 \mid (\text{crs}, \pi) \leftarrow \text{Sim}(1^{\lambda}, x) \right] \right| \leq \text{negl}(\lambda) .$$

4 A Fiat–Shamir Transformation for Private-Coin Protocols

In this section, we present our main contribution: a Fiat–Shamir transformation for 3-message private-coin protocols. Assuming the existence of input-hiding obfuscation and that the underlying private-coin protocol has a sufficiently small soundness error (which can be achieved using standard amplification), the transformation yields a non-interactive argument with adaptive soundness. The construction appears in Section 4.1, followed by the security analysis in Section 4.2. In Section 4.3, we show that if the interactive protocol was honest-verifier statistical zero-knowledge, then the non-interactive protocol generated by the transformation is honest-verifier computational zero-knowledge.

Theorem 4.1. *Let Π be a 3-message private-coin protocol for instances of size n , with first prover message of size ℓ_1 , and soundness error ϵ .*

For every $T \leq 2^{n+\ell_1} \cdot \text{poly}(\lambda)$, assuming the existence of a sub-exponentially secure puncturable pseudorandom function, sub-exponentially secure indistinguishability obfuscation, and T -input-hiding obfuscation, then for some polynomial function poly , the protocol generated by applying the Fiat–Shamir transformation described in Section 4.1 on Π is (ϵ', T) -secure non-interactive argument (as defined in Definition 3.12) for $\epsilon'(T, \lambda) \leq 2^{(n+\ell_1)} \cdot \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) + \text{negl}(\lambda)$.

4.1 The Transformation

Let $(\mathbf{P}, \mathbf{V})$ be a 3-message private-coin protocol for some language $\mathcal{L}$. We think of the prover and the verifier as two pairs of algorithms: $(\mathbf{P}_1, \mathbf{P}_2)$ and $(\mathbf{V}_1, \mathbf{V}_2)$. For instance size n , the protocol has the following properties: soundness error ϵ , first prover message of size ℓ_1 , second prover message, verifier uses r bits of randomness, verifier runs in $\text{poly}(n)$ time.

Let iO be an indistinguishability obfuscator, and let $\Pi_{\text{PRF}} = (\text{F.Gen}, \text{F.Eval}, \text{F.Punc})$ be a puncturable pseudorandom function. For a key k and input x , denote the PRF evaluation by $F_k(x) := \text{F.Eval}(k, x)$. Let $0 < c < 1$ be a constant that we will define later in the proof. Let $s(\lambda, n)$ denote the maximum size of the circuits in the following hybrids. We define this bound to ensure that iO outputs the same circuit size across all hybrids. This maximum size is bounded by $\text{poly}(\lambda, n)$.

Fiat–Shamir transformation for $(\mathbf{P}, \mathbf{V})$	
Chal $[k](x, m_1)$: 1. Compute $\rho \leftarrow F_k(x \ m_1)$. 2. Output $a \leftarrow \mathbf{V}_1(x, m_1; \rho)$.	Ver $[k](x, m_1, m_2)$: 1. Compute $\rho \leftarrow F_k(x \ m_1)$. 2. Output $\mathbf{V}_2(x, m_1, m_2; \rho)$.
Setup $(1^\lambda, \mathbf{V})$: 1. Set $\lambda_{\text{PRF}} := (2\lambda + 2n + 2\ell_1)^{\frac{1}{c}}$, $\ell_{\text{in}} := n + \ell_1$, and $\ell_{\text{out}} := r$. 2. Compute $k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$. 3. Set $\lambda_{\text{iO}} := (2\lambda + 2n + 2\ell_1)^{\frac{1}{c}}$. 4. Compute $\widehat{\text{Chal}}[k] \leftarrow \text{iO}(1^{\lambda_{\text{iO}}}, 1^s, \text{Chal}[k])$. 5. Compute $\widehat{\text{Ver}}[k] \leftarrow \text{iO}(1^{\lambda_{\text{iO}}}, 1^s, \text{Ver}[k])$.	

6. Output $\text{crs} := (\widehat{\text{Chal}}[\mathbf{k}], \widehat{\text{Ver}}[\mathbf{k}])$.	
Prove (crs, x):	Verify (crs, x, π):
1. Parse crs as $(\widehat{\text{Chal}}[\mathbf{k}], \widehat{\text{Ver}}[\mathbf{k}])$.	1. Parse crs as $(\widehat{\text{Chal}}[\mathbf{k}], \widehat{\text{Ver}}[\mathbf{k}])$.
2. Compute,	2. Parse π as (m_1, m_2) .
1. $(m_1, \text{aux}) \leftarrow \mathbf{P}_1(x)$.	3. Output $\widehat{\text{Ver}}[\mathbf{k}](x, m_1, m_2)$.
2. $\mathbf{a} \leftarrow \widehat{\text{Chal}}[\mathbf{k}](x, m_1)$.	
3. $m_2 \leftarrow \mathbf{P}_2(x, \text{aux}, \mathbf{a})$.	
3. Output $\pi := (m_1, m_2)$.	

4.2 Adaptive Soundness

In this section we prove that the transformation described in Section 4.1 generates (ϵ', T) -secure non-interactive argument (as defined in Definition 3.12) for every $T \leq 2^{n+\ell_1} \cdot \text{poly}(\lambda)$, and $\epsilon'(T, \lambda) \leq 2^{(n+\ell_1)} \cdot \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) + \text{negl}(\lambda)$.

Assume sub-exponentially secure indistinguishability obfuscator iO , IHO for $\sqrt{\epsilon}$ -sparse circuit families, and sub-exponentially secure puncturable pseudorandom function $\Pi_{\text{PPRF}} = (\text{F.Gen}, \text{F.Eval}, \text{F.Punc})$. For a key $\mathbf{k}$ and input x , we denote the PRF evaluation by $F_{\mathbf{k}}(x) := \text{F.Eval}(\mathbf{k}, x)$. Fix some constant $0 < c < 1$ such that the iO and PPRF schemes are 2^{λ^c} -secure. (There exists such a constant by the sub-exponential security of the schemes.)

Let $\tilde{\mathbf{P}}$ be a T -size adversary for $T \leq 2^{n+\ell_1} \cdot \text{poly}(\lambda)$, and let $\nu(\lambda)$ be its advantage.

$$\nu(\lambda) := \Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ \widehat{\text{Ver}}[\mathbf{k}](x, m_1, m_2) = 1 \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (\widehat{\text{Chal}}[\mathbf{k}], \widehat{\text{Ver}}[\mathbf{k}]) := \text{crs} \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right]. \quad (1)$$

Let ACC be the following set,

$$\text{ACC} := \{ (x, m_1, \rho, m_2) \mid \mathbf{V}_2(x, m_1, m_2; \rho) = 1 \}.$$

Therefore, by construction and by Equation 1, we get that,

$$\nu(\lambda) = \Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ (x, m_1, F_{\mathbf{k}}(x \| m_1), m_2) \in \text{ACC} \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right]. \quad (2)$$

where $\mathbf{k}$ is the PRF key generated by the setup. We continue the proof with a sequence of indistinguishable hybrids. To simplify the notation, we will omit the explicit security parameter and circuit size in the iO notation when they are clear from the context.

We define Hybrid 1 with respect to a fixed pair (x^*, m_1^*) . In the setup algorithm, we puncture the PRF key at the input $(x^* \| m_1^*)$ and replace the key $\mathbf{k}$ with the punctured key $\mathbf{k}^*$. We then pre-compute the PRF output for this specific input and embed it in the Ver circuit. For the Chal circuit, we compute the challenge relative to (x^*, m_1^*) and the pre-computed PRF values, and embed the challenge in the circuit.

Hybrid 1.

- $\text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V})$:
 1. Compute

- (a) $k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$.
 - (b) $\rho^* \leftarrow \text{F}_k(x^* \| m_1^*)$.
 - (c) $k^* \leftarrow \text{F.Punc}(k, (x^* \| m_1^*))$.
 - (d) $a^* \leftarrow \text{V}_1(x^*, m_1^*; \rho^*)$.
 - (e) $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k^*, x^*, m_1^*, a^*])$.
 - (f) $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k^*, x^*, m_1^*, \rho^*])$.
2. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

- $\text{Chal}[k^*, x^*, m_1^*, a^*](x, m_1)$:
 1. If $(x^*, m_1^*) = (x, m_1)$, output a^* .
 2. Else, compute $\rho \leftarrow \text{F}_{k^*}(x \| m_1)$ and output $\text{V}_1(x, m_1; \rho)$.
- $\text{Ver}[k^*, x^*, m_1^*, \rho^*](x, m_1, m_2)$:
 1. If $(x^*, m_1^*) = (x, m_1)$, output $\text{V}_2(x^*, m_1^*, m_2; \rho^*)$.
 2. Else, compute $\rho \leftarrow \text{F}_{k^*}(x \| m_1)$ and output $\text{V}_2(x, m_1, m_2; \rho)$.

Claim 4.2. For every fixed $x^* \in \{0, 1\}^n$ and $m_1^* \in \{0, 1\}^{\ell_1}$, it holds that,

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, \text{F}_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, \text{F}_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - 2 \cdot 2^{-2(\lambda+n+\ell_1)},$$

where k is the PRF key generated by the setup.

Proof. Note that for every fixed key k , the circuits $(\text{Chal}[k], \text{Ver}[k])$ generated by Setup and the circuits $(\text{Chal}[k^*, x^*, m_1^*, a^*], \text{Ver}[k^*, x^*, m_1^*, \rho^*])$ generated by Setup_1 have exactly the same functionality. Therefore, the claim follows from the security of the iO scheme.

Formally, by the definition of Setup_1 ,

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, \text{F}_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ = \Pr \left[\begin{array}{c} (x^*, m_1^*, \text{F}_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ \rho^* \leftarrow \text{F}_k(x^* \| m_1^*) \\ k^* \leftarrow \text{F.Punc}(k, (x^* \| m_1^*)) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k^*, x^*, m_1^*, \rho^*]) \\ \widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k^*, x^*, m_1^*, \rho^*]) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\widehat{\text{Chal}}, \widehat{\text{Ver}}) \end{array} \right].$$

By sub-exponential security of the iO, initiated with security parameter $\lambda_{\text{iO}} = (2\lambda + 2n + 2\ell_1)^{\frac{1}{c}}$, and since the event above is checkable in time $\text{poly}(n)$ using the original protocol's verifier and the

prover $\tilde{\mathbf{P}}$ can be simulated in time $T \leq 2^{n+\ell_1} \cdot \text{poly}(\lambda)$,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ \rho^* \leftarrow F_k(x^* \| m_1^*) \\ k^* \leftarrow \text{F.Punc}(k, (x^* \| m_1^*)) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k^*, x^*, m_1^*, \rho^*]) \\ \widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k^*, x^*, m_1^*, \rho^*]) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\widehat{\text{Chal}}, \widehat{\text{Ver}}) \end{array} \right] \\ & \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k]) \\ \widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k]) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\widehat{\text{Chal}}, \widehat{\text{Ver}}) \end{array} \right] - 2 \cdot 2^{-2(\lambda+n+\ell_1)} . \end{aligned}$$

By the definition of **Setup**,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k]) \\ \widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k]) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\widehat{\text{Chal}}, \widehat{\text{Ver}}) \end{array} \right] \\ & = \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] . \end{aligned}$$

Overall, by combining the above equations, we get the required bound:

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \text{crs} \leftarrow \text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - 2 \cdot 2^{-2(\lambda+n+\ell_1)} . \end{aligned}$$

□

We define Hybrid 2 with respect to a fixed (x^*, m_1^*, ρ^*) as follows.

Hybrid 2.

- $\text{Setup}_2[x^*, m_1^*, \rho^*](1^\lambda, \mathbf{V})$:
 1. Compute
 - (a) $k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$.
 - (b) ~~$\rho^* \leftarrow F_k(x^* \| m_1^*)$.~~
 - (c) $k^* \leftarrow \text{F.Punc}(k, (x^* \| m_1^*))$.
 - (d) $a^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho^*)$.
 - (e) $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[k^*, x^*, m_1^*, a^*])$.
 - (f) $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[k^*, x^*, m_1^*, \rho^*])$.

2. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.
- $\text{Chal}[\mathbf{k}^*, x^*, m_1^*, \mathbf{a}^*](x, m_1)$:
 1. If $(x^*, m_1^*) = (x, m_1)$, output $\mathbf{a}^*$.
 2. Else, compute $\rho \leftarrow \mathbf{F}_{\mathbf{k}^*}(x \| m_1)$ and output $\mathbf{V}_1(x, m_1; \rho)$.
 - $\text{Ver}[\mathbf{k}^*, x^*, m_1^*, \rho^*](x, m_1, m_2)$:
 1. If $(x^*, m_1^*) = (x, m_1)$, output $\mathbf{V}_2(x^*, m_1^*, m_2; \rho^*)$.
 2. Else, compute $\rho \leftarrow \mathbf{F}_{\mathbf{k}^*}(x \| m_1)$ and output $\mathbf{V}_2(x, m_1, m_2; \rho)$.

Claim 4.3. For every fixed $x^* \in \{0, 1\}^n$ and $m_1^* \in \{0, 1\}^{\ell_1}$, it holds that,

$$\Pr \left[\begin{array}{c|c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} & \rho^* \leftarrow \{0, 1\}^r \\ \wedge & \text{crs} \leftarrow \text{Setup}_2[x^*, m_1^*, \rho^*](1^\lambda, \mathbf{V}) \\ (x \| m_1) = (x^* \| m_1^*) & (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ \geq \Pr \left[\begin{array}{c|c} (x^*, m_1^*, \mathbf{F}_{\mathbf{k}}(x^* \| m_1^*), m_2) \in \text{ACC} & \text{crs} \leftarrow \text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V}) \\ \wedge & (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \\ (x \| m_1) = (x^* \| m_1^*) & \end{array} \right] - 2^{-2(\lambda+n+\ell_1)},$$

where $\mathbf{k}$ is the PRF key generated by the setup.

Proof. The claim holds by sub-exponential security of the puncturable PRF, initiated with security parameter $\lambda_{\text{PRF}} = (2\lambda + 2n + 2\ell_1)^{\frac{1}{c}}$, and since the event above is checkable in time $\text{poly}(n)$ using the original protocol's verifier and the prover $\tilde{\mathbf{P}}$ can be simulated in time $T \leq 2^{n+\ell_1} \cdot \text{poly}(\lambda)$. $\square$

- Let $\mathcal{D}^{x, m_1, \mathbf{a}}$ be the set of all randomnesses ρ such the for $\mathbf{a} = \mathbf{V}_1(x, m_1; \rho)$. Formally,

$$\mathcal{D}^{x, m_1, \mathbf{a}} := \{ \rho \in \{0, 1\}^r \mid \mathbf{a} = \mathbf{V}_1(x, m_1; \rho) \}.$$

- Let $\mathcal{C}^{x, m_1, \mathbf{a}} := \{C_\rho : \{0, 1\}^{\ell_2} \rightarrow \{0, 1\}\}_{\rho \leftarrow \mathcal{D}^{x, m_1, \mathbf{a}}}$ be a circuit family, where every circuit C_ρ is defined as follow.

$$C_\rho(m_2) := \mathbf{V}_2(x, m_1, m_2; \rho).$$

- Let $s^{x, m_1, \mathbf{a}}$ be the evasiveness of the circuit family $\mathcal{C}^{x, m_1, \mathbf{a}}$. By Definition 3.8, that is,

$$s^{x, m_1, \mathbf{a}} := \max_{m_2} (\Pr[C_\rho(m_2) = 1 \mid C_\rho \leftarrow \mathcal{C}^{x, m_1, \mathbf{a}}]).$$

We define Hybrid 2 with respect to a fixed $(x^*, m_1^*, \mathbf{a}^*, \widehat{C}_{\rho^*})$, where $\widehat{C}_{\rho^*}$ is the obfuscation of the circuit $C_{\rho^*} \in \mathcal{C}^{x, m_1, \mathbf{a}}$.

Hybrid 3.

- $\text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \widehat{C}_{\rho^*}](1^\lambda, \mathbf{V})$:
 1. Compute
 - (a) $\mathbf{k} \leftarrow \mathbf{F}.\text{Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$.
 - (b) $\mathbf{k}^* \leftarrow \mathbf{F}.\text{Punc}(\mathbf{k}, (x^* \| m_1^*))$.
 - (c) $\widehat{\text{Chal}} \leftarrow \text{iO}(\text{Chal}[\mathbf{k}^*, x^*, m_1^*, \mathbf{a}^*])$.
 - (d) $\widehat{\text{Ver}} \leftarrow \text{iO}(\text{Ver}[\mathbf{k}^*, x^*, m_1^*, \widehat{C}_{\rho^*}])$.

2. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

- $\text{Chal}[k^*, x^*, m_1^*, a^*](x, m_1)$:
 1. If $(x^*, m_1^*) = (x, m_1)$, output a^* .
 2. Else, compute $\rho \leftarrow F_{k^*}(x \| m_1)$ and output $a \leftarrow \mathbf{V}_1(x, m_1; \rho)$.
- $\text{Ver}[k^*, x^*, m_1^*, \widehat{C}_{\rho^*}](x, m_1, m_2)$:
 1. If $(x^*, m_1^*) = (x, m_1)$, output $\widehat{C}_{\rho^*}(m_2)$.
 2. Else, compute $\rho \leftarrow F_{k^*}(x \| m_1)$ and output $\mathbf{V}_2(x, m_1, m_2; \rho)$.

Claim 4.4. For every fixed $x^* \in \{0, 1\}^n$ and $m_1^* \in \{0, 1\}^{\ell_1}$, it holds that,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ a^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, a^*} \\ \widehat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, a^*, \widehat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \widehat{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \rho^* \leftarrow \{0, 1\}^r \\ \text{crs} \leftarrow \text{Setup}_2[x^*, m_1^*, \rho^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \widehat{\mathbf{P}}(\text{crs}) \end{array} \right] - 2^{-2(\lambda+n+\ell_1)} . \end{aligned}$$

Proof. The proof consists of two steps. First, we prove that the distribution of ρ^* is identical in both experiments. Second, we prove that conditioned on any fixed ρ^* , the circuits in Setup_2 and Setup_3 are functionally equivalent. Therefore, by the iO security, the hybrids are indistinguishable.

To initiate Setup_2 , we uniformly sample ρ^* . To initiate Setup_3 , we first sample ρ uniformly, then compute the corresponding challenge $a^* = \mathbf{V}_1(x^*, m_1^*; \rho)$. Then, by the definition of $\mathcal{C}^{x^*, m_1^*, a^*}$, sample ρ^* uniformly from the set $\mathcal{D}^{x^*, m_1^*, a^*}$. Fix some randomness $\rho' \in \{0, 1\}^r$. By the law of total probability, the probability to sample a specific $\rho' \in \{0, 1\}^r$ is the probability of sampling the challenge a^* such that $a^* = \mathbf{V}_1(x^*, m_1^*; \rho')$, which is $|\mathcal{D}^{x^*, m_1^*, a^*}| \cdot 2^{-r}$, times the probability to sample ρ' given that the challenge a^* was sampled, which is $1/|\mathcal{D}^{x^*, m_1^*, a^*}|$. Multiplying both probabilities cancels out $|\mathcal{D}^{x^*, m_1^*, a^*}|$, and we get 2^{-r} . Formally,

$$\begin{aligned} & \Pr \left[\rho^* = \rho' \middle| \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ a^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, a^*} \\ \widehat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, a^*, \widehat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \widehat{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & = |\mathcal{D}^{x^*, m_1^*, a'}| \cdot 2^{-r} \cdot \frac{1}{|\mathcal{D}^{x^*, m_1^*, a'}|} = 2^{-r} , \end{aligned}$$

where $a' := \mathbf{V}_1(x^*, m_1; \rho')$. Therefore, ρ^* is uniformly sampled in both experiments.

Next, we consider the functionality of the circuit Ver for a fixed ρ^* . In Setup_2 , the circuit uses the hardwired randomness to directly compute the predicate $\mathbf{V}_2(x^*, m_1^*, m_2; \rho^*)$. In Setup_3 , the circuit executes an obfuscated circuit $\widehat{C}_{\rho^*}$ instead, where $\widehat{C}_{\rho^*} = \mathbf{V}_2(x^*, m_1^*, m_2; \rho^*)$.

Since the functionality of the circuits is identical, we rely on the security of the iO scheme. Observe that the distinguishing event is checkable in time $\text{poly}(n)$ and the adversary $\tilde{\mathbf{P}}$ runs in time $T \leq 2^{n+\ell_1} \cdot \text{poly}(\lambda)$. Therefore, the sub-exponential security of the iO scheme (instantiated with parameter λ_{io}) guarantees that the distinguishing advantage is bounded by $2^{-(\lambda+n+\ell_1)}$. $\square$

We bound the success probability of the adversary on the last hybrid.

Claim 4.5. *For every $x^* \in \{0,1\}^n$ such that $x^* \notin \mathcal{L}$, and $m_1^* \in \{0,1\}^{\ell_1}$:*

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \rho \leftarrow \{0,1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \leq \sqrt{\epsilon} \cdot \text{poly}(\lambda) .$$

Proof. By the definition of the circuit $\mathcal{C}^{x^*, m_1^*, \mathbf{a}}$ and by the law of total probability, we get that,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \rho \leftarrow \{0,1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ &= \Pr \left[\begin{array}{c} C_{\rho^*}(m_2) = 1 \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \rho \leftarrow \{0,1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ &= \sum_{\mathbf{a}} \Pr \left[\mathbf{V}_1(x^*, m_1^*; \rho) = \mathbf{a} \mid \rho \leftarrow \{0,1\}^r \right] \\ &\quad \cdot \Pr \left[\begin{array}{c} C_{\rho^*}(m_2) = 1 \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] . \end{aligned} \tag{3}$$

Let $p^{x^*, m_1^*, \mathbf{a}}$ be the following probability,

$$p^{x^*, m_1^*, \mathbf{a}} := \Pr[\mathbf{V}_1(x^*, m_1^*; \rho) = \mathbf{a} \mid \rho \leftarrow \{0,1\}^r] .$$

Let S be the set of all challenges $\mathbf{a}$ such that the circuit family $\mathcal{C}^{x^*, m_1^*, \mathbf{a}}$ has evasiveness smaller than $\sqrt{\epsilon}$ (i.e, $s^{x^*, m_1^*, \mathbf{a}} < \sqrt{\epsilon}$), and let $\bar{S}$ be the set of all challenges $\mathbf{a}$ such that the circuit family

$\mathcal{C}^{x^*, m_1^*, \mathbf{a}}$ has evasiveness larger than $\sqrt{\epsilon}$ (i.e, $s^{x^*, m_1^*, \mathbf{a}} \geq \sqrt{\epsilon}$). By Equation 3, we get that,

$$\begin{aligned}
& \Pr \left[\begin{array}{c|l} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} & \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \\ \wedge & \\ (x \| m_1) = (x^* \| m_1^*) & \end{array} \right] \\
&= \sum_{\mathbf{a}} p^{x^*, m_1^*, \mathbf{a}} \cdot \Pr \left[\begin{array}{c|l} C_{\rho^*}(m_2) = 1 & \begin{array}{l} C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \\ \wedge & \\ (x \| m_1) = (x^* \| m_1^*) & \end{array} \right] \\
&= \sum_{\mathbf{a}} p^{x^*, m_1^*, \mathbf{a}} \cdot \Pr \left[\begin{array}{c|l} C_{\rho^*}(m_2) = 1 & \begin{array}{l} C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \\ \wedge & \\ (x \| m_1) = (x^* \| m_1^*) & \end{array} \right] \\
&= \sum_{\mathbf{a} \in \mathcal{S}} p^{x^*, m_1^*, \mathbf{a}} \cdot \Pr \left[\begin{array}{c|l} C_{\rho^*}(m_2) = 1 & \begin{array}{l} C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \\ \wedge & \\ (x \| m_1) = (x^* \| m_1^*) & \end{array} \right] \\
&+ \sum_{\mathbf{a} \in \bar{\mathcal{S}}} p^{x^*, m_1^*, \mathbf{a}} \cdot \Pr \left[\begin{array}{c|l} C_{\rho^*}(m_2) = 1 & \begin{array}{l} C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \\ \wedge & \\ (x \| m_1) = (x^* \| m_1^*) & \end{array} \right] \\
&\leq \sum_{\mathbf{a} \in \mathcal{S}} p^{x^*, m_1^*, \mathbf{a}} \cdot \Pr \left[\begin{array}{c|l} C_{\rho^*}(m_2) = 1 & \begin{array}{l} C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \\ \wedge & \\ (x \| m_1) = (x^* \| m_1^*) & \end{array} \right] \\
&+ \sum_{\mathbf{a} \in \bar{\mathcal{S}}} p^{x^*, m_1^*, \mathbf{a}} \cdot \mathbf{1} ,
\end{aligned}$$

By the above, we get that,

$$\begin{aligned}
& \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\
& \leq \sum_{\mathbf{a} \in S} p^{x^*, m_1^*, \mathbf{a}} \cdot \Pr \left[\begin{array}{c} C_{\rho^*}(m_2) = 1 \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] + \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \\
& \leq \sum_{\mathbf{a} \in S} p^{x^*, m_1^*, \mathbf{a}} \cdot s^{x^*, m_1^*, \mathbf{a}} \cdot T \cdot \text{poly}(\lambda) + \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} ,
\end{aligned} \tag{5}$$

where the second inequality is by the IHO security of the circuit family $\mathcal{C}^{x^*, m_1^*, \mathbf{a}}$ against adversaries of size $|\tilde{\mathbf{P}}| + \text{poly}(\lambda) = T + \text{poly}(\lambda)$. By the definition of the set S that includes only challenges $\mathbf{a}$ such that $s^{x^*, m_1^*, \mathbf{a}} \leq \sqrt{\epsilon}$, we get that,

$$\begin{aligned}
& \sum_{\mathbf{a} \in S} p^{x^*, m_1^*, \mathbf{a}} \cdot s^{x^*, m_1^*, \mathbf{a}} \cdot T \cdot \text{poly}(\lambda) + \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \\
& \leq \sum_{\mathbf{a} \in S} p^{x^*, m_1^*, \mathbf{a}} \cdot \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) + \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \\
& \leq \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) + \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} .
\end{aligned} \tag{6}$$

Combining Equation 5 and Equation 6, to conclude the proof it suffices to show that

$$\sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \leq \sqrt{\epsilon}$$

If this holds, the success probability is bounded by,

$$\begin{aligned}
& \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\
& \leq \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) + \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \leq \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) + \sqrt{\epsilon} \leq \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) ,
\end{aligned}$$

as required. We now prove the following claim, which concludes the proof of Claim 4.5.

Claim 4.6.

$$\sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \leq \sqrt{\epsilon} .$$

Proof. We start by proving that

$$\sum_{\mathbf{a}} p^{x^*, m_1^*, \mathbf{a}} \cdot s^{x^*, m_1^*, \mathbf{a}} \leq \epsilon .$$

Let $\tilde{\mathbf{P}}$ be the following prover to the original protocol.

1. Send m_1^* to the verifier.
2. Receive $\mathbf{a}$ from the verifier.
3. Send $m_2 = \operatorname{argmax}_{m_2} (\Pr[C_{\rho^*}(m_2) = 1 \mid C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}}])$.

By the law of total probability, the success probability of the above adversary is as follows,

$$\Pr[\langle \tilde{\mathbf{P}}, \mathbf{V} \rangle(x^*) = 1] \tag{7}$$

$$\begin{aligned} &= \Pr \left[\mathbf{V}_2(x^*, m_1^*, m_2; \rho) = 1 \mid \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ m_2 \leftarrow \tilde{\mathbf{P}}(\mathbf{a}) \end{array} \right] \\ &= \sum_{\mathbf{a}} \Pr \left[\mathbf{a}^* = \mathbf{a} \mid \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \end{array} \right] . \end{aligned} \tag{8}$$

$$\begin{aligned} &\Pr \left[\begin{array}{l} \mathbf{V}_2(x^*, m_1^*, m_2; \rho) = 1 \\ \text{conditioned on} \\ \mathbf{a}^* = \mathbf{a} \end{array} \mid \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ m_2 \leftarrow \tilde{\mathbf{P}}(\mathbf{a}) \end{array} \right] \\ &= \sum_{\mathbf{a}} \Pr \left[\mathbf{V}_1(x^*, m_1^*; \rho) = \mathbf{a} \mid \rho \leftarrow \{0, 1\}^r \right] . \end{aligned} \tag{9}$$

$$\Pr \left[\begin{array}{l} \mathbf{V}_2(x^*, m_1^*, m_2; \rho) = 1 \\ \text{conditioned on} \\ \mathbf{V}_1(x^*, m_1^*; \rho) = \mathbf{a} \end{array} \mid \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ \mathbf{a} \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ m_2 \leftarrow \tilde{\mathbf{P}}(\mathbf{a}) \end{array} \right] .$$

For every challenge $\mathbf{a}$, let $m_2^{\mathbf{a}}$ be the following second prover message:

$$m_2^{\mathbf{a}} := \operatorname{argmax}_{m_2} \left(\Pr[C_{\rho^*}(m_2) = 1 \mid C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}}] \right) ,$$

By Equation 7 and by the definition of $\tilde{\mathbf{P}}$,

$$\begin{aligned} &\Pr[\langle \tilde{\mathbf{P}}, \mathbf{V} \rangle(x^*) = 1] \\ &= \sum_{\mathbf{a}} p^{x^*, m_1^*, \mathbf{a}} \cdot \Pr \left[\begin{array}{l} \mathbf{V}_2(x^*, m_1^*, m_2^{\mathbf{a}}; \rho) = 1 \\ \text{conditioned on} \\ \mathbf{V}_1(x^*, m_1^*; \rho) = \mathbf{a} \end{array} \mid \begin{array}{l} \rho \leftarrow \{0, 1\}^r \\ \mathbf{a} \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \end{array} \right] \\ &= \sum_{\mathbf{a}} p^{x^*, m_1^*, \mathbf{a}} \cdot \Pr \left[C_{\rho^*}(m_2^{\mathbf{a}}) = 1 \mid C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}} \right] . \end{aligned}$$

By the above and by the definition of $m_2^{\mathbf{a}}$,

$$\begin{aligned} &\Pr[\langle \tilde{\mathbf{P}}, \mathbf{V} \rangle(x^*) = 1] \\ &= \sum_{\mathbf{a}} p^{x^*, m_1^*, \mathbf{a}} \cdot \max_{m_2} \left(\Pr[C_{\rho^*}(m_2) = 1 \mid C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}}] \right) \\ &= \sum_{\mathbf{a}} p^{x^*, m_1^*, \mathbf{a}} \cdot s^{x^*, m_1^*, \mathbf{a}} \leq \epsilon , \end{aligned}$$

where the last inequality is by the soundness of the original protocol. Therefore,

$$\begin{aligned} \epsilon &\geq \sum_{\mathbf{a}} p^{x^*, m_1^*, \mathbf{a}} \cdot s^{x^*, m_1^*, \mathbf{a}} = \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \cdot s^{x^*, m_1^*, \mathbf{a}} + \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \cdot s^{x^*, m_1^*, \mathbf{a}} \\ &\geq \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \cdot s^{x^*, m_1^*, \mathbf{a}} \geq \sqrt{\epsilon} \cdot \sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} , \end{aligned}$$

where the last inequality is by the definition of the set $\bar{S}$. By rearranging the terms, we obtain,

$$\sum_{\mathbf{a} \in \bar{S}} p^{x^*, m_1^*, \mathbf{a}} \geq \sqrt{\epsilon} ,$$

as required. □

□

By union bound, we get that,

$$\begin{aligned} &\Pr \left[\begin{array}{l} x \notin \mathcal{L} \\ (x, m_1, F_k(x \| m_1), m_2) \in \text{ACC} \end{array} \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ &= \sum_{\substack{x' \in \{0,1\}^n \\ m'_1 \in \{0,1\}^{\ell_1}}} \Pr \left[\begin{array}{l} x \notin \mathcal{L} \\ \wedge \\ (x, m_1, F_k(x \| m_1), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x' \| m'_1) \end{array} \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] . \end{aligned}$$

By Equation 2,

$$\Pr \left[\begin{array}{l} x \notin \mathcal{L} \\ (x, m_1, F_k(x \| m_1), m_2) \in \text{ACC} \end{array} \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] = \nu(\lambda) .$$

Therefore, by an averaging argument, we get that,

$$\max_{\substack{x' \in \{0,1\}^n \\ m'_1 \in \{0,1\}^{\ell_1}}} \Pr \left[\begin{array}{l} x \notin \mathcal{L} \\ \wedge \\ (x, m_1, F_k(x \| m_1), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x' \| m'_1) \end{array} \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \geq \frac{\nu(\lambda)}{2^{n+\ell_1}} .$$

Note that for every fixed x', m'_1 ,

$$\begin{aligned} &\Pr \left[\begin{array}{l} x \notin \mathcal{L} \\ \wedge \\ (x, m_1, F_k(x \| m_1), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x' \| m'_1) \end{array} \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ &= \Pr \left[\begin{array}{l} x' \notin \mathcal{L} \\ \wedge \\ (x', m'_1, F_k(x' \| m'_1), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x' \| m'_1) \end{array} \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \end{aligned}$$

Therefore,

$$\max_{\substack{x' \in \{0,1\}^n \\ m'_1 \in \{0,1\}^{\ell_1}}} \Pr \left[\begin{array}{c} x' \notin \mathcal{L} \\ \wedge \\ (x', m'_1, F_k(x' \| m'_1), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x' \| m'_1) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \geq \frac{\nu(\lambda)}{2^{n+\ell_1}} .$$

By the above, there exists some $x^* \notin \mathcal{L}$, and m_1^* such that

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \geq \frac{\nu(\lambda)}{2^{n+\ell_1}} .$$

By Claim 4.2 to Claim 4.3, we get that,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} \rho \leftarrow \{0,1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & - 4 \cdot 2^{-2(\lambda+n+\ell_1)} . \end{aligned}$$

By the above two equations, we get that,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} \rho \leftarrow \{0,1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & \geq \frac{\nu(\lambda)}{2^{(n+\ell_1)}} - 4 \cdot 2^{-2(\lambda+n+\ell_1)} . \end{aligned}$$

By Claim 4.5,

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} \rho \leftarrow \{0,1\}^r \\ \mathbf{a}^* \leftarrow \mathbf{V}_1(x^*, m_1^*; \rho) \\ C_{\rho^*} \leftarrow \mathcal{C}^{x^*, m_1^*, \mathbf{a}^*} \\ \hat{C}_{\rho^*} \leftarrow \text{IHO}(C_{\rho^*}) \\ \text{crs} \leftarrow \text{Setup}_3[x^*, m_1^*, \mathbf{a}^*, \hat{C}_{\rho^*}](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \leq \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) .$$

By combining the above two equations, we get that,

$$\frac{\nu(\lambda)}{2^{(n+\ell_1)}} - 4 \cdot 2^{-2(\lambda+n+\ell_1)} \leq \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) .$$

By moving the terms, we get the required bound,

$$\nu(\lambda) \leq 2^{(n+\ell_1)} \cdot \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) + 2^{-(\lambda+n+\ell_1)} = 2^{(n+\ell_1)} \cdot \sqrt{\epsilon} \cdot T \cdot \text{poly}(\lambda) + \text{negl}(\lambda) .$$

4.3 Zero-Knowledge in the Honest-Verifier Setting

In this section, we prove that if a 3-message protocol $\Pi = (\mathbf{P}, \mathbf{V})$ satisfies statistically zero-knowledge in the honest-verifier setting as defined in Definition 3.11, then the non-interactive protocol generated by our Fiat-Shamir transformation in Section 4.1 satisfies honest-verifier zero-knowledge as defined in Definition 3.14.

Theorem 4.7. *Assume the existence of a secure puncturable pseudorandom function and secure indistinguishability obfuscation. If Π is a 3-message private-coin protocol that satisfies statistical zero-knowledge in the honest-verifier setting as defined in Definition 3.11, then the non-interactive protocol generated by applying the Fiat-Shamir transformation from Section 4.1 to Π satisfies computational honest-verifier zero-knowledge as defined in Definition 3.14.*

Proof. Let $\Pi = (\mathbf{P} = (\mathbf{P}_1, \mathbf{P}_2), \mathbf{V} = (\mathbf{V}_1, \mathbf{V}_2))$ be a 3-message private-coin protocol for the language $\mathcal{L}$ that satisfies statistical zero-knowledge in the honest-verifier setting as defined in Definition 3.11. Let Sim be its PPT simulator. Note that Sim outputs the view of the verifier, which is the prover messages (m_1, m_2) , together with the verifier's randomness ρ . Let $\Pi' = (\text{FS.Setup}, \text{FS.Prove}, \text{FS.Verify})$ be the non-interactive protocol for $\mathcal{L}$ generated by applying the Fiat-Shamir transformation described in Section 4.1 on Π . We construct a PPT simulator Sim' and prove that its output distribution is computationally indistinguishable.

We define the simulator algorithm as follows:

- $\text{Sim}'(1^\lambda, x)$:
 1. Emulate $(m_1, \rho, m_2) \leftarrow \text{Sim}(x)$.
 2. Compute $\text{crs} \leftarrow \text{Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V})$.
(where Setup_2 is the setup defined in Hybrid 2 in Section 4.2)
 3. Output $(\text{crs}, \pi := (m_1, m_2))$.

For instance size $n \in \mathbb{N}$, fix an instance $x \in \mathcal{L} \cap \{0, 1\}^n$. Let $\mathcal{D}_{\text{Sim}'}$ denote the simulated distribution:

$$\mathcal{D}_{\text{Sim}'} := \{(\text{crs}, \pi) \mid (\text{crs}, \pi) \leftarrow \text{Sim}'(1^\lambda, x)\} .$$

Let $\mathcal{D}_{\text{Real}}$ denote the real-world distribution:

$$\mathcal{D}_{\text{Real}} := \left\{ (\text{crs}, \pi) \mid \begin{array}{l} \text{crs} \leftarrow \text{FS.Setup}(1^\lambda, \mathbf{V}) \\ \pi \leftarrow \text{FS.P}(\text{crs}, x) \end{array} \right\} .$$

Observe that the distributions can be expanded as follows:

$$\mathcal{D}_{\text{Real}} = \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \leftarrow \text{FS.Setup}(1^\lambda, \mathbf{V}) \\ (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ a \leftarrow \widehat{\text{Chal}}(x, m_1) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, a) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi := (m_1, m_2) \end{array} \right. \right\},$$

$$\mathcal{D}_{\text{Sim}'} = \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (m_1, \rho, m_2) \leftarrow \text{Sim}(x) \\ \text{crs} \leftarrow \text{Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ \pi \leftarrow \text{FS.P}(\text{crs}, x)(\text{crs}, \pi) \end{array} \right. \right\}.$$

Let $\mathcal{D}_{\text{Hyb}}$ be the following hybrid,

$$\mathcal{D}_{\text{Hyb}} = \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ \rho \leftarrow \{0, 1\}^r \\ (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \leftarrow \text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ a \leftarrow \widehat{\text{Chal}}(x, m_1) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, a) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi := (m_1, m_2) \end{array} \right. \right\}.$$

In what follows, we prove that (1) $\mathcal{D}_{\text{Real}} \approx_{\text{negl}(\lambda)} \mathcal{D}_{\text{Hyb}}$, and (2) $\mathcal{D}_{\text{Hyb}} \equiv_{\text{negl}(\lambda)} \mathcal{D}_{\text{Sim}'}$, which will conclude the proof that $\mathcal{D}_{\text{Real}} \approx_{\text{negl}(\lambda)} \mathcal{D}_{\text{Sim}'}$. To prove that $\mathcal{D}_{\text{Real}} \approx_{\text{negl}(\lambda)} \mathcal{D}_{\text{Hyb}}$ we need to show that,

$$\left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \leftarrow \text{FS.Setup}(1^\lambda, \mathbf{V}) \\ (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ a \leftarrow \widehat{\text{Chal}}(x, m_1) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, a) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi := (m_1, m_2) \end{array} \right. \right\}$$

$$\approx_{\text{negl}(\lambda)} \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ \rho \leftarrow \{0, 1\}^r \\ (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \leftarrow \text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ a \leftarrow \widehat{\text{Chal}}(x, m_1) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, a) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi := (m_1, m_2) \end{array} \right. \right\}.$$

We first prove that the marginal distribution of the crs is indistinguishable between the two distributions. By combining Claim 4.2 and Claim 4.3 from the soundness analysis of the transformation, it holds that $\text{FS.Setup}(1^\lambda, \mathbf{V})$ is indistinguishable from $\text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V})$ when ρ is sampled uniformly. Specifically, the modification that we did to the FS.Setup algorithm to get FS.Setup_2 involves puncturing the PRF key k at location $(x \| m_1)$ and replacing the value $F_k(x \| m_1)$ inside the

obfuscated circuits $(\text{Chal}, \text{Ver})$ with a uniform value. By the security of iO and the puncturable PRF, the resulting obfuscated circuits are indistinguishable.

Next, we prove that the joint distribution (crs, π) is indistinguishable. If the prover was efficient, this would follow directly from the indistinguishability of the crs . However, since the transformation does not restrict the prover's running time in the original protocol, we cannot rely on that. Note that the prover does not receive the full crs . Rather, it only receives the challenge $\mathbf{a}$, which depends on the crs . We show that $\mathbf{a}$ distributes identically in both experiments. Thus, since π is generated according to the same distribution as $\mathbf{a}$, the joint distribution (crs, π) is indistinguishable.

To prove that $\mathbf{a}$ distributes the same, we assume without loss of generality that the PRF is 1-universal. That is, for any y , the distribution $\{\text{PRF}_k(y) \mid k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})\}$ is uniform (the transformation from PRF to 1-universal PRF is the same as described in [CLW18][Construction 3.10]). Recall that $\mathbf{a}$ is generated by computing $\mathbf{V}_1(x, m_1; \rho)$. In $\mathcal{D}_{\text{Real}}$, the randomness is derived as $\rho \leftarrow \text{PRF}_k(x \| m_1)$, whereas in $\mathcal{D}_{\text{Hyb}}$, ρ is sampled uniformly at random. Since the computation of $\mathbf{a}$ depends on a *single* evaluation of ρ , the one-wise independence of the PRF ensures that ρ is uniformly distributed in the real experiment, just as it is in the hybrid experiment. Therefore, ρ is identically distributed in both cases, which implies that $\mathbf{a}$ is also identically distributed as required.

Next, we prove that $\mathcal{D}_{\text{Hyb}} \equiv_{\text{negl}(\lambda)} \mathcal{D}_{\text{Sim}'}$. By the definition of Chal , we can replace the evaluation of the challenge $\mathbf{a}$ in the experiment of $\mathcal{D}_{\text{Hyb}}$ with directly evaluating $\mathbf{a} \leftarrow \mathbf{V}_1(x, m_1; \rho)$, and later compute the crs . Therefore,

$$\begin{aligned} \mathcal{D}_{\text{Hyb}} &= \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ \rho \leftarrow \{0, 1\}^r \\ (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \leftarrow \text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ \mathbf{a} \leftarrow \widehat{\text{Chal}}(x, m_1) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, \mathbf{a}) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi := (m_1, m_2) \end{array} \right. \right\} \\ &\equiv \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ \rho \leftarrow \{0, 1\}^r \\ (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \leftarrow \text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ \mathbf{a} \leftarrow \mathbf{V}_1(x, m_1; \rho) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, \mathbf{a}) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi := (m_1, m_2) \end{array} \right. \right\}. \end{aligned}$$

Since the above experiment samples the crs independently of the proof, we can move it to the end

of the experiment.

$$\begin{aligned}
& \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ \rho \leftarrow \{0, 1\}^r \\ (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \leftarrow \text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ \mathbf{a} \leftarrow \mathbf{V}_1(x, m_1; \rho) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, \mathbf{a}) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi := (m_1, m_2) \end{array} \right. \right\} \\
& \equiv \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ \rho \leftarrow \{0, 1\}^r \\ \mathbf{a} \leftarrow \mathbf{V}_1(x, m_1; \rho) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, \mathbf{a}) \\ \text{crs} \leftarrow \text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ \pi := (m_1, m_2) \end{array} \right. \right\}.
\end{aligned}$$

Observe that (m_1, ρ, m_2) in the above experiment distributes exactly as the real distribution in the interactive protocol. Therefore, by the statistically honest-verifier zero-knowledge property of the interactive protocol,

$$\begin{aligned}
& \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (\text{aux}, m_1) \leftarrow \mathbf{P}_1(x) \\ \rho \leftarrow \{0, 1\}^r \\ \mathbf{a} \leftarrow \mathbf{V}_1(x, m_1; \rho) \\ m_2 \leftarrow \mathbf{P}_2(\text{aux}, x, \mathbf{a}) \\ \text{crs} \leftarrow \text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ \pi := (m_1, m_2) \end{array} \right. \right\} \\
& \equiv_{\text{negl}(\lambda)} \left\{ (\text{crs}, \pi) \left| \begin{array}{l} (m_1, \rho, m_2) \leftarrow \text{Sim}(x) \\ \text{crs} \leftarrow \text{FS.Setup}_2[x, m_1, \rho](1^\lambda, \mathbf{V}) \\ \pi := (m_1, m_2) \end{array} \right. \right\} = \mathcal{D}_{\text{Sim}'} .
\end{aligned}$$

Overall, $\mathcal{D}_{\text{Hyb}} \equiv_{\text{negl}(\lambda)} \mathcal{D}_{\text{Sim}'}$, as required. $\square$

5 On The Necessity of Input-Hiding Obfuscation

In this section, we formalize the notion of a Fiat-Shamir transformation for 3-message private-coin protocols. We prove that the existence of such a transformation, even with sub-exponential security loss, implies the existence of input-hiding obfuscation for evasive circuit families. This result justifies the IHO assumption in our Fiat-Shamir transformation.

Definition 5.1 (Fiat-Shamir for 3-message private-coin protocols). *We say that a transformation from a 3-message protocol private-coin to a non-interactive argument is a Fiat-Shamir transformation with loss $t(\cdot)$ if it consists of a setup algorithm **Setup** that outputs two circuits $(\text{Chal}, \text{Ver})$ such that the following completeness and soundness properties hold for every 3-message protocol $(\mathbf{P}, \mathbf{V} = (\mathbf{V}_1, \mathbf{V}_2))$ with first message size ℓ_1 , verifier randomness size r , and soundness error ϵ , and for every puncturable PRF scheme $\text{PRF} = (\text{PRF.Gen}, \text{PRF.Eval})$.*

Completeness. *For every security parameter $\lambda \in \mathbb{N}$, puncturable PRF key $k \in \text{Img}(\text{PRF.Gen}(1^\lambda, 1^{\ell_1}, 1^r))$, $(\text{Chal}, \text{Ver}) \in \text{Img}(\text{Setup}(1^\lambda, \mathbf{V}, \text{PRF.Eval}, k))$, and for every x, m_1, m_2 , and for $\rho \leftarrow \text{PRF.Eval}(k, m_1)$, the following two conditions hold:*

1. $\text{Chal}(x, m_1) = \mathbf{V}_1(x, m_1; \rho)$.
2. $\text{Ver}(x, m_1, m_2) = \mathbf{V}_2(x, m_1, m_2; \rho)$.

Soundness. *For every security parameter $\lambda \in \mathbb{N}$ and for every PPT adversary Adv ,*

$$\Pr \left[\begin{array}{l} x \notin \mathcal{L} \\ \text{Ver}(x, m_1, m_2) = 1 \end{array} \mid \begin{array}{l} k \leftarrow \text{PRF.Gen}(1^\lambda, 1^{\ell_1}, 1^r) \\ (\text{Chal}, \text{Ver}) \leftarrow \text{Setup}(1^\lambda, \mathbf{V}, \text{PRF.Eval}, k) \\ (x, m_1, m_2) \leftarrow \text{Adv}(\text{Chal}, \text{Ver}) \end{array} \right] \leq t(\lambda) \cdot \epsilon.$$

Theorem 5.2. *Assuming the existence of a secure puncturable PRF scheme, then if there exists a Fiat-Shamir transformation as defined in Definition 5.1 with loss function $t(\lambda) \leq 2^{\lambda^c}$ for some constant c , then there exists an input-hiding obfuscation scheme as defined in Definition 3.9 for every $\mu(\lambda)$ -evasive circuit family (Definition 3.8), where $\mu(\lambda) \leq \text{negl}(\lambda)$.*

Proof. Let $\mathcal{C}$ be an $\mu(\lambda)$ -evasive circuit family for circuits of the form $C : \{0, 1\}^n \rightarrow \{0, 1\}$, where $\mu(\lambda) \leq \text{negl}(\lambda)$. Let c be some constant, and let FS.Setup be a setup algorithm for a Fiat-Shamir transformation with loss $t(\lambda) \leq 2^{\lambda^c}$. Let $\text{PRF} = (\text{PRF.Gen}, \text{PRF.Eval})$ be a puncturable PRF scheme. Let $r(\lambda) := \lambda + \lambda^c$. For every security parameter $\lambda \in \mathbb{N}$, we define the following 3-message protocol $(\mathbf{P}_\lambda, \mathbf{V}_\lambda)$ for the empty language, where $\mathbf{P}_\lambda = (\mathbf{P}_{1,\lambda}, \mathbf{P}_{2,\lambda})$ and $\mathbf{V}_\lambda = (\mathbf{V}_{1,\lambda}, \mathbf{V}_{2,\lambda})$.

1. $\mathbf{P}_{1,\lambda}(x)$: The prover sends $m_1 := x$.
2. $\mathbf{V}_{1,\lambda}(x)$: The verifier samples $\rho \leftarrow \{0, 1\}^{2r(\lambda)}$, and sends the first $r(\lambda)$ bits of ρ as the challenge, $a := (\rho_1, \dots, \rho_{r(\lambda)})$.
3. $\mathbf{P}_{2,\lambda}(x)$: The prover sends $m_2 := \perp$.
4. $\mathbf{V}_{2,\lambda}(x)$: The verifier accepts if and only if $m_2 = \rho$.

Since any adversary can guess the last $r(\lambda)$ bits of ρ with probability at most $2^{-r(\lambda)}$, we get that the above protocol is $\epsilon(\lambda) := 2^{-r(\lambda)}$ secure.

We define a new PRF, PRF' , where the key consists of a key k from PRF and a circuit C that is sampled uniformly from the circuit family $\mathcal{C}$. The PRF evaluation is the same as $\text{PRF.Eval}(k, \cdot)$, except on inputs x where $C(x) = 1$ or $\text{PRF.Eval}(k, x) = \vec{1}$. The construction is defined as follows:

- $\text{PRF}'.\text{Gen}(1^\lambda, 1^{n_1}, 1^{n_2})$:
 1. Sample $k \leftarrow \text{PRF}.\text{Gen}(1^\lambda, 1^{n_1}, 1^{n_2})$.
 2. Sample $C \leftarrow \mathcal{C}$.
 3. Output $k_C := (k \| C)$.
- $\text{PRF}'.\text{Eval}(k_C, x)$:
 1. Parse k_C as $(k \| C)$.
 2. If $C(x) = 1$, output $\vec{1}$.
 3. Else,
 - (a) If $\text{PRF}.\text{Eval}(k, x) = \vec{1}$, then output $\vec{2}$.
 - (b) Else, output $\text{PRF}.\text{Eval}(k, x)$.

Claim 5.3. PRF' is a secure PRF scheme for output of size $\geq \lambda$. Moreover, if PRF is puncturable, then so is PRF' .

We will rely on this claim in the proof, and present the formal proof of the claim later. Based on the above protocols $(\mathbf{P}_\lambda, \mathbf{V}_\lambda)$ and the construction of PRF' , we introduce our IHO construction for the circuit family $\mathcal{C}$.

- $\text{IHO}(C; r)$.
 1. Generate PRF key $k \leftarrow \text{PRF}.\text{Gen}(1^\lambda, 1^n, 1^{2r(\lambda)})$.
 2. Set $k_C := (k \| C)$ as a PRF key.
 3. Apply the Fiat-Shamir transformation: $(\text{Ver}, \text{Chal}) \leftarrow \text{FS}.\text{Setup}(1^\lambda, \mathbf{V}_\lambda, \text{PRF}'.\text{Eval}, k_C)$.
 4. Output $\hat{C} := \text{Ver}(x, \cdot, \vec{1})$, where x is an arbitrary instance.

Preserving functionality. We prove that the construction above preserves functionality for every $C \in \mathcal{C}$. Fix $C \in \mathcal{C}$. Recall that in our 3-message protocol,

$$\mathbf{V}_{2,\lambda}(x, m_1, m_2; \rho) = 1 \iff m_2 = \rho .$$

Therefore, by the completeness of the Fiat-Shamir transformation (Definition 5.1), since $\rho = \text{PRF}'.\text{Eval}(k_C, m_1)$,

$$\text{Ver}(x, m_1, \vec{1}) = 1 \iff \text{PRF}'.\text{Eval}(k_C, m_1) = \vec{1} .$$

By the PRF' construction, $\text{PRF}'.\text{Eval}(k_C, m_1) = \vec{1}$ if and only if $C(m_1) = 1$. Therefore, we get that,

$$\text{Ver}(x, m_1, \vec{1}) = 1 \iff C(m_1) = 1 ,$$

as required.

Soundness. Let Adv be a PPT adversary to the IHO scheme, and let $\nu(\lambda)$ be its success probability.

$$\nu(\lambda) := \Pr \left[C(a) = 1 \mid \begin{array}{l} C \leftarrow \mathcal{C} \\ \hat{C} \leftarrow \text{IHO}(C) \\ a \leftarrow \text{Adv}(\hat{C}) \end{array} \right] .$$

Let $\tilde{\mathbf{P}}$ be the following adversary to the non-interactive protocol in the above experiment.

1. On input $\text{crs} = (\text{Chal}, \text{Ver})$, set $\text{Ver}_1 := \text{Ver}(x, \cdot, \vec{1})$.
(For the same arbitrary x used in the IHO construction.)

2. Emulate $\text{Adv}(\text{Ver}_1)$ to get a .
3. Output x , $m_1 := a$, and $m_2 := \vec{1}$.

By the soundness of the Fiat-Shamir transformation, and since, by Claim 5.3, PRF' is a secure PRF for output size $2r(\lambda) = 2(\lambda + \lambda^c) \geq \lambda$, we get that for any PPT adversary $\tilde{\mathbf{P}}$,

$$\begin{aligned} & \Pr \left[\text{Ver}(x, m_1, m_2) = 1 \mid \begin{array}{l} k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda, 1^n, 1^{2r(\lambda)}) \\ (\text{Chal}, \text{Ver}) \leftarrow \text{Setup}(\mathbf{V}_\lambda, \text{PRF}'.\text{Eval}, k_C) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{Chal}, \text{Ver}) \end{array} \right] \\ & \leq t(\lambda) \cdot \epsilon(\lambda) \leq 2^{\lambda^c} \cdot 2^{-r(\lambda)} = 2^{\lambda^c} \cdot 2^{-(\lambda + \lambda^c)} = 2^{-\lambda} = \text{negl}(\lambda) . \end{aligned} \quad (10)$$

By the construction of $\tilde{\mathbf{P}}$, and by our IHO construction,

$$\begin{aligned} & \Pr \left[\text{Ver}(x, m_1, m_2) = 1 \mid \begin{array}{l} k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda, 1^n, 1^{2r(\lambda)}) \\ (\text{Chal}, \text{Ver}) \leftarrow \text{Setup}(\mathbf{V}_\lambda, \text{PRF}'.\text{Eval}, k_C) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{Chal}, \text{Ver}) \end{array} \right] \\ & = \Pr \left[\text{Ver}_1(a) = 1 \mid \begin{array}{l} k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda, 1^n, 1^{2r(\lambda)}) \\ (\text{Chal}, \text{Ver}) \leftarrow \text{Setup}(\mathbf{V}_\lambda, \text{PRF}'.\text{Eval}, k_C) \\ \text{Ver}_1 := \text{Ver}(x, \cdot, \vec{1}) \\ a \leftarrow \text{Adv}(\text{Ver}_1) \end{array} \right] \\ & = \Pr \left[C(a) = 1 \mid \begin{array}{l} C \leftarrow \mathcal{C} \\ \hat{C} \leftarrow \text{IHO}(C) \\ a \leftarrow \text{Adv}(\hat{C}) \end{array} \right] = \nu(\lambda) . \end{aligned}$$

By the above and by Equation 10, we get that, $\nu(\lambda) = \text{negl}(\lambda)$, as required.

Proof of Claim 5.3. We prove two general claims showing that, with all but negligible probability: (1) an adversary with bounded oracle queries to a circuit $C \leftarrow \mathcal{C}$ cannot find an input x such that $C(x) \neq 0$ (Claim 5.4), and (2) an efficient adversary with oracle access to a PRF function cannot find an input x such that $\text{PRF}(x) = \vec{1}$ (Claim 5.5). We then use these claims to prove the security of PRF' for any output size $\geq \lambda$.

Claim 5.4. *For every evasive family $\mathcal{C}$ and for every adversary Adv that makes $N \leq \text{poly}(\lambda)$ queries,*

$$\Pr \left[\exists i \in [N] \text{ s.t. } C(\mathbf{q}_i) = 1 \mid \begin{array}{l} C \leftarrow \mathcal{C} \\ (\mathbf{q}_1, \dots, \mathbf{q}_N) \leftarrow \text{Adv}^C \end{array} \right] \leq N \cdot \text{negl}(\lambda) ,$$

where $(\mathbf{q}_1, \dots, \mathbf{q}_N)$ are the queries the adversary makes to the oracle.

Proof. Fix some PPT adversary Adv . Let $N \leq \text{poly}(\lambda)$ be the number of queries that the adversary makes. Let C_0 be the zero circuit. It holds that,

$$\begin{aligned} & \Pr \left[\exists i \text{ s.t. } C(\mathbf{q}_i) = 1 \mid \begin{array}{l} C \leftarrow \mathcal{C} \\ (\mathbf{q}_1, \dots, \mathbf{q}_N) \leftarrow \text{Adv}^C \end{array} \right] \\ & = \sum_{i \in [N]} \Pr \left[\begin{array}{l} C(\mathbf{q}_i) = 1 \\ \wedge \forall j \in [i-1], C(\mathbf{q}_j) = 0 \end{array} \mid \begin{array}{l} C \leftarrow \mathcal{C} \\ (\mathbf{q}_1, \dots, \mathbf{q}_N) \leftarrow \text{Adv}^C \end{array} \right] . \end{aligned}$$

Consider the i -th term above. Conditioned on the event that all previous queries $j < i$ evaluated to 0, the adversary's view up to the i -th query is identical to an interaction with the zero circuit C_0 . Therefore, we can replace the oracle C with C_0 for the generation of the queries:

$$\begin{aligned}
& \sum_{i \in [N]} \Pr \left[C(q_i) = 1 \mid \bigwedge_{j \in [i-1]} C(q_j) = 0 \mid \begin{array}{l} C \leftarrow \mathcal{C} \\ (q_1, \dots, q_N) \leftarrow \text{Adv}^C \end{array} \right] \\
&= \sum_{i \in [N]} \Pr \left[C(q_i) = 1 \mid \bigwedge_{j \in [i-1]} C(q_j) = 0 \mid \begin{array}{l} C \leftarrow \mathcal{C} \\ (q_1, \dots, q_N) \leftarrow \text{Adv}^{C_0} \end{array} \right] \\
&\leq \sum_{i \in [N]} \Pr \left[C(q_i) = 1 \mid \begin{array}{l} C \leftarrow \mathcal{C} \\ (q_1, \dots, q_N) \leftarrow \text{Adv}^{C_0} \end{array} \right] \\
&= \sum_{i \in [N]} \Pr \left[C(q_i) = 1 \mid \begin{array}{l} (q_1, \dots, q_N) \leftarrow \text{Adv}^{C_0} \\ C \leftarrow \mathcal{C} \end{array} \right] .
\end{aligned}$$

In the last equation, we reordered the experiment. This is valid since the query generation (under C_0) and the sampling of C are now independent. Therefore, by the evasiveness of the circuit family $\mathcal{C}$, for every $i \in [N]$,

$$\Pr \left[C(q_i) = 1 \mid \begin{array}{l} (q_1, \dots, q_N) \leftarrow \text{Adv}^{C_0} \\ C \leftarrow \mathcal{C} \end{array} \right] \leq \text{negl}(\lambda) .$$

Overall, we get that,

$$\begin{aligned}
& \sum_{i \in [N]} \Pr \left[C(q_i) = 1 \mid \begin{array}{l} (q_1, \dots, q_N) \leftarrow \text{Adv}^{C_0} \\ C \leftarrow \mathcal{C} \end{array} \right] \\
&\leq \sum_{i \in [N]} \text{negl}(\lambda) \\
&= N \cdot \text{negl}(\lambda) .
\end{aligned}$$

□

Claim 5.5. *For every ϵ -secure PRF, it holds that for every PPT adversary Adv that makes N queries $(q_1, \dots, q_N)$ and for every $n_1, n_2 \in \mathbb{N}$,*

$$\Pr \left[\exists i \in [N] \text{ s.t. } \text{PRF}(k, q_i) = \vec{1} \mid \begin{array}{l} k \leftarrow \text{PRF.Gen}(1^\lambda, 1^{n_1}, 1^{n_2}) \\ (q_1, \dots, q_N) \leftarrow \text{Adv}^{\text{PRF}(k, \cdot)} \end{array} \right] \leq 2 \cdot N \cdot \epsilon + \frac{N}{2^{n_2}} .$$

Proof. We begin by showing that for every fixed input $q \in \{0, 1\}^{n_1}$, the probability over the choice of the PRF key k that $\text{PRF}(k, q) = \vec{1}$ is small. We then apply a union bound to obtain the claim. Let D_q be the following adversary to the PRF scheme,

1. Query the oracle with query q to obtain the value y .
2. If $y = \vec{1}$, output 1.
3. Else, output $b \leftarrow \{0, 1\}$.

let U_{n_1, n_2} be a uniform distribution over functions $f : \{0, 1\}^{n_1} \rightarrow \{0, 1\}^{n_2}$. The success probability of the adversary in breaking the PRF security is as follows,

$$\begin{aligned}
\epsilon &\geq \Pr_{k \leftarrow \text{PRF.Gen}(1^\lambda, 1^{n_1}, 1^{n_2})} [\mathcal{D}_q^{\text{PRF.Eval}(k, \cdot)} = 1] - \Pr_{f \leftarrow U_{n_1, n_2}} [\mathcal{D}_q^f = 1] \\
&= \left(\Pr_k [\text{PRF}(k, q) = \vec{1}] + \Pr_k [\text{PRF}(k, q) \neq \vec{1} \wedge b = 1 \mid b \leftarrow \{0, 1\}] \right) \\
&\quad - \left(\Pr_f [f(q) = \vec{1}] + \Pr_f [f(q) \neq \vec{1} \wedge b = 1 \mid b \leftarrow \{0, 1\}] \right) \\
&= \left(\Pr_k [\text{PRF}(k, q) = \vec{1}] + \Pr_{b \leftarrow \{0, 1\}} [b = 1] \cdot \Pr_k [\text{PRF}(k, q) \neq \vec{1}] \right) \\
&\quad - \left(\Pr_f [f(q) = \vec{1}] + \Pr_{b \leftarrow \{0, 1\}} [b = 1] \cdot \Pr_f [f(q) \neq \vec{1}] \right) \\
&= \left(\Pr_k [\text{PRF}(k, q) = \vec{1}] + \frac{1}{2} \cdot \Pr_k [\text{PRF}(k, q) \neq \vec{1}] \right) \\
&\quad - \left(\Pr_f [f(q) = \vec{1}] + \frac{1}{2} \cdot \Pr_f [f(q) \neq \vec{1}] \right) \\
&= \Pr_k [\text{PRF}(k, q) = \vec{1}] + \frac{1}{2} \cdot \left(1 - \Pr_k [\text{PRF}(k, q) = \vec{1}] \right) - \left(\frac{1}{2^{n_2}} + \frac{1}{2} \cdot \left(1 - \frac{1}{2^{n_2}} \right) \right) \\
&= \frac{1}{2} \cdot \Pr_k [\text{PRF}(k, q) = \vec{1}] - \frac{1}{2} \cdot \frac{1}{2^{n_2}}.
\end{aligned}$$

Where the first inequality is by the PRF security, the first equality is by the definition of the adversary (if the adversary receives 1 from the oracle, it outputs 1. Otherwise, it outputs 1 when $b = 1$), the second equality is by the law of total probability, and the forth equality is since f is a random function that outputs n_2 random bits. By the most left and most right sides of the equations above, for every $q \in \{0, 1\}^n$,

$$\Pr_k [\text{PRF}(k, q) = \vec{1}] \leq 2\epsilon + \frac{1}{2^{n_2}}.$$

By union bound,

$$\begin{aligned}
&\Pr \left[\exists i \in [N] \text{ s.t. } \text{PRF}(k, q_i) = \vec{1} \mid \begin{array}{l} k \leftarrow \text{PRF.Gen}(1^n, 1^{n_2}, 1^\lambda) \\ (q_1, \dots, q_N) \leftarrow \text{Adv}^{\text{PRF}(k, \cdot)} \end{array} \right] \\
&\leq \sum_{i \in [N]} \Pr \left[\text{PRF}(k, q_i) = \vec{1} \mid \begin{array}{l} k \leftarrow \text{PRF.Gen}(1^n, 1^{n_2}, 1^\lambda) \\ (q_1, \dots, q_N) \leftarrow \text{Adv}^{\text{PRF}(k, \cdot)} \end{array} \right] \\
&\leq \sum_{i \in [N]} \left(2\epsilon + \frac{1}{2^{n_2}} \right) \\
&= 2 \cdot N \cdot \epsilon + \frac{N}{2^{n_2}},
\end{aligned}$$

as required. $\square$

We proceed to the security proof of PRF' . Let U_{n_1, n_2} denote the uniform distribution over all functions $f : \{0, 1\}^{n_1} \rightarrow \{0, 1\}^{n_2}$. In what follows, we use $\text{Gen}(1^\lambda)$ as shorthand for $\text{Gen}(1^\lambda, 1^{n_1}, 1^{n_2})$.

We need to prove that for every PPT distinguisher D .

$$\Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1] - \Pr_{f \leftarrow U_{n_1, n_2}} [D^f = 1] \leq \text{negl}(\lambda) .$$

Let E_C be the event where D queries the oracle at some location q such that $C(q) = 1$. Let E_1 be the event where D queries the oracle at some location q such that $\text{PRF}.\text{Eval}(k, q) = \vec{1}$. Observe that the function $\text{PRF}'.\text{Eval}(k_C, \cdot)$ differs from $\text{PRF}.\text{Eval}(k, \cdot)$ only for queries q where either $C(q) = 1$ or $\text{PRF}.\text{Eval}(k, q) = \vec{1}$. Therefore, conditioned on $\overline{E_C} \wedge \overline{E_1}$, the two oracles (PRF and PRF') agrees on all queries. By the law of total probability,

$$\begin{aligned} & \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1] \\ &= \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [E_C] \cdot \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [D^{\text{PRF}'.\text{Eval}(k, \cdot)} = 1 \mid E_C] \\ &+ \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [E_1 \wedge \overline{E_C}] \cdot \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1 \mid E_1 \wedge \overline{E_C}] \\ &+ \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [\overline{E_1} \wedge \overline{E_C}] \cdot \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1 \mid \overline{E_1} \wedge \overline{E_C}] . \end{aligned}$$

Observe that the PRF' oracle can be emulated given access to a PRF oracle (by locally sampling $C \leftarrow \mathcal{C}$ and following the $\text{PRF}'.\text{Eval}$ algorithm). Therefore, by Claim 5.5, we get that,

$$\begin{aligned} \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [E_1] &\leq 2 \cdot N \cdot \epsilon + \frac{N}{2^{n_2}} \\ &\leq \text{negl}(\lambda) + \frac{\text{poly}(\lambda)}{2^{n_2}} , \end{aligned}$$

where the second inequality is by the PRF security, and since the distinguisher is PPT and therefore makes $N \leq \text{poly}(\lambda)$ queries. By Claim 5.4, since $\text{PRF}'.\text{Gen}(1^\lambda)$ samples $C \leftarrow \mathcal{C}$,

$$\begin{aligned} \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [E_C] &\leq N \cdot \text{negl}(\lambda) \\ &\leq \text{negl}(\lambda) , \end{aligned}$$

where the last inequality is since the adversary makes at most $\text{poly}(\lambda)$ queries. By combining the above three equations,

$$\begin{aligned} & \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1] \\ &\leq \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [\overline{E_1} \wedge \overline{E_C}] \cdot \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} [D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1 \mid \overline{E_1} \wedge \overline{E_C}] \\ &+ \text{negl}(\lambda) + \frac{\text{poly}(\lambda)}{2^{n_2}} \\ &= \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} \left[\frac{D^{\text{PRF}'.\text{Eval}(k_C, \cdot)}}{\overline{E_1} \wedge \overline{E_C}} = 1 \right] + \text{negl}(\lambda) + \frac{\text{poly}(\lambda)}{2^{n_2}} . \end{aligned}$$

where the last inequality is by the law of total probability. Recall that if the event $\overline{E_1} \wedge \overline{E_C}$ occurs, then the view of the distinguisher is the same for oracle $\text{PRF}'.\text{Eval}(k_C, \cdot)$ and $\text{PRF}.\text{Eval}(k, \cdot)$.

Therefore,

$$\begin{aligned}
\Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} \left[\frac{D^{\text{PRF}'.\text{Eval}(k_C, \cdot)}}{\overline{E}_1 \wedge \overline{E}_C} = 1 \right] &= \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} \left[\frac{D^{\text{PRF}.\text{Eval}(k, \cdot)}}{\overline{E}_1 \wedge \overline{E}_C} = 1 \right] \\
&\leq \Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} \left[D^{\text{PRF}.\text{Eval}(k, \cdot)} = 1 \right] \\
&\leq \Pr_{k \leftarrow \text{PRF}.\text{Gen}(1^\lambda)} \left[D^{\text{PRF}.\text{Eval}(k, \cdot)} = 1 \right].
\end{aligned}$$

By the above two equations,

$$\begin{aligned}
&\Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} \left[D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1 \right] \\
&\leq \Pr_{k \leftarrow \text{PRF}.\text{Gen}(1^\lambda)} \left[D^{\text{PRF}.\text{Eval}(k, \cdot)} = 1 \right] + \text{negl}(\lambda) + \frac{\text{poly}(\lambda)}{2^{n_2}}.
\end{aligned}$$

By the above and by the security of the underlying PRF, we get that,

$$\begin{aligned}
&\Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} \left[D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1 \right] - \Pr_{f \leftarrow U_{n_1, n_2}} \left[D^f = 1 \right] \\
&\leq \Pr_{k \leftarrow \text{PRF}.\text{Gen}(1^\lambda)} \left[D^{\text{PRF}.\text{Eval}(k, \cdot)} = 1 \right] - \Pr_{f \leftarrow U_{n_1, n_2}} \left[D^f = 1 \right] + \text{negl}(\lambda) + \frac{\text{poly}(\lambda)}{2^{n_2}} \\
&\leq \text{negl}(\lambda) + \frac{\text{poly}(\lambda)}{2^{n_2}}.
\end{aligned}$$

Therefore, for $n_2 \geq \lambda$,

$$\Pr_{k_C \leftarrow \text{PRF}'.\text{Gen}(1^\lambda)} \left[D^{\text{PRF}'.\text{Eval}(k_C, \cdot)} = 1 \right] - \Pr_{f \leftarrow U_{n_1, n_2}} \left[D^f = 1 \right] \leq \text{negl}(\lambda),$$

as required.

Puncturability. Suppose that PRF is puncturable. Define

$$\text{PRF}'.\text{Punc}((k \| C), x) := (\text{PRF}.\text{Punc}(k, x) \| C).$$

To evaluate at any $x' \neq x$, run the same evaluation algorithm as above, using the punctured underlying key in place of k . Puncturing correctness follows immediately from that of PRF.

For puncturing security, fix any input $x \in \{0, 1\}^{n_1}$, as in the definition of puncturable PRFs, and write $\ell := n_2 \geq \lambda$. Let k and C be sampled independently as in $\text{PRF}'.\text{Gen}$, and let $k^* := \text{PRF}.\text{Punc}(k, x)$. For $y \in \{0, 1\}^\ell$, define

$$h_{C, x}(y) := \begin{cases} \vec{1} & \text{if } C(x) = 1, \\ \vec{2} & \text{if } C(x) = 0 \text{ and } y = \vec{1}, \\ y & \text{otherwise.} \end{cases}$$

Thus $\text{PRF}'.\text{Eval}((k \| C), x) = h_{C, x}(\text{PRF}.\text{Eval}(k, x))$. By puncturing security of PRF, for an independent uniform $U \leftarrow \{0, 1\}^\ell$, the distributions

$$((k^* \| C), x, h_{C, x}(\text{PRF}.\text{Eval}(k, x))) \quad \text{and} \quad ((k^* \| C), x, h_{C, x}(U))$$

are computationally indistinguishable: a reduction samples C independently and applies $h_{C,x}$ to its challenge. This holds even though the punctured key reveals C . Finally, couple the latter distribution with $((\mathbf{k}^* \| C), x, U)$ using the same $\mathbf{k}^*$, C , and U . The coupled tuples agree unless $C(x) = 1$ or $U = \vec{1}$, so their statistical distance is at most

$$\Pr_{C \leftarrow \mathcal{C}}[C(x) = 1] + \Pr[U = \vec{1}] \leq \mu(\lambda) + 2^{-\ell} \leq \text{negl}(\lambda).$$

Together, these two steps prove puncturing security of PRF' . □

□

6 Beyond 3-Message Protocols

In this section, we show that the Fiat-Shamir transformation does not extend to private-coin protocols that have more than 3-messages. To establish this, we provide a generalized definition of the Fiat-Shamir transformation for private-coin protocols that captures an arbitrary number of messages. We then prove that, assuming the existence of a collision-resistant hash function, there is no Fiat-Shamir transformation for protocols with 5-messages or more.

Definition 6.1 (Fiat-Shamir for k -message private-coin protocols). *We say that a transformation from an k -message private-coin protocol to a non-interactive argument is a Fiat-Shamir transformation if it consists of a setup algorithm Setup that outputs $k - 1$ challenge-generation circuits $\text{Chal}_1, \dots, \text{Chal}_{k-1}$ and a verification circuit Ver . For every k -message protocol $(\mathbf{P}, \mathbf{V})$ with soundness error ϵ , the transformation must satisfy the following completeness and soundness properties:*

Completeness. *For every security parameter $\lambda \in \mathbb{N}$, $(\text{Chal}_1, \dots, \text{Chal}_{k-1}, \text{Ver}) \in \text{Img}(\text{Setup}(1^\lambda, (\mathbf{P}, \mathbf{V})))$, instance x , and a sequence of prover messages $m_1, \dots, m_k$, the following holds. If we define the challenges as $\mathbf{a}_i := \text{Chal}_i(x, m_1, \dots, m_i)$ for $i \in [k - 1]$, and the final decision as $b := \text{Ver}(x, m_1, \dots, m_k)$, then there exists verifier randomness ρ such that the generated transcript $\text{tr} := (m_1, \mathbf{a}_1, \dots, \mathbf{a}_{k-1}, m_k, b)$ is consistent with the interactive execution where the verifier uses randomness ρ : $\text{tr} = \langle \mathbf{P}, \mathbf{V}(\rho) \rangle(x)$.*

Soundness. *For every security parameter $\lambda \in \mathbb{N}$ and every PPT adversary Adv ,*

$$\Pr \left[\begin{array}{l} x \notin \mathcal{L} \\ \text{Ver}(x, m_1, \dots, m_k) = 1 \end{array} \mid \begin{array}{l} k \leftarrow \text{PRF.Gen}(1^\lambda, 1^{\ell_1}, 1^r) \\ (\text{Chal}_1, \dots, \text{Chal}_{k-1}, \text{Ver}) \leftarrow \text{Setup}(1^\lambda, (\mathbf{P}, \mathbf{V})) \\ (x, m_1, \dots, m_k) \leftarrow \text{Adv}(\text{Chal}_1, \dots, \text{Chal}_{k-1}, \text{Ver}) \end{array} \right] \leq \epsilon \cdot \text{poly}(\lambda) .$$

Theorem 6.2. *Assuming the existence of a collision-resistant hash function, then there is no Fiat-Shamir transformation satisfying Definition 6.1 exists for protocols with 5 or more messages.*

Proof. Assume towards contradiction that there exists a Fiat-Shamir transformation for 5 message protocols FS.Setup as defined in Definition 6.1, and let $(\text{H.Gen}, \text{Hash})$ be a collision-resistant hash scheme that for security parameter $\lambda \in \mathbb{N}$, gets 2λ bits as input and outputs λ bits.

For every security parameter $\lambda \in \mathbb{N}$, and for every hash key $\text{hk} \in \text{Img}(\text{H.Gen}(1^\lambda))$, we define the following 5-message protocol $(\mathbf{P}_{\text{hk}}, \mathbf{V}_{\text{hk}})$ for the empty language.

1. $\mathbf{P}_{\text{hk}}(x)$: The prover sends $m_1 := \perp$.
2. $\mathbf{V}_{\text{hk}}(x)$: The verifier samples $\rho \leftarrow \{0, 1\}^{2\lambda}$, and sends $y := \text{Hash}(\text{hk}, \rho)$ to the prover.
3. $\mathbf{P}_{\text{hk}}(x)$: The prover sends $m_2 := \perp$.
4. $\mathbf{V}_{\text{hk}}(x)$: The verifier sends ρ to the prover.
5. $\mathbf{P}_{\text{hk}}(x)$: The prover sends $m_3 := \perp$.
6. $\mathbf{V}_{\text{hk}}(x)$: The verifier accepts if and only if $m_2 = \rho$.

We start by proving that the above protocol is secure. Let $\tilde{\mathbf{P}}$ be some adversary to the above protocol. Fix any instance x , and any hash key hk . Let $S_y := \{\rho \in \{0, 1\}^{2\lambda} \mid y := \text{Hash}(\text{hk}, \rho)\}$ be the set of preimages for a given challenge y . Because the verifier applies the hash function to a uniformly random input, the probability that $\tilde{\mathbf{P}}$ correctly guesses the exact ρ chosen by the verifier

is at most $1/|S_y|$. By the law of total probability, the success probability of $\tilde{\mathbf{P}}$ is bounded as follows.

$$\begin{aligned}
\Pr[\tilde{\mathbf{P}} \text{ Succeeds}] &\leq \sum_{\mathbf{hk}} \Pr_{\mathbf{hk}' \leftarrow \text{H.Gen}(1^\lambda)}[\mathbf{hk}' = \mathbf{hk}] \cdot \sum_{y \in \{0,1\}^\lambda} \Pr_{\rho \leftarrow \{0,1\}^{2\lambda}}[y = \text{Hash}(\mathbf{hk}, \rho)] \cdot \frac{1}{|S_y|} \\
&= \sum_{\mathbf{hk}} \Pr_{\mathbf{hk}' \leftarrow \text{H.Gen}(1^\lambda)}[\mathbf{hk}' = \mathbf{hk}] \cdot \sum_{y \in \{0,1\}^\lambda} \frac{|S_y|}{2^{2\lambda}} \cdot \frac{1}{|S_y|} \\
&= \sum_{\mathbf{hk}} \Pr_{\mathbf{hk}' \leftarrow \text{H.Gen}(1^\lambda)}[\mathbf{hk}' = \mathbf{hk}] \cdot \frac{1}{2^\lambda} = \frac{1}{2^\lambda} .
\end{aligned}$$

Therefore, the protocol is $\epsilon = 2^{-\lambda}$ secure.

Fix an arbitrary instance x and arbitrary prover messages $m_1 = \perp$, $m_2 = \perp$, and $m_3 = \perp$, and consider the following PPT adversary $\text{Adv}(\mathbf{hk})$ to the collision-resistant hash.

1. Sample $(\text{Chal}_1, \text{Chal}_2, \text{Ver}) \leftarrow \text{FS.Setup}(1^\lambda, (\mathbf{P}_{\mathbf{hk}}, \mathbf{V}_{\mathbf{hk}}))$.
2. Compute
 - (a) $\rho := \text{Chal}_2(x, m_1, m_2)$.
 - (b) $\rho' := \text{Chal}_2(x, m_1, \rho)$.
3. Output (ρ, ρ') .

Let $y := \text{Chal}_1(x, m_1)$. By the perfect completeness of the transformation, for $\rho = \text{Chal}_2(x, m_1, m_2)$, it holds that $y = \text{Hash}(\mathbf{hk}, \rho)$. Similarly, for $\rho' = \text{Chal}_2(x, m_1, \rho)$, it holds that $y = \text{Hash}(\mathbf{hk}, \rho')$. Therefore, the adversary always outputs $\text{Hash}(\mathbf{hk}, \rho) = \text{Hash}(\mathbf{hk}, \rho')$, and we get that,

$$\Pr \left[\text{Hash}(\mathbf{hk}, \rho) = \text{Hash}(\mathbf{hk}, \rho') \mid \begin{array}{l} \mathbf{hk} \leftarrow \text{Hash.Gen}(1^\lambda) \\ (\rho, \rho') \leftarrow \text{Adv}(\mathbf{hk}) \end{array} \right] = 1 .$$

In the next step, we prove that for every hash key $\mathbf{hk}$, with overwhelming probability, the adversary outputs $\rho \neq \rho'$. Fix $\mathbf{hk} \in \text{Img}(\text{Hash.Gen}(1^\lambda))$. By the soundness of the Fiat-Shamir transformation, an adversary that computes $\rho := \text{Chal}_2(m_1, m_2)$, and then sends (m_1, ρ, m_3) as the proof, should not convince the verifier with more than negligible probability. Therefore,

$$\Pr \left[\text{Ver}(x, m_1, \rho, m_3) = 1 \mid \begin{array}{l} (\text{Chal}_1, \text{Chal}_2, \text{Ver}) \leftarrow \text{FS.Setup}(1^\lambda, (\mathbf{P}_{\mathbf{hk}}, \mathbf{V}_{\mathbf{hk}})) \\ \rho := \text{Chal}_2(x, m_1, m_2) \end{array} \right] \leq \text{negl}(\lambda) .$$

By the completeness of the transformation, for $\rho' := \text{Chal}_2(x, m_1, \rho)$, it holds that $\text{Ver}(x, m_1, \rho, m_3) = 1$ if and only if $\rho = \rho'$. Therefore,

$$\Pr \left[\rho = \rho' \mid \begin{array}{l} (\text{Chal}_1, \text{Chal}_2, \text{Ver}) \leftarrow \text{FS.Setup}(1^\lambda, (\mathbf{P}_{\mathbf{hk}}, \mathbf{V}_{\mathbf{hk}})) \\ \rho := \text{Chal}_2(x, m_1, m_2) \\ \rho' := \text{Chal}_2(x, m_1, \rho) \end{array} \right] \leq \text{negl}(\lambda) .$$

Since the above holds for any $\mathbf{hk}$, and by the definition of Adv ,

$$\Pr \left[\rho = \rho' \mid \begin{array}{l} \mathbf{hk} \leftarrow \text{Hash.Gen}(1^\lambda) \\ (\rho, \rho') \leftarrow \text{Adv}(\mathbf{hk}) \end{array} \right] \leq \text{negl}(\lambda) .$$

Overall, we get that,

$$\Pr \left[\text{Hash}(\mathbf{hk}, \rho) = \text{Hash}(\mathbf{hk}, \rho') \mid \begin{array}{l} \mathbf{hk} \leftarrow \text{Hash.Gen}(1^\lambda) \\ (\rho, \rho') \leftarrow \text{Adv}(\mathbf{hk}) \end{array} \right] \geq 1 - \text{negl}(\lambda) .$$

as required. $\square$

7 On The Existence of 3-Message Zero-Knowledge Proofs for NP

In this section, we show that any 3-message interactive proof that remains secure under the Fiat–Shamir transformation (as described in Section 4.1) cannot satisfy zero-knowledge with p -bounded auxiliary input for some polynomial p .

This definition of zero-knowledge is a weaker notion than the one proposed in [FGJ18]. Specifically, in our definition, all involved algorithms - the malicious verifier, the simulator, and the distinguisher - receive the same auxiliary input $\mathbf{aux}$. In contrast, the definition in [FGJ18] provides the distinguisher with a larger auxiliary input $\mathbf{aux}'$ that may contain $\mathbf{aux}$ alongside additional information.² Notably, our proof implies an even stronger impossibility result, as it rules out zero-knowledge even in settings where the auxiliary input $\mathbf{aux}$ is restricted to being efficiently sampled.

Definition 7.1 (Zero-knowledge with bounded auxiliary input). *An interactive proof system $(\mathbf{P}, \mathbf{V})$ for language $\mathcal{L}$ satisfies zero-knowledge with p -bounded auxiliary input if for every (possibly malicious) PPT algorithms $\mathbf{V}^*$, and PPT distinguisher $\mathbf{D}$, there exists a PPT simulator $\text{Sim}(\cdot)$, such that for every instance $x \in \mathcal{L}$, and for every auxiliary input $\mathbf{aux} \in \{0, 1\}^{p(|x|)}$,*

$$|\Pr[\mathbf{D}(\langle \mathbf{P}(x), \mathbf{V}^*(x, \mathbf{aux}) \rangle, \mathbf{aux}) = 1] - \Pr[\mathbf{D}(\text{Sim}(x, \mathbf{aux}), \mathbf{aux}) = 1]| \leq \text{negl}(\lambda) .$$

Theorem 7.2. *Let Π be a 3-message interactive proof system for a language $\mathcal{L} \notin \text{BPP}$. If the protocol Π' obtained by the transformation described in Section 4.1 on Π has soundness error $\epsilon' \leq \text{negl}(n)$ against polynomial-size adversaries, then Π does not satisfy zero-knowledge p -bounded auxiliary for some polynomial p .*

Proof. In the following proof, we couple the security parameter λ with the instance size n . Let $\Pi = (\mathbf{P}, \mathbf{V})$ be a 3-message proof system for some language $\mathcal{L}$. Let $\Pi' = (\text{FS.Setup}, \text{FS.P}, \text{FS.V})$ denote the protocol obtained after applying the transformation, and let $p(n) = \text{poly}(n)$ be the size of the crs generated by $\text{FS.Setup}(1^n, \mathbf{V})$ plus the size of the instance. Assuming that Π satisfies zero-knowledge with p -bounded auxiliary input and that Π' has soundness error $\epsilon' \leq \text{negl}(n)$, we show that $\mathcal{L} \in \text{BPP}$. Let $\mathbf{V}^* = (\mathbf{V}_1^*, \mathbf{V}_2^*)$ be the following verifier:

- $\mathbf{V}_1^*(x, m_1, \mathbf{aux})$: on instance x , first prover message m_1 , and auxiliary input $\mathbf{aux}$, the verifier parses $\mathbf{aux}$ as (x, crs) where $\text{crs} = (\widehat{\text{Chal}}, \widehat{\text{Ver}})$, and outputs $\mathbf{a} \leftarrow \widehat{\text{Chal}}(x, m_1)$.
- $\mathbf{V}_2^*(x, m_1, m_2, \mathbf{aux})$: on instance x , first prover message m_1 , and auxiliary input $\mathbf{aux}$, the verifier parses $\mathbf{aux}$ as (x, crs) where $\text{crs} = (\widehat{\text{Chal}}, \widehat{\text{Ver}})$, and outputs (m_1, m_2) .

Note that for $\mathbf{aux} = \text{crs}$, the interaction $\langle \mathbf{P}(x), \mathbf{V}^*(x, \mathbf{aux}) \rangle$ produces the same prover messages (m_1, m_2) as generated by the non interactive prover $\text{FS.P}(\text{crs}, x)$. Let $\mathbf{D}$ be the distinguisher that given $(m_1, m_2), \mathbf{aux}$, interprets $\mathbf{aux}$ as (crs, x) and outputs 1 if and only if $\text{FS.V}(\text{crs}, x, (m_1, m_2))$ accepts. Assume that Π satisfies zero-knowledge with p -bounded auxiliary, and let Sim be its simulator relative to the verifier $\mathbf{V}^*$ and the distinguisher $\mathbf{D}$. Let $\mathbf{A}$ be the following PPT algorithm that gets x as an input.

1. Sample $\text{crs} \leftarrow \text{FS.Setup}(1^n, \mathbf{V})$.

²Goldreich and Oren [GO94] showed that these two notions are equivalent when the simulator's runtime is polynomial only in its first input (thereby limiting its ability to read the full auxiliary input). This equivalence does not necessarily hold in our context. Since our simulator is a standard PPT algorithm that runs in time polynomial in the length of its entire input, restricting the distinguisher to the same auxiliary input $\mathbf{aux}$ as provided to the simulator results in a weaker security guarantee.

2. Emulate $(m_1, m_2) \leftarrow \text{Sim}(x, \text{aux})$, where $\text{aux} := (\text{crs}, x)$.
3. Output 1 if and only if $\text{FS.V}(\text{crs}, x, (m_1, m_2))$ accepts.

For instances $x \in \mathcal{L}$: By the completeness of the transformation, the output of the interaction $\langle \mathbf{P}(x), \mathbf{V}^*(x, \text{aux}) \rangle$ must result in a transcript (m_1, m_2) such that $\text{FS.V}(\text{crs}, x, m_1, m_2) = 1$. By the zero-knowledge property, since $\mathbf{D}$ receives (crs, x) as its auxiliary input and distinguishes between the real interaction and the simulated transcript with only negligible probability, it follows that $\text{Sim}(x, \text{crs})$ must also output a transcript (m_1, m_2) such that $\text{FS.V}(\text{crs}, x, m_1, m_2) = 1$ with all but negligible probability.

For instances $x \notin \mathcal{L}$: Assume towards contradiction that for some instance $x \notin \mathcal{L}$ the algorithm outputs 1 with non-negligible probability. Then, an adversary for that transformed protocol that gets $\text{crs} \leftarrow \text{FS.Setup}(1^n, \mathbf{V})$ as an input and emulates the simulator as defined in the algorithm $\mathbf{A}$, will make $\text{FS.V}(\text{crs}, \cdot)$ accept with non-negligible probability. This contradicts the ϵ' soundness of the transformed protocol. Therefore, for every $x \notin \mathcal{L}$, the algorithm must output 0 with overwhelming probability.

Overall, since $\mathbf{A}$ is a PPT algorithm that decides $\mathcal{L}$ with overwhelming probability, we get that $\mathcal{L} \in \text{BPP}$. $\square$

8 Statistical Difference Problem

In this section, we construct an honest-verifier statistical zero-knowledge non-interactive argument for the Statistical Difference problem SD (see Definition 3.13). Recall that SD is complete for the class of statistical zero-knowledge (SZK) [SV03]. Consequently, our result extends to all languages in SZK. We present the protocol in Section 8.1, analyze its completeness and soundness in Section 8.2, and prove zero-knowledge in Section 8.3.

Definition 8.1 (Statistical Difference Problem). *Let C be a circuit and $\mathcal{D}_C$ denote the distribution of its output on uniform input. The statistical difference problem is defined by the following two sets:*

$$\begin{aligned} \text{SD}_Y &:= \left\{ (C_0, C_1) \mid \Delta(\mathcal{D}_{C_0}, \mathcal{D}_{C_1}) \geq \frac{2}{3} \right\} \\ \text{SD}_N &:= \left\{ (C_0, C_1) \mid \Delta(\mathcal{D}_{C_0}, \mathcal{D}_{C_1}) \leq \frac{1}{3} \right\}. \end{aligned}$$

The statistical difference problem is the problem of deciding, given a pair $(C_0, C_1) \in \text{SD}_Y \cup \text{SD}_N$, whether $(C_0, C_1) \in \text{SD}_Y$ or $(C_0, C_1) \in \text{SD}_N$. The behavior on inputs outside $\text{SD}_Y \cup \text{SD}_N$ is undefined.

Theorem 8.2. *Assume the existence of a secure puncturable pseudorandom function and indistinguishability obfuscation. Then, the protocol described in Construction 8.4 is an honest-verifier statistical zero-knowledge non-interactive argument for the SD problem.*

Note that we can replace puncturable PRFs with a weaker complexity assumption. Indeed, by [KMN+22], iO together with $\text{NP} \not\subseteq \text{io-BPP}$ implies one-way functions, and hence also puncturable PRFs.

8.1 Protocol

Our construction on the interactive protocol introduced by Sahai and Vadhan [SV03]. The construction is the non-interactive argument generated by applying the Fiat–Shamir transformation described in Section 4.1 on an amplified version of the interactive protocol for the SD problem introduced in [SV03]. In their construction, they use the following polarization lemma.

Lemma 8.3 (Polarization Lemma [SV00]). *There exists a deterministic polynomial time computable function $\text{Polarize}(\cdot, \cdot, \cdot)$ that, given circuits C_0, C_1 and a parameter $\lambda \in \mathbb{N}$, outputs circuits $C_0^\lambda, C_1^\lambda := \text{Polarize}(C_0, C_1, 1^\lambda)$ satisfying:*

$$\begin{aligned} \Delta(\mathcal{D}_{C_0}, \mathcal{D}_{C_1}) \geq \frac{2}{3} &\implies \Delta(\mathcal{D}_{C_0^\lambda}, \mathcal{D}_{C_1^\lambda}) \geq 1 - 2^{-\lambda}, \\ \Delta(\mathcal{D}_{C_0}, \mathcal{D}_{C_1}) < \frac{1}{3} &\implies \Delta(\mathcal{D}_{C_0^\lambda}, \mathcal{D}_{C_1^\lambda}) < 2^{-\lambda}. \end{aligned}$$

In terms of randomness complexity, if sampling from $\mathcal{D}_{C_b}$ requires n bits, then sampling from the polarized distributions requires $(n + 1)\lambda - 1$ bits.

Note that for every $b \in \{0, 1\}$, the polarized distribution in the lemma above is the following distribution,

$$\mathcal{D}_{C_b^\lambda} := \left\{ (C_{b_1}(z_1), \dots, C_{b_\lambda}(z_\lambda)) \mid \begin{array}{l} b_1 \dots, b_{\lambda-1} \leftarrow \{0, 1\} \\ b_\lambda := b_1 \dots, b_{\lambda-1} \oplus b \\ z_1, \dots, z_\lambda \leftarrow \{0, 1\}^{s_{\text{in}}} \end{array} \right\}.$$

We now turn to introduce our construction. Let iO be an indistinguishability obfuscator for Boolean circuits. Let $\Pi_{\text{PRF}} = (\text{F.Gen}, \text{F.Eval}, \text{F.Punc})$ be a puncturable pseudorandom function. For a key $\mathbf{k}$ and input x , denote the PRF evaluation by $\text{F}(\mathbf{k}, x) := \text{F.Eval}(\mathbf{k}, x)$.

Construction 8.4. The protocol is as follows.

- **Setup**($1^\lambda, 1^n, 1^{s_{\text{in}}}$):
 1. Let $R := ((s_{\text{in}} + 1)\lambda - 1)$.
 2. Compute $\mathbf{k}_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^1 := \lambda \cdot R$.
 3. Compute $\mathbf{k}_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^2 := \lambda$.
 4. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[\mathbf{k}_1, \mathbf{k}_2])$, where $s = \text{poly}(n, \lambda)$.
 5. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[\mathbf{k}_1])$, where $s = \text{poly}(n, \lambda)$.
 6. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.
- **Chal**[\mathbf{k}_1, \mathbf{k}_2](x):
 1. Parse x as (C_1, C_2) .
 2. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 3. Compute $(r_1, \dots, r_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_1, x)$, where $r_i \in \{0, 1\}^R$.
 4. Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_2, x)$, where $b_i \in \{0, 1\}$.
 5. Output $Y := (y_1, \dots, y_\lambda)$, where $y_i \leftarrow C_{b_i}^\lambda(r_i)$.
- **Ver**[\mathbf{k}_2](x, π):
 1. Parse x as (C_1, C_2) .
 2. Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_2, x)$, where $b_i \in \{0, 1\}$.
 3. If $\pi = (b_1, \dots, b_\lambda)$, output 1.
 4. Else, output 0.
- **Prove**(crs, x):
 1. Parse crs as $(\widehat{\text{Chal}}, \widehat{\text{Ver}})$.
 2. Parse x as (C_1, C_2) .
 3. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 4. Compute $Y := \widehat{\text{Chal}}(x)$, and parse Y as $(y_1, \dots, y_\lambda)$.
 5. For each $i \in [\lambda]$,
 - (a) If $\Pr[y_i \in C_0^\lambda] > \Pr[y_i \in C_1^\lambda]$, set $b_i := 0$.
 - (b) Else, set $b_i := 1$.
 6. Output $\pi := (b_1, \dots, b_\lambda)$.
- **Verify**(crs, x, π):
 1. Accept if and only if $\widehat{\text{Ver}}(x, \pi) = 1$.

8.2 Analysis

We prove non-adaptive soundness and completeness.

Non-adaptive soundness. Let $\lambda \in \mathbb{N}$ be the security parameter. Let $n = \text{poly}(\lambda)$ be the instance size, and let $s_{\text{in}} \leq n$ be the circuit input size. Fix a PPT adversary $\tilde{\mathbf{P}}$ and a false instance $x^* = (C_0, C_1) \in \text{SD}_{\mathbb{N}} \cap \{0, 1\}^n$. Let $\text{PRF} := (\text{F.Gen}, \text{PRF.F.Eval})$ be a pseudo random function, and

let iO be an indistinguishability obfuscator, and let $\text{OWF} := (\text{OWF.Gen}, f)$ be an injective one-way function. Note that we assume iO and OWF , and therefore, there exists an injective one-way function as constructed in [BPW16].

Hybrid 1.

- $\text{Setup}_1(1^\lambda, 1^n, 1^{s_{\text{in}}})$:
 1. Let $R := ((s_{\text{in}} + 1)\lambda - 1)$.
 2. Compute $k_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^1 := \lambda \cdot R$.
 3. Compute $k_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^2 := \lambda$.
 4. Compute $(r_1^*, \dots, r_\lambda^*) \leftarrow \text{F.Eval}(k_1, x^*)$.
 5. Compute $(b_1^*, \dots, b_\lambda^*) \leftarrow \text{F.Eval}(k_2, x^*)$, and set $\pi^* := (b_1^*, \dots, b_\lambda^*)$.
 6. Puncture $k_1^* \leftarrow \text{F.Punc}(k_1, x^*)$.
 7. Puncture $k_2^* \leftarrow \text{F.Punc}(k_2, x^*)$.
 8. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 9. Compute $Y^* := (y_1^*, \dots, y_\lambda^*)$, where $y_i^* := C_{b_i^*}^\lambda(r_i^*)$.
 10. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[k_1^*, k_2^*, x^*, Y^*])$, where $s = \text{poly}(n, \lambda)$.
 11. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[k_2^*, x^*, \pi^*])$, where $s = \text{poly}(n, \lambda)$.
 12. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.
- $\text{Chal}[k_1^*, k_2^*, x^*, Y^*](x)$:
 1. Parse x as (C_1, C_2) .
 2. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 3. If $x = x^*$,
 - (a) Output Y^* .
 4. Else
 - (a) Compute $(r_1, \dots, r_\lambda) \leftarrow \text{F.Eval}(k_1^*, x)$ with $r_i \in \{0, 1\}^R$.
 - (b) Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(k_2^*, x)$ with $b_i \in \{0, 1\}$.
 - (c) Output $Y := (y_1, \dots, y_\lambda)$, where $y_i := C_{b_i}^\lambda(r_i)$.
- $\text{Ver}[k_2^*, x^*, \pi^*](x, \pi)$:
 1. Parse x as (C_1, C_2) .
 2. If $x = x^*$, then
 - (a) If $\pi = \pi^*$, output 1.
 - (b) Else, output 0.
 3. Else,
 - (a) Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(k_2^*, x)$ with $b_i \in \{0, 1\}$.
 - (b) If $\pi = (b_1, \dots, b_\lambda)$, output 1.
 - (c) Else, output 0.

Claim 8.5. *It holds that,*

$$\left| \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_1(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \right| \leq \text{negl}(\lambda) .$$

Proof. We argue that the view of the adversary in Hybrid 1 is computationally indistinguishable from the construction. Observe that the circuits obfuscated in Hybrid 1, $\text{Chal}[\mathbf{k}_1^*, \mathbf{k}_2^*, x^*, Y^*]$ and $\text{Ver}[\mathbf{k}_2^*, x^*, \pi^*]$, are functionally equivalent to the original circuits $\text{Chal}[\mathbf{k}_1, \mathbf{k}_2]$ and $\text{Ver}[\mathbf{k}_2]$.

Specifically, on input x^* , the hardcoded values Y^* and π^* in Hybrid 1 match exactly the output of the original circuits (by definition of the PRF evaluation). For any input $x \neq x^*$, the punctured keys $\mathbf{k}_1^*, \mathbf{k}_2^*$ preserve the evaluation functionality of the original keys. Therefore, by the security guarantee of the Indistinguishability Obfuscation scheme, the obfuscations in both experiments are computationally indistinguishable. $\square$

Hybrid 2.

- $\text{Setup}_2(1^\lambda, 1^n, 1^{s_{\text{in}}})$:
 1. Let $R := ((s_{\text{in}} + 1)\lambda - 1)$.
 2. Compute $\mathbf{k}_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^1 := \lambda \cdot R$.
 3. Compute $\mathbf{k}_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^2 := \lambda$.
 4. Sample $(r_1^*, \dots, r_\lambda^*) \leftarrow \{0, 1\}^{\lambda \times R}$ uniformly at random.
 5. Sample $(b_1^*, \dots, b_\lambda^*) \leftarrow \{0, 1\}^\lambda$ uniformly at random, and set $\pi^* := (b_1^*, \dots, b_\lambda^*)$.
 6. Puncture $\mathbf{k}_1^* \leftarrow \text{F.Punc}(\mathbf{k}_1, x^*)$.
 7. Puncture $\mathbf{k}_2^* \leftarrow \text{F.Punc}(\mathbf{k}_2, x^*)$.
 8. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 9. Compute $Y^* := (y_1^*, \dots, y_\lambda^*)$, where $y_i^* := C_{b_i^*}^\lambda(r_i^*)$.
 10. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[\mathbf{k}_1^*, \mathbf{k}_2^*, x^*, Y^*])$, where $s = \text{poly}(n, \lambda)$.
 11. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[\mathbf{k}_2^*, x^*, \pi^*])$, where $s = \text{poly}(n, \lambda)$.
 12. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.
- $\text{Chal}[\mathbf{k}_1^*, \mathbf{k}_2^*, x^*, Y^*](x)$:
 1. Parse x as (C_1, C_2) .
 2. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 3. If $x = x^*$,
 - (a) Output Y^* .
 4. Else,
 - (a) Compute $(r_1, \dots, r_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_1^*, x)$, where $r_i \in \{0, 1\}^R$.
 - (b) Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_2^*, x)$, where $b_i \in \{0, 1\}$.
 - (c) Output $Y := (y_1, \dots, y_\lambda)$, where $y_i := C_{b_i}^\lambda(r_i)$.
- $\text{Ver}[\mathbf{k}_2^*, x^*, \pi^*](x, \pi)$:
 1. Parse x as (C_1, C_2) .
 2. If $x = x^*$, then
 - (a) If $\pi = \pi^*$, output 1.
 - (b) Else, output 0.
 3. Else,
 - (a) Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_2^*, x)$, where $b_i \in \{0, 1\}$.
 - (b) If $\pi = (b_1, \dots, b_\lambda)$, output 1.
 - (c) Else, output 0.

Claim 8.6. *It holds that,*

$$\left| \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_1(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_2(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \right| \leq \text{negl}(\lambda) .$$

Proof. In Hybrid 2, we replace the PRF values in location x^* with uniformly random bits. Indistinguishability follows by the puncturable PRF security. $\square$

Hybrid 3.

- $\text{Setup}_3(1^\lambda, 1^n, 1^{s_{\text{in}}})$:
 1. Let $R := ((s_{\text{in}} + 1)\lambda - 1)$.
 2. Compute $\mathbf{k}_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^1 := \lambda \cdot R$.
 3. Compute $\mathbf{k}_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^2 := \lambda$.
 4. Sample $(r_1^*, \dots, r_\lambda^*) \leftarrow \{0, 1\}^{\lambda \times R}$ uniformly at random.
 5. Sample $(b_1^*, \dots, b_\lambda^*) \leftarrow \{0, 1\}^\lambda$ uniformly at random, and set $\pi^* := (b_1^*, \dots, b_\lambda^*)$.
 6. Puncture $\mathbf{k}_1^* \leftarrow \text{F.Punc}(\mathbf{k}_1, x^*)$.
 7. Puncture $\mathbf{k}_2^* \leftarrow \text{F.Punc}(\mathbf{k}_2, x^*)$.
 8. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 9. **Compute** $Y^* := (y_1^*, \dots, y_\lambda^*)$, where $y_i^* := C_0^\lambda(r_i^*)$.
 10. Puncture $\mathbf{k}_1^* \leftarrow \text{F.Punc}(\mathbf{k}_1, x^*)$ and $\mathbf{k}_2^* \leftarrow \text{F.Punc}(\mathbf{k}_2, x^*)$.
 11. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[\mathbf{k}_1^*, \mathbf{k}_2^*, x^*, Y^*])$, where $s = \text{poly}(n, \lambda)$.
 12. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[\mathbf{k}_2^*, x^*, \pi^*])$, where $s = \text{poly}(n, \lambda)$.
 13. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.
- $\text{Chal}[\mathbf{k}_1^*, \mathbf{k}_2^*, x^*, Y^*](x)$:
 1. Parse x as (C_1, C_2) .
 2. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 3. If $x = x^*$,
 - (a) Output Y^* .
 4. Else,
 - (a) Compute $(r_1, \dots, r_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_1^*, x)$, where $r_i \in \{0, 1\}^R$.
 - (b) Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_2^*, x)$, where $b_i \in \{0, 1\}$.
 - (c) Output $Y := (y_1, \dots, y_\lambda)$, where $y_i := C_{b_i}^\lambda(r_i)$.
- $\text{Ver}[\mathbf{k}_2^*, x^*, \pi^*](x, \pi)$:
 1. Parse x as (C_1, C_2) .
 2. If $x = x^*$, then
 - (a) If $\pi = \pi^*$, output 1.
 - (b) Else, output 0.
 3. Else,
 - (a) Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(\mathbf{k}_2^*, x)$, where $b_i \in \{0, 1\}$.
 - (b) If $\pi = (b_1, \dots, b_\lambda)$, output 1.
 - (c) Else, output 0.

Claim 8.7. *It holds that,*

$$\left| \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_2(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_3(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \right| \leq \text{negl}(\lambda) .$$

Proof. In Setup_3 we change the generation of the value $Y^* = (y_1^*, \dots, y_\lambda^*)$, by replacing the computation of $y_i^* \leftarrow C_{b_i^*}^\lambda(r_i^*)$ with $y_i^* \leftarrow C_0^\lambda(r_i^*)$.

To bound the statistical distance between these distributions, we define a sequence of intermediate distributions. Let $\mathcal{D}_0$ denote the distribution of Y^* in Setup_2 . For every $j \in [\lambda]$, let $\mathcal{D}_j$ be identical to $\mathcal{D}_{j-1}$, except that the j -th component y_j^* is sampled from $C_0^\lambda(r_j^*)$ instead of $C_{b_j^*}^\lambda(r_j^*)$. Note that $\mathcal{D}_\lambda$ corresponds exactly to the distribution of Y^* in Setup_3 . Thus, it suffices to prove that:

$$\Delta(\mathcal{D}_0, \mathcal{D}_\lambda) \leq \text{negl}(\lambda) .$$

Consider the transition from $\mathcal{D}_{j-1}$ to $\mathcal{D}_j$. Since the instance is invalid ($x^* \in \text{SD}_\mathbf{N}$), we have $\Delta(\mathcal{D}_{C_0}, \mathcal{D}_{C_1}) < 1/3$. Therefore, by the Polarization Lemma (Lemma 8.3),

$$\Delta(\mathcal{D}_{C_0^\lambda}, \mathcal{D}_{C_1^\lambda}) < 2^{-\lambda} .$$

Consequently, regardless of the specific value of the challenge bit b_j^* , replacing the source distribution $\mathcal{D}_{C_{b_j^*}^\lambda}$ with $\mathcal{D}_{C_0^\lambda}$ introduces a statistical difference of at most $2^{-\lambda}$ for the j -th component. Therefore, for every step j :

$$\Delta(\mathcal{D}_{j-1}, \mathcal{D}_j) < 2^{-\lambda} .$$

By the triangle inequality, summing over all λ components yields:

$$\Delta(\mathcal{D}_0, \mathcal{D}_\lambda) \leq \sum_{j=1}^{\lambda} \Delta(\mathcal{D}_{j-1}, \mathcal{D}_j) < \lambda \cdot 2^{-\lambda} = \text{negl}(\lambda) .$$

Since the statistical distance between the distributions is negligible, the distributions of the crs in Setup_2 and Setup_3 are statistically close. Consequently, no adversary (whether computationally bounded or unbounded) can distinguish between the two experiments with more than negligible probability. In particular, we get that,

$$\left| \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_2(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_3(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \right| \leq \text{negl}(\lambda) ,$$

which concludes the proof. $\square$

Hybrid 4.

- $\text{Setup}_4(1^\lambda, 1^n, 1^{s_{\text{in}}})$:
 1. Let $R := ((s_{\text{in}} + 1)\lambda - 1)$.
 2. Compute $k_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^1 := \lambda \cdot R$.
 3. Compute $k_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^2 := \lambda$.
 4. Sample $(r_1^*, \dots, r_\lambda^*) \leftarrow \{0, 1\}^{\lambda \times R}$ uniformly at random.
 5. Sample $(b_1^*, \dots, b_\lambda^*) \leftarrow \{0, 1\}^\lambda$ uniformly at random, and set $\pi^* := (b_1^*, \dots, b_\lambda^*)$.
 6. Puncture $k_1^* \leftarrow \text{F.Punc}(k_1, x^*)$.
 7. Puncture $k_2^* \leftarrow \text{F.Punc}(k_2, x^*)$.
 8. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 9. Compute $Y^* := (y_1^*, \dots, y_\lambda^*)$, where $y_i^* := C_0^\lambda(r_i^*)$.
 10. **Run $s \leftarrow \text{OWF.Gen}(1^\lambda)$.**
 11. **Compute $v \leftarrow f_s(b_1^*, \dots, b_\lambda^*)$.**
 12. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[k_1^*, k_2^*, x^*, Y^*])$, where $s = \text{poly}(n, \lambda)$.
 13. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[k_2^*, x^*, f_s, v])$, where $s = \text{poly}(n, \lambda)$.
 14. Output $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.
- $\text{Chal}[k_1^*, k_2^*, x^*, Y^*](x)$:
 1. Parse x as (C_1, C_2) .
 2. Compute $(C_0^\lambda, C_1^\lambda) \leftarrow \text{Polarize}(C_0, C_1, 1^\lambda)$.
 3. If $x = x^*$,
 - (a) Output Y^* .
 4. Else,
 - (a) Compute $(r_1, \dots, r_\lambda) \leftarrow \text{F.Eval}(k_1^*, x)$, where $r_i \in \{0, 1\}^R$.
 - (b) Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(k_2^*, x)$, where $b_i \in \{0, 1\}$.
 - (c) Output $Y := (y_1, \dots, y_\lambda)$, where $y_i := C_0^\lambda(r_i)$.
- $\text{Ver}[k_2^*, x^*, f_s, v](x, \pi)$:
 1. Parse x as (C_1, C_2) .
 2. If $x = x^*$, then
 - (a) **If $f_s(\pi) = v$, output 1.**
 - (b) Else, output 0.
 3. Else,
 - (a) Compute $(b_1, \dots, b_\lambda) \leftarrow \text{F.Eval}(k_2^*, x)$, where $b_i \in \{0, 1\}$.
 - (b) If $\pi = (b_1, \dots, b_\lambda)$, output 1.
 - (c) Else, output 0.

Claim 8.8. *It holds that,*

$$\left| \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_3(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - \Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_4(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \right| \leq \text{negl}(\lambda) .$$

Proof. The only difference between Setup_3 and Setup_4 is the generation of the circuit Ver . In Setup_3 , the circuit has the value π^* hardwired and it accepts an input π if and only if $\pi = \pi^*$. In Setup_4 , the circuit has the value $v = f_s(\pi^*)$ hardwired and accepts π if and only if $f_s(\pi) = v$.

Since f is an injective function, the condition $f_s(\pi) = f_s(\pi^*)$ holds if and only if $\pi = \pi^*$. Therefore, the two circuits implement the exact same functionality. By the iO security, the distributions of the obfuscated circuits in both hybrids are computationally indistinguishable. $\square$

To conclude the proof we prove the following claim.

Claim 8.9. *It holds that,*

$$\Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}_4(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \leq \text{negl}(\lambda) .$$

Proof. We now show that the success probability of the adversary in Hybrid 4 is negligible. In Setup_4 , the value π^* is sampled uniformly at random from $\{0, 1\}^\lambda$. Note that the only information about π^* available to the adversary is the image $v = f_s(\pi^*)$ hardcoded in $\widehat{\text{Ver}}$.

Consequently, for an adversary to output a proof π such that $\mathbf{V}(\text{crs}, x^*, \pi) = 1$, they must produce π such that $f_s(\pi) = v$. Since π^* is uniform and independent of the rest of the view, this is equivalent to inverting the one-way function on a random image. By the security of the OWF, this probability is negligible. $\square$

By combining Claim 8.5 to Claim 8.8, and bounding the final hybrid via Claim 8.9, we conclude that the soundness error of the protocol is negligible.

Completeness. Fix a security parameter $\lambda \in \mathbb{N}$ and an instance $x^* = (C_0, C_1) \in \text{SD}_Y$. By the definition of the problem, $\Delta(\mathcal{D}_{C_0}, \mathcal{D}_{C_1}) \geq 2/3$. We assume without loss of generality that the PRF is 1-universal. That is, for any y , the distribution $\{\text{PRF}_k(y) \mid k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})\}$ is uniform (the transformation from PRF to 1-universal PRF is the same as described in [CLW18][Construction 3.10]).

Since $\Delta(\mathcal{D}_{C_0}, \mathcal{D}_{C_1}) \geq 2/3$, the Polarization Lemma (Lemma 8.3) implies that the statistical distance between the polarized distributions is amplified to:

$$\delta := \Delta(\mathcal{D}_{C_0^\lambda}, \mathcal{D}_{C_1^\lambda}) \geq 1 - 2^{-\lambda} .$$

The challenge that the honest prover gets for instance x^* is the challenge $Y = (y_1, \dots, y_\lambda)$, where $y_i := C_{b_i}^\lambda(r_i)$. Note that r_i is computed by $(r_1, \dots, r_\lambda) \leftarrow \text{F.Eval}(k_1, x)$. Since the PRF is 1-universal, the distribution of $(r_1, \dots, r_\lambda)$ is uniform. Therefore, the challenge y_i is a uniform sample from $\mathcal{D}_{C_{b_i}^\lambda}$.

The honest prover strategy is the Maximum Likelihood decision rule. For each $i \in [\lambda]$, the prover sets $b'_i = 0$ if the sample y_i is more likely to have come from $\mathcal{D}_{C_0^\lambda}$, and 1 otherwise. It is a standard result (e.g., [SV03, Lemma 3.7], [BY25, Claim 5.2]) that for any two distributions with statistical distance δ , the optimal predictor is the Maximum Likelihood, and that this strategy succeeds with probability exactly $\frac{1+\delta}{2}$.

Therefore, the probability that the prover correctly identifies a single bit (i.e., $b'_i = b_i^*$) is:

$$\Pr[b'_i = b_i^*] = \frac{1 + \delta}{2} \geq \frac{1 + (1 - 2^{-\lambda})}{2} = 1 - 2^{-(\lambda+1)} .$$

The verifier accepts if the prover is correct for all λ indices. By applying a union bound, the probability that the prover guesses incorrectly in one of the bits is,

$$\Pr[\exists i : b'_i \neq b_i^*] \leq \sum_{i=1}^{\lambda} \Pr[b'_i \neq b_i^*] \leq \lambda \cdot 2^{-(\lambda+1)} = \text{negl}(\lambda) .$$

Therefore, the honest prover succeeds with overwhelming probability,

$$\Pr \left[\mathbf{V}(\text{crs}, x^*, \pi) = 1 \mid \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \mathbf{P}(\text{crs}) \end{array} \right] \geq 1 - \text{negl}(\lambda) .$$

8.3 Statistical Zero-Knowledge in the Honest-Verifier Setting

In this section we prove that the non-interactive argument for the SD problem introduced in Construction 8.4 also satisfies the zero-knowledge property as defined in Definition 3.13.

Theorem 8.10. *The SD protocol presented in Construction 8.4 satisfies honest-verifier statistical zero-knowledge as defined in Definition 3.13.*

Proof. Let $\Pi = (\text{FS.Setup}, \text{FS.Prove}, \text{FS.Verify})$ be the SD protocol from Construction 8.4. We construct a PPT simulator $\text{Sim}(\cdot)$ and prove that its output distribution is statistically indistinguishable from that of the real protocol.

We define the simulator algorithm as follows:

- $\text{Sim}(1^\lambda, x)$:
 1. Let $R := ((s_{\text{in}} + 1)\lambda - 1)$.
 2. Compute $k_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^1 := \lambda \cdot R$.
 3. Compute $k_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2})$, where $\ell_{\text{in}} := n$, and $\ell_{\text{out}}^2 := \lambda$.
 4. Compute $\widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[k_1, k_2])$.
 5. Compute $\widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[k_2])$.
 6. Compute $\pi \leftarrow \text{F.Eval}(k_2, x)$.
 7. Output (crs, π) , where $\text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}})$.

Fix an instance $x \in \text{SD}_Y$. Let $\mathcal{D}_{\text{Sim}}$ denote the simulated distribution:

$$\mathcal{D}_{\text{Sim}} := \{(\text{crs}, \pi) \mid (\text{crs}, \pi) \leftarrow \text{Sim}(1^\lambda, x)\} .$$

Let $\mathcal{D}_{\text{Real}}$ denote the real world distribution:

$$\mathcal{D}_{\text{Real}} := \left\{ (\text{crs}, \pi) \mid \begin{array}{l} \text{crs} \leftarrow \text{FS.Setup}(1^\lambda, 1^n, 1^{s_{\text{in}}}) \\ \pi \leftarrow \text{FS.P}(\text{crs}, x) \end{array} \right\} .$$

Observe that the distributions can be expanded as follows:

$$\begin{aligned} \mathcal{D}_{\text{Real}} &= \left\{ (\text{crs}, \pi) \mid \begin{array}{l} k_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1}) \\ k_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2}) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[k_1, k_2]) \\ \widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[k_2]) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi \leftarrow \text{FS.Prove}(\text{crs}, x) \end{array} \right\} , \\ \mathcal{D}_{\text{Sim}} &= \left\{ (\text{crs}, \pi) \mid \begin{array}{l} k_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1}) \\ k_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2}) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[k_1, k_2]) \\ \widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[k_2]) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi \leftarrow \text{F.Eval}(k_2, x) \end{array} \right\} . \end{aligned}$$

The crs is distributed identically in both experiments. The only difference is in the generation of the proof π . Recall that the verifier FS.Verify uses the functionality of $\text{Ver}[k_2]$ to verify the proof, accepting a proof π if and only if $\pi = \text{F.Eval}(k_2, x)$. Since $x \in \text{SD}_Y$, by the completeness of the protocol, the verifier accepts the proof generated by the honest prover $\mathbf{P}$ with all but negligible probability. Consequently, in the distribution $\mathcal{D}_{\text{Real}}$, it holds that $\pi = \text{F.Eval}(k_2, x)$ with all but negligible probability. Therefore, we obtain:

$$\begin{aligned} & \left\{ (\text{crs}, \pi) \left| \begin{array}{l} k_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1}) \\ k_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2}) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[k_1, k_2]) \\ \widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[k_2]) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi \leftarrow \text{FS.Prove}(\text{crs}, x) \end{array} \right. \right\} \\ & \equiv_{\text{negl}(\lambda)} \left\{ (\text{crs}, \pi) \left| \begin{array}{l} k_1 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^1}) \\ k_2 \leftarrow \text{F.Gen}(1^\lambda, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}^2}) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Chal}[k_1, k_2]) \\ \widehat{\text{Ver}} \leftarrow \text{iO}(1^\lambda, 1^s, \text{Ver}[k_2]) \\ \text{crs} := (\widehat{\text{Chal}}, \widehat{\text{Ver}}) \\ \pi \leftarrow \text{F.Eval}(k_2, x) \end{array} \right. \right\}. \end{aligned}$$

The left distribution corresponds to $\mathcal{D}_{\text{Real}}$, and the right corresponds to $\mathcal{D}_{\text{Sim}}$. We conclude that $\Delta(\mathcal{D}_{\text{Sim}}, \mathcal{D}_{\text{Real}}) \leq \text{negl}(\lambda)$, as required. $\square$

9 A Fiat–Shamir Transformation for Public-Coin Protocols

While our primary focus is private-coin protocols, our framework also provides a fresh perspective on transforming public-coin protocols. Kalai et al. [KRR17] introduced the first Fiat–Shamir transformation for 3-message public-coin protocols achieving negligible soundness, relying on iO and PRF. Their original soundness analysis required an additional assumption that an input-hiding obfuscation scheme exists for the family of point functions.

In this section, we present an alternative analysis of the same construction from [KRR17]. Crucially, our analysis removes this additional assumption regarding input-hiding obfuscation. Instead, our construction requires only sufficiently strong soundness of the underlying protocol (which can be achieved using parallel repetition). The construction appears in Section 9.1, and the corresponding soundness analysis is given in Section 9.2.

Theorem 9.1. *Let Π be a 3-message public-coin protocol for instances of size n , in which the first prover message has length ℓ_1 , and its soundness error ϵ .*

Assume the existence of a sub-exponentially secure puncturable pseudorandom function and sub-exponentially secure indistinguishability obfuscation. Then, the non-interactive argument obtained by applying the Fiat-Shamir transformation described in Section 9.1 to Π is $2^{(n+\ell_1)} \cdot \epsilon + \text{negl}(\lambda)$ secure.

Remark 9.2. *Assume the existence of a secure puncturable pseudorandom function and secure indistinguishability obfuscation. If Π is a 3-message private-coin protocol that satisfies statistical zero-knowledge in the honest-verifier setting as defined in Definition 3.11, then the non-interactive protocol generated by applying the Fiat-Shamir transformation from Section 9.1 to Π satisfies computational honest-verifier zero-knowledge as defined in Definition 3.14.*

The proof follows exactly the same steps as the proof for Theorem 4.7.

9.1 Protocol

Let $(\mathbf{P} = (\mathbf{P}_1, \mathbf{P}_2), \mathbf{V})$ be a 3-message public coin protocol for some language $\mathcal{L}$, where for instance size n the protocol has the following properties:

- Soundness error ϵ .
- First prover message includes ℓ_1 bits.
- Verifier runs in $2^{O(n)}$ time.

We assume the following:

- Sub-exponentially secure indistinguishability obfuscator iO.
- Sub-exponentially secure puncturable pseudorandom function $\Pi_{\text{PPRF}} = (\text{F.Gen}, \text{F.Eval}, \text{F.Punc})$. For a key k and input x , denote the PRF evaluation by $F_k(x) \leftarrow \text{F.Eval}(k, x)$.

By the sub-exponential security of the iO and the PPRF, there exists some constant $0 < c < 1$ such that the iO and PPRF schemes are T -secure for $T(\lambda) \geq 2^{\lambda^c}$.

Construction 9.3. The non-interactive protocol is the following.

- $\text{Setup}(1^\lambda, (\mathbf{P}, \mathbf{V}))$:
 1. Compute $k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$, for $\lambda_{\text{PRF}} := (2(\lambda + n + \ell_1))^{\frac{1}{c}}$, $\ell_{\text{in}} := n + \ell_1$, and $\ell_{\text{out}} := r$.

2. Compute $\widehat{\text{Chal}}[k] \leftarrow \text{iO}(1^{\lambda_{\text{io}}}, 1^s, \text{Chal}[k])$, for $\lambda_{\text{io}} := (2(\lambda + n + \ell_1))^{\frac{1}{c}}$, $s := \text{poly}(\lambda, n, \ell_1, r)$ for some large enough polynomial.
 3. Output $\text{crs} := \widehat{\text{Chal}}[k]$.
- $\text{Chal}[k](x, m_1)$:
 1. Output $\rho \leftarrow F_k(x \| m_1)$.
 - $\text{Prove}(\text{crs}, x)$:
 1. Parse crs as $\widehat{\text{Chal}}[k]$.
 2. Compute,
 - (a) $(m_1, \text{aux}) \leftarrow P_1(x)$.
 - (b) $\rho \leftarrow \widehat{\text{Chal}}[k](x, m_1)$.
 - (c) $m_2 \leftarrow P_2(x, \text{aux}, a)$.
 3. Output $\pi := (m_1, m_2)$.
 - $\text{Verify}(\text{crs}, x, \pi)$:
 1. Parse crs as $\widehat{\text{Chal}}[k]$.
 2. Parse π as (m_1, m_2) .
 3. Compute $\rho \leftarrow \widehat{\text{Chal}}[k](x, m_1)$.
 4. Output $V(x, m_1, m_2; \rho)$.

9.2 Adaptive Soundness

Proof. Let (P, V) be a 3-message public-coin protocol for a language $\mathcal{L}$. Let ℓ_1 and ℓ_2 denote the lengths of the first and second prover messages, respectively, and let r denote the number of random bits used by the verifier. Let n denote the instance size. We assume that the verifier runs on time $2^{O(n)}$.

Assume towards contradiction that there exists an adversary $\tilde{P}$ such that for some function $\nu(\lambda) \geq 2^{(n+\ell_1)} \cdot \epsilon + 1/\text{poly}(\lambda)$,

$$\Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ \widehat{\text{Ver}}[k](x, m_1, m_2) = 1 \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, V) \\ \widehat{\text{Chal}}[k] := \text{crs} \\ (x, m_1, m_2) \leftarrow \tilde{P}(\text{crs}) \end{array} \right] = \nu(\lambda) . \quad (11)$$

Let ACC be the following set,

$$\text{ACC} := \{ (x, m_1, \rho, m_2) \mid V(x, m_1, m_2; \rho) = 1 \} .$$

Therefore, by construction and by Equation 11, we get that,

$$\Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ (x, m_1, F_k(x \| m_1), m_2) \in \text{ACC} \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, V) \\ (x, m_1, m_2) \leftarrow \tilde{P}(\text{crs}) \end{array} \right] = \nu(\lambda) . \quad (12)$$

Hybrid 1.

- $\text{Setup}_1[x^*, m_1^*](1^\lambda, V)$:
 1. Run $k \leftarrow F.\text{Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$.
 2. Let $\rho^* \leftarrow F_k(x^* \| m_1^*)$.

3. **Puncture** $k^* \leftarrow \text{F.Punc}(k, (x^* \| m_1^*))$.
4. Compute $\widehat{\text{Chal}}[k^*, x^*, m_1^*, \rho^*]$.
5. Output $\text{crs} := \widehat{\text{Chal}}[k^*, x^*, m_1^*, \rho^*]$.
- $\text{Chal}[k^*, x^*, m_1^*, \rho^*](x, m_1)$:
 1. **If** $(x^*, m_1^*) = (x, m_1)$,
 - (a) **Output** ρ^* .
 2. **Else**
 - (a) Output $\rho \leftarrow F_{k^*}(x \| m_1)$.

Claim 9.4. Fix some $x^* \in \{0, 1\}^n$ and $m_1^* \in \{0, 1\}^{\ell_1}$. It holds that

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - 2^{-2(\lambda+n+\ell_1)},$$

where k is the PRF key generated by the setup.

Proof. In the construction, the adversary gets the obfuscation of the circuit $\text{Chal}[k]$, and in Hybrid 1 the adversary gets the obfuscation of the circuits $\text{Chal}[k^*, x^*, m_1^*, \rho^*]$. Since the circuits maintain the same functionality, the claim follows by iO.

Formally, by the definition of Setup_1 ,

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ = \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ \rho^* \leftarrow F_k(x^* \| m_1^*) \\ k^* \leftarrow \text{F.Punc}(k, (x^* \| m_1^*)) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(1^{\lambda_{\text{iO}}}, 1^s, \text{Chal}[k^*, x^*, m_1^*, \rho^*]) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\widehat{\text{Chal}}) \end{array} \right].$$

Note that since the iO is T -secure, where $T(\lambda_{\text{iO}}) \geq 2^{(\lambda_{\text{iO}})^c}$, and $\lambda_{\text{iO}} = (2(\lambda + n + \ell_1))^{\frac{1}{c}}$, we get that the iO is $2^{2(\lambda+n+\ell_1)}$ -secure. Since the event above is checkable in time $2^{O(n)}$ using the original protocol's verifier,

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ \rho^* \leftarrow F_k(x^* \| m_1^*) \\ k^* \leftarrow \text{F.Punc}(k, (x^* \| m_1^*)) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(1^{\lambda_{\text{iO}}}, 1^s, \text{Chal}[k^*, x^*, m_1^*, \rho^*]) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\widehat{\text{Chal}}) \end{array} \right] \\ \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \middle| \begin{array}{c} k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(1^{\lambda_{\text{iO}}}, 1^s, \text{Chal}[k]) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\widehat{\text{Chal}}) \end{array} \right] - 2^{-2(\lambda+n+\ell_1)}.$$

By the definition of **Setup**,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}}) \\ \widehat{\text{Chal}} \leftarrow \text{iO}(1^{\lambda_{\text{iO}}}, 1^s, \text{Chal}[k]) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\widehat{\text{Chal}}) \end{array} \right] \\ &= \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right]. \end{aligned}$$

Overall, by combining the above equations, we get that,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - 2^{-2(\lambda+n+\ell_1)}, \end{aligned}$$

as required. $\square$

Hybrid 2.

- **Setup**₂[x^*, m_1^*, ρ^*]($1^\lambda, \mathbf{V}$):
 1. Run $k \leftarrow \text{F.Gen}(1^{\lambda_{\text{PRF}}}, 1^{\ell_{\text{in}}}, 1^{\ell_{\text{out}}})$.
 2. ~~Let $\rho^* \leftarrow F_k(x^* \| m_1^*)$.~~
 3. Puncture $k^* \leftarrow \text{F.Punc}(k, (x^* \| m_1^*))$.
 4. Compute $\widehat{\text{Chal}}[k^*, x^*, m_1^*, \rho^*]$.
 5. Output $\text{crs} := (\widehat{\text{Chal}}[k^*, x^*, m_1^*, \rho^*])$.

Chal[k^*, x^*, m_1^*, ρ^*](x, m_1):

1. If $(x^*, m_1^*) = (x, m_1)$,
 - (a) Output ρ^* .
2. Else
 - (a) Output $\rho \leftarrow F_{k^*}(x \| m_1)$.

Claim 9.5. Fix some $x^* \in \{0, 1\}^n$ and $m_1^* \in \{0, 1\}^{\ell_1}$. It holds that

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \rho^* \leftarrow \{0, 1\}^r \\ \text{crs} \leftarrow \text{Setup}_2[x^*, m_1^*, \rho^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}_1[x^*, m_1^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - 2^{-2(\lambda+n+\ell_1)}, \end{aligned}$$

where k is the PRF key generated by the setup.

Proof. By the puncturable PRF security, and since the event above is checkable in time $2^{O(n)}$ using the original protocol's verifier. $\square$

By applying the union bound, we obtain,

$$\begin{aligned} & \Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ \wedge \\ (x, m_1, F_k(x\|m_1), m_2) \in \text{ACC} \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ &= \sum_{\substack{x' \in \{0,1\}^n \\ m'_1 \in \{0,1\}^{\ell_1}}} \Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ \wedge \\ (x, m_1, F_k(x\|m_1), m_2) \in \text{ACC} \\ \wedge \\ (x\|m_1) = (x'\|m'_1) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right]. \end{aligned}$$

As established in Equation 12,

$$\Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ \wedge \\ (x, m_1, F_k(x\|m_1), m_2) \in \text{ACC} \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] = \nu(\lambda).$$

Therefore, we conclude that,

$$\max_{\substack{x' \in \{0,1\}^n \\ m'_1 \in \{0,1\}^{\ell_1}}} \Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ \wedge \\ (x, m_1, F_k(x\|m_1), m_2) \in \text{ACC} \\ \wedge \\ (x\|m_1) = (x'\|m'_1) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \geq \frac{\nu(\lambda)}{2^{n+\ell_1}}.$$

Note that for any x', m'_1 ,

$$\begin{aligned} & \Pr \left[\begin{array}{c} x \notin \mathcal{L} \\ \wedge \\ (\mathbf{x}, \mathbf{m}_1, F_k(x\|m_1), m_2) \in \text{ACC} \\ \wedge \\ (x\|m_1) = (x'\|m'_1) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ &= \Pr \left[\begin{array}{c} \mathbf{x}' \notin \mathcal{L} \\ \wedge \\ (\mathbf{x}', \mathbf{m}'_1, F_k(x\|m_1), m_2) \in \text{ACC} \\ \wedge \\ (x\|m_1) = (x'\|m'_1) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \end{aligned}$$

Therefore,

$$\max_{\substack{x' \in \{0,1\}^n \\ m'_1 \in \{0,1\}^{\ell_1}}} \Pr \left[\begin{array}{c} x' \notin \mathcal{L} \\ \wedge \\ (x', m'_1, F_k(x\|m_1), m_2) \in \text{ACC} \\ \wedge \\ (x\|m_1) = (x'\|m'_1) \end{array} \middle| \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \geq \frac{\nu(\lambda)}{2^{n+\ell_1}}.$$

By the above, there exists some $x^* \notin \mathcal{L}$, and m_1^* such that

$$\Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \geq \frac{\nu(\lambda)}{2^{n+\ell_1}} .$$

By Claim 9.4 and Claim 9.5, we get that,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \rho^* \leftarrow \{0, 1\}^r \\ \text{crs} \leftarrow \text{Setup}_2[x^*, m_1^*, \rho^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, F_k(x^* \| m_1^*), m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \text{crs} \leftarrow \text{Setup}(1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] - 2 \cdot 2^{-2(\lambda+n+\ell_1)} . \end{aligned}$$

By the above two equations, we get that,

$$\begin{aligned} & \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \rho^* \leftarrow \{0, 1\}^r \\ \text{crs} \leftarrow \text{Setup}_2[x^*, m_1^*, \rho^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] \\ & \geq \frac{\nu(\lambda)}{2^{(n+\ell_1)}} - 2 \cdot 2^{-2(\lambda+n+\ell_1)} . \end{aligned} \tag{13}$$

Let $\tilde{\mathbf{P}}'$ be the following adversary to the original protocol.

1. Send m_1^* to $\mathbf{V}$.
2. On receiving ρ^* from the verifier,
 - (a) Generate $\text{crs} \leftarrow \text{Setup}_2[x^*, m_1^*, \rho^*](1^\lambda, \mathbf{V})$.
 - (b) Emulate $(x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs})$.
3. Output m_2 .

Note that,

$$\Pr_{\rho} [\langle \tilde{\mathbf{P}}', \mathbf{V} \rangle(x^*) = 1] \geq \Pr \left[\begin{array}{c} (x^*, m_1^*, \rho^*, m_2) \in \text{ACC} \\ \wedge \\ (x \| m_1) = (x^* \| m_1^*) \end{array} \mid \begin{array}{c} \rho^* \leftarrow \{0, 1\}^r \\ \text{crs} \leftarrow \text{Setup}_2[x^*, m_1^*, \rho^*](1^\lambda, \mathbf{V}) \\ (x, m_1, m_2) \leftarrow \tilde{\mathbf{P}}(\text{crs}) \end{array} \right] .$$

Therefore, by Equation 13 and by the soundness of the original protocol, we get that,

$$\epsilon \geq \Pr_{\rho} [\langle \tilde{\mathbf{P}}', \mathbf{V} \rangle(x^*) = 1] \geq \frac{\nu(\lambda)}{2^{(n+\ell_1)}} - 2 \cdot 2^{-2(\lambda+n+\ell_1)} .$$

By the above, we get that,

$$\nu(\lambda) \leq 2^{(n+\ell_1)} \cdot \left(\epsilon + 2 \cdot 2^{-2(\lambda+n+\ell_1)} \right) = 2^{(n+\ell_1)} \cdot \epsilon + \text{negl}(\lambda) . ,$$

As contradiction to $\nu(\lambda) \geq 2^{(n+\ell_1)} \cdot \epsilon + 1/\text{poly}(\lambda)$. □

Acknowledgments

Shany Ben-David is supported by the European Research Union (ERC, CRYPTOPROOF, 101164375), and by the Clore Israel Foundation. Eylon Yogev is supported by the Israel Science Foundation (Grant No. 2302/22) and the European Research Union (ERC, CRYPTOPROOF, 101164375). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Part of the work was done when Agni Datta was an intern at CISA Helmholtz Center for Security and Privacy.

References

- [ACFY24] GAL ARNON, ALESSANDRO CHIESA, GIACOMO FENZI, and EYLON YOGEV. “STIR: Reed–Solomon Proximity Testing with Fewer Queries”. In: *Proceedings of the 44th Annual International Cryptology Conference*. CRYPTO ’24. 2024, pp. 380–413 (cit. on p. 1).
- [ACFY25] GAL ARNON, ALESSANDRO CHIESA, GIACOMO FENZI, and EYLON YOGEV. “WHIR: Reed–Solomon Proximity Testing with Super-Fast Verification”. In: *Proceedings of the 44th Annual International Conference on Theory and Application of Cryptographic Techniques*. EURO-CRYPT ’25. 2025, pp. 214–243 (cit. on p. 1).
- [ACY23] GAL ARNON, ALESSANDRO CHIESA, and EYLON YOGEV. “IOPs with Inverse Polynomial Soundness Error”. In: *Proceedings of the 64th Annual IEEE Symposium on Foundations of Computer Science*. FOCS ’23. 2023, pp. 752–761 (cit. on p. 1).
- [AHIV17] SCOTT AMES, CARMIT HAZAY, YUVAL ISHAI, and MUTHURAMAKRISHNAN VENKITASUBRAMANIAM. “Ligero: Lightweight Sublinear Arguments Without a Trusted Setup”. In: *Proceedings of the 24th ACM Conference on Computer and Communications Security*. CCS ’17. 2017, pp. 2087–2104 (cit. on p. 1).
- [AR21] GAL ARNON and GUY N. ROTHBLUM. “On Prover-Efficient Public-Coin Emulation of Interactive Proofs”. In: *2nd Conference on Information-Theoretic Cryptography, ITC 2021, Virtual Conference, July 23-26, 2021*. Ed. by STEFANO TESSARO. Vol. 199. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 3:1–3:15 (cit. on p. 2).
- [AY25] GAL ARNON and EYLON YOGEV. “Towards a White-Box Secure Fiat-Shamir Transformation”. In: *Proceedings of the 45th Annual International Cryptology Conference*. CRYPTO ’25. 2025, pp. 27–56 (cit. on p. 1).
- [Bar01] BOAZ BARAK. “How to Go Beyond the Black-Box Simulation Barrier”. In: *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*. FOCS ’11. 2001, pp. 106–115 (cit. on p. 1).
- [BBBF18] DAN BONEH, JOSEPH BONNEAU, BENEDIKT BÜNZ, and BEN FISCH. “Verifiable Delay Functions”. In: *Advances in Cryptology – CRYPTO 2018, Part I*. Ed. by HOVAV SHACHAM and ALEXANDRA BOLDYREVA. Vol. 10991. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Cham, Switzerland, 2018, pp. 757–788. DOI: 10.1007/978-3-319-96884-1_25 (cit. on p. 1).
- [BBHR18] ELI BEN-SASSON, IDDO BENTOV, YINON HORESH, and MICHAEL RIABZEV. “Fast Reed–Solomon Interactive Oracle Proofs of Proximity”. In: *Proceedings of the 45th International Colloquium on Automata, Languages and Programming*. ICALP ’18. 2018, 14:1–14:17 (cit. on p. 1).

- [BCKP14] NIR BITANSKY, RAN CANETTI, Yael TAUMAN KALAI, and OMER PANETH. “On Virtual Grey Box Obfuscation for General Circuits”. In: *Advances in Cryptology – CRYPTO 2014, Part II*. Ed. by JUAN A. GARAY and ROSARIO GENARO. Vol. 8617. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer Berlin Heidelberg, Germany, 2014, pp. 108–125. DOI: 10.1007/978-3-662-44381-1_7 (cit. on pp. 1, 21).
- [BCS16] ELI BEN-SASSON, ALESSANDRO CHIESA, and NICHOLAS SPOONER. “Interactive Oracle Proofs”. In: *Proceedings of the 14th Theory of Cryptography Conference*. TCC ’16-B. 2016, pp. 31–60 (cit. on p. 1).
- [BGI+01] BOAZ BARAK, ODED GOLDBREICH, RUSSELL IMPAGLIAZZO, STEVEN RUDICH, AMIT SAHAI, SALIL P. VADHAN, and KE YANG. “On the (Im)possibility of Obfuscating Programs”. In: *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*. Ed. by JOE KILIAN. Vol. 2139. Lecture Notes in Computer Science. Springer, 2001, pp. 1–18 (cit. on p. 20).
- [BKM20] ZVIKA BRAKERSKI, VENKATA KOPPULA, and TAMER MOUR. “NIZK from LPN and Trapdoor Hash via Correlation Intractability for Approximable Relations”. In: *Proceedings of the 40th Annual International Cryptology Conference*. CRYPTO ’20. 2020, pp. 738–767 (cit. on p. 1).
- [BKP19] NIR BITANSKY, DAKSHITA KHURANA, and OMER PANETH. “Weak zero-knowledge beyond the black-box barrier”. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. 2019, pp. 1091–1102 (cit. on pp. 3, 17).
- [BPW16] NIR BITANSKY, OMER PANETH, and DANIEL WICHS. “Perfect Structure on the Edge of Chaos - Trapdoor Permutations from Indistinguishability Obfuscation”. 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. 474–502 (cit. on p. 53).
- [BR93] MIHIR BELLARE and PHILLIP ROGAWAY. “Random Oracles are Practical: A Paradigm for Designing Efficient Protocols”. In: *ACM CCS 93: 1st Conference on Computer and Communications Security*. Ed. by DOROTHY E. DENNING, RAYMOND PYLE, RAVI GANESAN, RAVI S. SANDHU, and VICTORIA ASHBY. Fairfax, Virginia, USA: ACM Press, 1993, pp. 62–73. DOI: 10.1145/168588.168596 (cit. on p. 1).
- [BY25] SHANY BEN-DAVID and EYLON YOGEV. “Incompressible Encryption with Everlasting Security”. In: *Theory of Cryptography - 23rd International Conference, TCC 2025, Aarhus, Denmark, December 1-5, 2025, Proceedings, Part II*. Ed. by BENNY APPLEBAUM and HUIJIA RACHEL LIN. Vol. 16269. Lecture Notes in Computer Science. Springer, 2025, pp. 66–96 (cit. on p. 58).
- [CCH+19] RAN CANETTI, YILEI CHEN, JUSTIN HOLMGREN, ALEX LOMBARDI, GUY N. ROTHBLUM, RON D. ROTHBLUM, and DANIEL WICHS. “Fiat-Shamir: from practice to theory”. In: *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*. STOC ’19. 2019, pp. 1082–1090 (cit. on p. 1).
- [CCR16] RAN CANETTI, YILEI CHEN, and LEONID REYZIN. “On the Correlation Intractability of Obfuscated Pseudorandom Functions”. In: *Proceedings of the 13th Theory of Cryptography Conference*. TCC ’16. 2016, pp. 389–415 (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 (cit. on pp. 1, 5).

- [CGH98] RAN CANETTI, ODED GOLDREICH, and SHAI HALEVI. “The Random Oracle Methodology, Revisited (Preliminary Version)”. In: *30th Annual ACM Symposium on Theory of Computing*. Dallas, TX, USA: ACM Press, 1998, pp. 209–218. DOI: 10.1145/276698.276741 (cit. on p. 1).
- [CJJ21] ARKA RAI CHOUDHURI, ABHISHEK JAIN, and ZHENGZHONG JIN. “Non-interactive Batch Arguments for NP from Standard Assumptions”. In: *Proceedings of the 41st Annual International Cryptology Conference*. CRYPTO ’21. 2021, pp. 394–423 (cit. on p. 1).
- [CLW18] RAN CANETTI, ALEX LOMBARDI, and DANIEL WICHS. *Fiat-Shamir: From Practice to Theory, Part II (NIZK and Correlation Intractability from Circular-Secure FHE)*. Cryptology ePrint Archive, Paper 2018/1248. 2018. URL: <https://eprint.iacr.org/2018/1248> (cit. on pp. 37, 58).
- [CY21a] ALESSANDRO CHIESA and EYLON YOGEV. “Subquadratic SNARGs in the Random Oracle Model”. In: *Proceedings of the 41st Annual International Cryptology Conference*. CRYPTO ’21. 2021, pp. 711–741 (cit. on p. 1).
- [CY21b] ALESSANDRO CHIESA and EYLON YOGEV. “Tight Security Bounds for Micali’s SNARGs”. In: *Proceedings of the 19th Theory of Cryptography Conference*. TCC ’21. 2021, pp. 401–434 (cit. on p. 1).
- [DNRS99] CYNTHIA DWORK, MONI NAOR, OMER REINGOLD, and LARRY STOCKMEYER. “Magic functions”. In: *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*. IEEE. 1999, pp. 523–534 (cit. on pp. 3, 17).
- [FGJ18] NILS FLEISCHHACKER, VIPUL GOYAL, and ABHISHEK JAIN. “On the existence of three round zero-knowledge proofs”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2018, pp. 3–33 (cit. on pp. 3, 6, 14, 18, 49).
- [FS87] AMOS FIAT and ADI SHAMIR. “How to Prove Yourself: Practical Solutions to Identification and Signature Problems”. In: *Advances in Cryptology – CRYPTO’86*. Ed. by ANDREW M. ODLYZKO. Vol. 263. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer Berlin Heidelberg, Germany, Aug. 1987, pp. 186–194. DOI: 10.1007/3-540-47721-7_12 (cit. on p. 1).
- [GGM19] ODED GOLDREICH, SHAFI GOLDWASSER, and SILVIO MICALI. “How to construct random functions”. In: *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. Ed. by ODED GOLDREICH. ACM, 2019, pp. 241–264 (cit. on p. 19).
- [GGM86] ODED GOLDREICH, SHAFI GOLDWASSER, and SILVIO MICALI. “How to Construct Random Functions”. In: *Journal of the ACM* 33.4 (Oct. 1986), pp. 792–807. DOI: 10.1145/6490.6503 (cit. on p. 20).
- [GK03] SHAFI GOLDWASSER and Yael TAUMAN KALAI. “On the (In)security of the Fiat-Shamir Paradigm”. In: *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*. FOCS ’03. 2003, pp. 102–113 (cit. on p. 1).
- [GMR89] SHAFI GOLDWASSER, SILVIO MICALI, and CHARLES RACKOFF. “The knowledge complexity of interactive proof systems”. In: *SIAM Journal on Computing* 18.1 (1989). Preliminary version appeared in STOC ’85., pp. 186–208 (cit. on p. 21).
- [GMW86] ODED GOLDREICH, SILVIO MICALI, and AVI WIGDERSON. “Proofs that yield nothing but their validity and a methodology of cryptographic protocol design”. In: *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*. 1986, pp. 174–187. DOI: 10.1109/SFCS.1986.47 (cit. on p. 1).
- [GO94] ODED GOLDREICH and YAIR OREN. “Definitions and properties of zero-knowledge proof systems”. In: *Journal of Cryptology* 7.1 (1994), pp. 1–32 (cit. on pp. 3, 18, 49).

- [GS86] SHAFI GOLDWASSER and MICHAEL SIPSER. “Private Coins versus Public Coins in Interactive Proof Systems”. In: *18th Annual ACM Symposium on Theory of Computing*. Berkeley, CA, USA: ACM Press, 1986, pp. 59–68. DOI: 10.1145/12130.12137 (cit. on p. 1).
- [HILL99] JOHAN HÅSTAD, RUSSELL IMPAGLIAZZO, LEONID A. LEVIN, and MICHAEL LUBY. “A Pseudo-random Generator from any One-way Function”. In: *SIAM J. Comput.* 28.4 (1999), pp. 1364–1396 (cit. on p. 20).
- [HJKS22] JAMES HULETT, RUTA JAWALE, DAKSHITA KHURANA, and AKSHAYARAM SRINIVASAN. “SNARGs for P from Sub-exponential DDH and QR”. In: *Proceedings of the 41st Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT ’22. 2022, pp. 520–549 (cit. on p. 1).
- [HL18] JUSTIN HOLMGREN and ALEX LOMBARDI. “Cryptographic Hashing from Strong One-Way Functions (Or: One-Way Product Functions and Their Applications)”. In: *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science*. FOCS ’18. 2018, pp. 850–858 (cit. on p. 1).
- [HLR21] JUSTIN HOLMGREN, ALEX LOMBARDI, and RON D. ROTHBLUM. “Fiat-Shamir via list-recoverable codes (or: parallel repetition of GMW is not zero-knowledge)”. In: *Proceedings of the 53rd Annual ACM Symposium on Theory of Computing*. STOC ’21. 2021, pp. 750–760 (cit. on p. 1).
- [HT98] SATOSHI HADA and TOSHIKI TANAKA. “On the Existence of 3-Round Zero-Knowledge Protocols”. In: *Proceedings of the 18th Annual International Cryptology Conference*. CRYPTO ’98. 1998, pp. 408–423 (cit. on p. 3).
- [JJ21] ABHISHEK JAIN and ZHENGZHONG JIN. “Non-interactive Zero Knowledge from Sub-exponential DDH”. In: *Proceedings of the 40th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT ’21. 2021, pp. 3–32 (cit. on p. 1).
- [KLV23] Yael TAUMAN KALAI, ALEX LOMBARDI, and VINOD VAIKUNTANATHAN. “SNARGs and PPAD Hardness from the Decisional Diffie-Hellman Assumption”. In: *Proceedings of the 42nd Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT ’23. 2023, pp. 470–498 (cit. on p. 1).
- [KMN+22] ILAN KOMARGODSKI, TAL MORAN, MONI NAOR, RAFAEL PASS, ALON ROSEN, and EYLON YOGEV. “One-Way Functions and (Im)perfect Obfuscation”. In: *SIAM J. Comput.* 51.6 (2022), pp. 1769–1795 (cit. on p. 51).
- [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 (cit. on pp. 1–3, 5, 61).
- [KRS25] DMITRY KHOVRATOVICH, RON D. ROTHBLUM, and LEV SOUKHANOV. “How to Prove False Statements: Practical Attacks on Fiat-Shamir”. In: *Proceedings of the 45th Annual International Cryptology Conference*. CRYPTO ’25. 2025, pp. 3–26 (cit. on p. 1).
- [Mic00] SILVIO MICALI. “Computationally Sound Proofs”. In: *SIAM Journal on Computing* 30.4 (2000). Preliminary version appeared in FOCS ’94., pp. 1253–1298 (cit. on p. 1).
- [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*. CRYPTO ’19. 2019, pp. 89–114 (cit. on p. 1).
- [PS96] DAVID POINTCHEVAL and JACQUES STERN. “Security Proofs for Signature Schemes”. In: *Proceedings of the 14th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT ’96. 1996, pp. 387–398 (cit. on p. 1).

- [Sch91] CLAUS-PETER SCHNORR. “Efficient Signature Generation by Smart Cards”. In: *Journal of Cryptology* 4.3 (Jan. 1991), pp. 161–174. DOI: 10.1007/BF00196725 (cit. on p. 1).
- [SV00] AMIT SAHAI and SALIL VADHAN. *A Complete Problem for Statistical Zero Knowledge*. Cryptology ePrint Archive, Report 2000/056. 2000. URL: <https://eprint.iacr.org/2000/056> (cit. on pp. 1, 51).
- [SV03] AMIT SAHAI and SALIL P. VADHAN. “A complete problem for statistical zero knowledge”. In: *J. ACM* 50.2 (2003), pp. 196–249 (cit. on pp. 51, 58).
- [SW14] AMIT SAHAI and BRENT WATERS. “How to use indistinguishability obfuscation: deniable encryption, and more”. In: *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*. Ed. by DAVID B. SHMOYS. ACM, 2014, pp. 475–484 (cit. on p. 19).
- [ZCF24] HADAS ZEILBERGER, BINYI CHEN, and BEN FISCH. “BaseFold: Efficient Field-Agnostic Polynomial Commitment Schemes from Foldable Codes”. In: *Proceedings of the 44th Annual International Cryptology Conference*. CRYPTO ’24. 2024, pp. 138–169 (cit. on p. 1).