

Non-Interactive Black-Box Witness-Indistinguishable Commit-and-Prove via Targeted Hitting-Set Generators*

Susumu Kiyoshima

NTT Social Informatics Laboratories
susumu.kiyoshima@ntt.com

September 24, 2026

Abstract

Targeted hitting-set generators (HSGs) are primitives that have recently attracted attention in the study of pseudorandomness and derandomization. Recent works have shown that sufficiently strong targeted HSGs can be used to soundly instantiate the Fiat–Shamir transformation, yielding non-interactive protocols against uniform PPT adversaries in the plain model (i.e., without any setup phase for selecting a cryptographic hash function). In particular, Chen et al. (STOC 2025) combined this paradigm with the MPC-in-the-head technique to construct a non-interactive witness-indistinguishable argument for NP from a non-interactive commitment scheme and a targeted HSG.

Inspired by these results, we explore other cryptographic applications of this paradigm. Our main result is a black-box construction of a non-interactive witness-indistinguishable commit-and-prove scheme against uniform PPT adversaries in the plain model. Following Chen et al., we obtain our construction by combining the MPC-in-the-head technique with the targeted-HSG-based Fiat–Shamir paradigm. The construction relies on an injective one-way function with efficiently recognizable range and a sufficiently strong targeted HSG. The latter is the same targeted HSG used by Chen et al. and can be instantiated under a very strong but plausible complexity-theoretic assumption.

*© IACR 2026. This article is the full version of the article published by Springer-Verlag in the proceedings of TCC 2026, available at <DOI>.

Contents

1	Introduction	3
1.1	Background	3
1.2	Our Contribution	5
1.3	Additional Related Work	5
2	Technical Overview	6
2.1	Interactive Black-Box WI Commit-and-Prove	6
2.2	Non-Interactive Black-Box WI Commit-and-Prove	10
3	Preliminaries	10
3.1	Notations and Conventions	10
3.2	Non-interactive Commitment	11
3.3	Non-Interactive Witness-Indistinguishable Commit and Prove	11
3.4	Shamir's Secret Sharing	12
3.5	Targeted Hitting-Set Generator	13
3.6	Secure Computation	14
4	Non-Interactive Witness-Indistinguishable Commit and Prove	15
4.1	Step 1. Interactive SHVZK Commit and Prove	15
4.2	Step 2. Interactive WI Commit and Prove	22
4.3	Step 3. Non-interactive WI Commit and Prove	30

1 Introduction

1.1 Background

1.1.1 NW-Type PRGs/HSGs

The pseudorandom generators (PRGs) of Nisan and Wigderson [NW94] and follow-up work—hereafter called *NW-type PRGs*—play a central role in *derandomization* (i.e., deterministic simulation of randomized algorithms). Unlike cryptographic PRGs, NW-type PRGs may run longer than the distinguishers, and they are often only required to fool restricted classes of distinguishers with larger error.¹ This relaxation enables substantially shorter seed lengths than cryptographic PRGs; for instance, under strong (but plausible) complexity assumptions,² there exists a PRG with seed length $O(\log N)$, error $1/\text{poly}(N)$, and running time $\text{poly}(N)$ that fools size- $O(N)$ circuits [IW97]. (Here N is the input length of the distinguisher.)

An NW-type PRG can be used to derandomize a randomized algorithm A on input x as follows: enumerate the PRG outputs for all seeds, run $A(x)$ on each pseudorandom string in the list, and take a majority vote. This derandomization works as long as the PRG is strong enough to fool $A(x; \cdot)$ on every input x . If the PRG has seed length $O(\log N)$, then enumerating all seeds takes $\text{poly}(N)$ time, so the running-time overhead is polynomially bounded.

For derandomizing one-sided error algorithms, a weaker primitive called a *hitting-set generator* (HSG) suffices. An HSG guarantees that, for any x , if $A(x)$ outputs 1 with sufficiently high probability over true randomness, then at least one pseudorandom string in the list makes $A(x)$ output 1. This guarantee is enough because, for the one-sided derandomization, the majority vote above can be replaced by checking whether any string in the list makes $A(x)$ output 1.

In what follows, we always use the term PRG to refer to an NW-type PRG. We also use the generator in its derandomization form. That is, rather than treating it as taking a seed and outputting a single pseudorandom string (e.g., $r \leftarrow G(\sigma)$), we view it as outputting the list of pseudorandom strings indexed by all seeds (e.g., $\{r_\sigma\}_{\sigma \in \{0,1\}^\ell} \leftarrow G(1^\ell)$). Note that the list size $L := 2^\ell$ must be roughly as large as the distinguisher size (or, for a uniform distinguisher, the distinguisher input length); otherwise, the list could be hardwired into the distinguisher, making pseudorandomness impossible.

1.1.2 Targeted PRGs/HSGs and Their Application to Fiat–Shamir

Recent progress has achieved even shorter seed lengths (and therefore smaller list sizes) by considering *targeted PRGs/HSGs* [Gol11, CT22a]. A targeted PRG/HSG is a non-black-box variant of a standard PRG/HSG whose output may be tailored to the particular distinguisher being fooled. That is, when fooling a randomized uniform algorithm A with input length N , the generator G no longer needs to produce a single list of pseudorandom strings that fools every algorithm in $\{A(x; \cdot)\}_{x \in \{0,1\}^N}$; instead, it suffices that for each input $x \in \{0,1\}^N$, $G(x)$ produces a list that fools $A(x; \cdot)$.

To obtain shorter seed lengths (and therefore smaller list sizes), Chen and Tell [CT22a] consider an average-case formulation of targeted PRGs/HSGs. In this formulation, the input x is drawn from an arbitrary polynomial-time samplable distribution, and $G(x)$ is required to fool the distinguisher with high probability over the choice of x . They then construct, under a strong and non-standard complexity assumption, a targeted HSG with list size N^ϵ for any constant $\epsilon > 0$. This result yields a very small overhead for derandomizing uniform algorithms on polynomial-time samplable inputs: a randomized algorithm running in time T can be deterministically simulated in roughly $N^\epsilon T$ time. (For more details on recent progress on targeted PRGs/HSGs, see the survey by Chen and Tell [CT23a] and Tell’s invited talk at TCC 2025 [Tel25].)

¹Recall that a cryptographic PRG has a fixed polynomial-time running time and is required to be pseudorandom against any PPT distinguisher with negligible error (i.e., with at most negligible distinguishing advantage).

²Specifically, assume that a problem in $E = \text{DTIME}(2^{O(n)})$ has circuit complexity $2^{\Omega(n)}$.

As an application, Chen and Tell [CT23b] and Chen et al. [CRT25] use a targeted HSG in place of a cryptographic hash function to soundly instantiate the *Fiat–Shamir transformation* [FS87] for certain public-coin interactive protocols. That is, they consider a transformation where verifier challenge messages are deterministically generated by feeding the current transcript into a targeted HSG, and they derive the soundness of the transformed protocol from pseudorandomness of the targeted HSG.³ In particular, Chen et al. [CRT25] show (among other results) that a non-interactive witness-indistinguishable argument (NIWI) can be constructed from a targeted HSG and a non-interactive commitment scheme. Compared with NIWIs obtained by soundly instantiating the Fiat–Shamir transformation with cryptographic hash functions (such as *correlation-intractable hash functions* [CGH04]), the resulting NIWI works in the plain model—i.e., there is no need to sample a hash function as a common reference string (CRS). The tradeoff is that the soundness of the NIWI is guaranteed only against uniform PPT adversaries; this is because the targeted HSG is only guaranteed to be pseudorandom against uniform adversaries on polynomial-time samplable inputs.

1.1.3 Targeted PRGs/HSGs and MPC in the Head

From a cryptographer’s perspective, the result by Chen et al. [CRT25] is obtained in an intriguing way: they use a targeted HSG to soundly combine the Fiat–Shamir transformation with the *MPC-in-the-head technique* [IKOS09].⁴ Recall that the MPC-in-the-head technique was originally proposed to realize zero-knowledge (ZK) protocols from secure multi-party computation (MPC): given a public statement $x \in \mathbf{L}$ and a secret witness w , the prover proves the relation $\mathbf{R}(x, w) = 1$ to the verifier as follows.

1. The prover generates a secret sharing $(w_1, \dots, w_\nu)$ of w such that $w_1 \oplus \dots \oplus w_\nu = w$, and runs in its head an MPC protocol that computes the functionality $\mathcal{F}(x, w_1, \dots, w_\nu) := \mathbf{R}(x, w_1 \oplus \dots \oplus w_\nu)$. The prover commits to each party’s view $\text{view}_1, \dots, \text{view}_\nu$ obtained from this MPC execution.
2. The verifier selects a random subset of the parties $Q \subset [\nu]$.
3. The prover opens the views $\{\text{view}_i\}_{i \in Q}$ of the parties selected by Q , and the verifier checks whether the opened views are consistent (e.g., whether each party outputs 1 and whether the message sent by Party i matches the message received by Party j for every $i, j \in Q$).

Soundness holds since when $\mathbf{R}(x, w) = 0$ for every w , the security of the MPC protocol implies that there are many inconsistencies among the views (and therefore when Q is large enough, inconsistent views are selected by Q with high probability). Zero-knowledge holds since when Q is not too large, opening $\{\text{view}_i\}_{i \in Q}$ does not leak any information about the unopened shares $\{w_i\}_{i \notin Q}$.

Chen et al. [CRT25] apply the Fiat–Shamir transformation to the MPC-in-the-head technique using a targeted HSG as follows. The transcript up to the first message (consisting of the statement x and the commitments $c_1, \dots, c_\nu$ to the views) is fed into the targeted HSG to deterministically generate a list of challenges $\{Q^{(\alpha)}\}_{\alpha \in [L]}$, and the prover opens the views selected by $Q^{(1)} \cup \dots \cup Q^{(L)}$. For soundness, it is shown that one of the challenges in the list selects inconsistent views when the targeted HSG is pseudorandom for a sufficiently strong class of distinguishers.⁵ For WI, it is shown that, with appropriate MPC parameters, the witness w remains hidden even when views are opened based on a list of challenges rather than a single

³As observed by Chen and Tell [CT23b], this transformation can be viewed as derandomizing the verifier of a public-coin interactive protocol via a targeted HSG.

⁴More precisely, Chen et al. [CRT25] combine the Fiat–Shamir transformation with the *zero-knowledge probabilistically checkable proof* (ZK-PCP) of Hazay et al. [HVV22], and Hazay et al. obtain their ZK-PCP using the MPC-in-the-head technique.

⁵Roughly speaking, it suffices that the targeted HSG is pseudorandom for the algorithm $A^{\text{SAT}}(x, c_1, \dots, c_\nu; Q)$ that uses its SAT oracle to recover the values committed in $c_1, \dots, c_\nu$ and checks whether the challenge Q selects inconsistent views.

challenge. (Note that using a *targeted* HSG is important here. If a standard NW-type HSG is used, the list size is larger than the input length, so we would have $L \geq |c_1| + \dots + |c_\nu| \geq \nu$. Thus, the union of the challenges $Q^{(1)} \cup \dots \cup Q^{(L)}$ could select all parties, thereby leaking w to the verifier.)

1.2 Our Contribution

Our starting point is the question of whether we can build other cryptographic protocols by using a targeted HSG to soundly combine the Fiat–Shamir transformation with the MPC-in-the-head technique. In this paper, we answer this question affirmatively and construct a non-interactive black-box witness-indistinguishable (WI) *commit-and-prove* scheme.

Commit-and-prove. A commit-and-prove scheme is a commitment scheme equipped with a “prove phase,” where the sender can prove a statement about the committed value without opening the commitment. Such schemes were implicitly used by Goldreich et al. [GMW87] for obtaining a secure multi-party computation protocol with malicious security, and a formalization is given by Canetti et al. [CLOS02]. (Essentially, in this application, commit-and-prove schemes are used to bind a party to a hidden input and later enforce that its protocol messages are consistent with that input.) A construction of commit-and-prove schemes is called *black-box* if it uses the underlying cryptographic primitive only via its input/output interface. Such commit-and-prove schemes have been used as a tool for enforcing honest behavior on malicious parties without relying on non-black-box uses of the underlying cryptographic primitives, thereby avoiding costs that depend on the circuit complexity of their concrete implementations (see, e.g., [GLOV12, LP12, GOSV14]). Khurana et al. [KOS18] constructed, among other results, a 3-round black-box WI commit-and-prove scheme (i.e., the commit and the prove phases consist of three rounds in total).

Our result. Our result is a non-interactive black-box WI commit-and-prove scheme (i.e., both the commit phase and the prove phase consist of a single message from the sender).

Theorem (informal; see Theorem 3). *Assume the existence of an injective one-way function with efficiently recognizable range and a targeted HSG for all distinguishers in $\text{DTIME}[N]^{\text{SAT}}$. Then, there exists a black-box construction of a non-interactive WI commit-and-prove scheme against uniform PPT adversaries in the plain model.*

The targeted HSG used in this theorem is the same as the one used by Chen et al. [CRT25] in their application to the Fiat–Shamir transformation and the MPC-in-the-head technique. Therefore, it can be instantiated under the same (very strong) complexity-theoretic assumption (see Theorem 2 and Corollary 1 for details).

Our result is incomparable to the 3-round black-box WI commit-and-prove scheme of Khurana et al. [KOS18]. Specifically, our result is non-interactive but requires a strong targeted HSG as an additional assumption and provides security only against uniform PPT adversaries. Also, while Khurana et al. achieve a proof-of-knowledge-style soundness definition, we do not guarantee the existence of an efficient extractor. (This relaxation seems reasonable given that the prove phase of our scheme is non-interactive.)

1.3 Additional Related Work

Khurana [Khu21] constructed a relaxed non-interactive commit-and-prove argument in the plain model via non-black-box techniques, assuming sub-exponentially secure indistinguishability obfuscation and one-way functions. Khurana’s result differs from ours in two respects: its construction is non-black-box, and its notion of non-interactive commit-and-prove differs

from ours in how the commitment string is determined. Specifically, under Khurana’s definition, after receiving a single message from the committer, the receiver uses private randomness to produce a local output that is treated as the commitment string. Thus, under this definition, the same committer message may induce different commitment strings depending on the receiver’s private randomness. In our notion, by contrast, the commitment string is generated by the sender and is fixed by the sender’s commit-phase message, independently of any receiver randomness.

2 Technical Overview

We obtain a non-interactive black-box WI commit-and-prove scheme in two steps.

1. Using the MPC-in-the-head technique, we obtain an interactive black-box WI commit-and-prove scheme with constant soundness error.
2. We apply the Fiat–Shamir transformation, instantiated with a targeted HSG, to obtain a non-interactive scheme with negligible soundness error.

Let us first give an informal definition of a commit-and-prove scheme. A commit-and-prove scheme extends a standard statistically binding commitment scheme by adding a prove phase. The prove phase is required to satisfy the following security properties.

- **Soundness.** If the sender provides a convincing proof for a commitment c and a statement C (represented as a Boolean circuit), then there exists an opening that opens the commitment c to a message m such that $C(m) = 1$.
- **WI.** For any messages m_0, m_1 and statement C with $C(m_0) = C(m_1) = 1$, the commit and prove phases executed with respect to m_0 and C are computationally indistinguishable from those executed with respect to m_1 and C .

With this definition in mind, we first outline the interactive construction and then the transformation to a non-interactive construction.

2.1 Interactive Black-Box WI Commit-and-Prove

For ease of presentation, we first present a non-black-box construction of an interactive WI commit-and-prove scheme and then transform it into a black-box construction.

2.1.1 Non-Black-Box Construction

The starting point is the following naive (and insecure) construction, which combines any perfectly binding, computationally hiding non-interactive commitment scheme (NIC) with a WI proof system.

Construction 1.
• Commit phase. To commit to a message m, the sender computes a commitment c to m using the NIC. • Open phase. To open a commitment to m, the sender opens c to m. • Prove phase. To prove a statement C, the sender uses the WI proof system to prove the following statement: c is a commitment to a message m such that $C(m) = 1$.

Soundness is immediate from the soundness of the WI proof system. However, this construction is not WI. Indeed, by the binding of the NIC, the statement proven by the WI proof is true for only a single m , and in this case the WI proof provides no privacy about m .

As a modification of the above construction, consider using the NIC twice in parallel.

Construction 2.

- **Commit phase.** To commit to a message m , the sender computes commitments c_0, c_1 to m using the NIC twice.
- **Open phase.** To open a commitment to m , the sender opens c_0 and c_1 to m .
- **Prove phase.** To prove a statement C , the sender uses the WI proof system to prove the following statement: there exists $b \in \{0, 1\}$ such that c_b is a commitment to a message m with $C(m) = 1$.

Now WI is immediate from the hiding of the NIC and the WI of the WI proof system. However, this construction does not satisfy soundness. Indeed, the sender can produce a convincing WI proof even if one of c_0 or c_1 is a commitment to a message m' such that $C(m') \neq 1$. (If the WI proof is modified to prove the statement for both c_0 and c_1 , then WI no longer holds for the same reason as in Construction 1.)

Our first non-trivial observation is that we can obtain a secure WI commit-and-prove scheme by using the NIC four times in parallel (with carefully designed open and prove phases).

Construction 3.

- **Commit phase.** To commit to a message m , the sender computes commitments c_0, c_1, c_2, c_3 to m using the NIC four times.
- **Open phase.** To open a commitment to m , the sender sends m to the receiver and also opens c_0, c_1, c_2, c_3 . Let m_0, m_1, m_2, m_3 denote the opened messages. Then, the receiver checks whether the most frequent value among m_0, m_1, m_2, m_3 is equal to m , with ties broken canonically. (We always have $m_0 = m_1 = m_2 = m_3 = m$ in an honest execution, but we intentionally relax the receiver's verification condition so that even if $m_i \neq m$ for some $i \in \{0, 1, 2, 3\}$, the receiver accepts as long as the most frequent value equals m .)^a
- **Prove phase.** To prove a statement C , the sender uses the WI proof system to prove the following statement: there exist messages $m^\dagger, m^\ddagger$ and an index $i^* \in \{0, 1, 2, 3\}$ such that for every $i \in \{0, 1, 2, 3\} \setminus \{i^*\}$, c_i is a commitment to a message m_i with $m_i \in \{m^\dagger, m^\ddagger\} \wedge C(m_i) = 1$.

^aSimilar relaxations of full consistency verification have appeared in the literature on black-box constructions; see, e.g., [CDMW18, CDMW09].

WI is easy to prove from the hiding of the NIC and the WI property of the WI proof. The key points are (i) there is a commitment c_{i^*} that is not used in the WI proof and (ii) each of the remaining three commitments is only required to be a commitment to $m_i \in \{m^\dagger, m^\ddagger\}$ such that $C(m_i) = 1$. We can in particular use a simple hybrid argument that switches the committed value of c_{i^*} and the index i^* one by one. Soundness is proved as follows. From the soundness of the WI proof, (i) there exists at most one index $i^* \in \{0, 1, 2, 3\}$ such that c_{i^*} is a commitment to some m' such that $C(m') \neq 1$, and moreover (ii) there exist two distinct indices $i, i' \in \{0, 1, 2, 3\} \setminus \{i^*\}$ and a message m such that both c_i and $c_{i'}$ are commitments to m with $C(m) = 1$. From the latter, the most frequent value m_k is committed by at least two commitments, and together with the former this implies $C(m_k) = 1$. Therefore, when the sender provides a convincing proof, we can open c to some m such that $C(m) = 1$ by opening c_0, c_1, c_2, c_3 to their committed values.⁶

⁶This argument assumes that we can correctly open c_0, c_1, c_2, c_3 to some m_0, m_1, m_2, m_3 even if these commit-

2.1.2 Black-Box Construction

We now gradually modify the above non-black-box construction to obtain a black-box construction. In particular, we modify Construction 3 so that the statement proven by the WI proof becomes an “MPC-in-the-head friendly” form.

First, consider the following construction, where we eliminate $m^\dagger, m^\ddagger$ from the witness of the WI proof. (This is achieved by adding interaction and borrowing the technique used by Khurana et al. [KOS18].)

Construction 4.

- **Commit phase.** To commit to a message m , the sender views m as an element of a finite field $\mathbb{F}$, samples a random element $r \in \mathbb{F}$, and computes commitments c_0, c_1, c_2, c_3 to (m, r) using the NIC.
- **Open phase.** To open a commitment to m , the sender sends m to the receiver and also opens c_0, c_1, c_2, c_3 . Let $(m_0, r_0), (m_1, r_1), (m_2, r_2), (m_3, r_3)$ denote the opened messages. Then, the receiver checks whether the most frequent pair (m_k, r_k) among them satisfies $m_k = m$ (with ties broken canonically).
- **Prove phase.** In a proof for a statement C , the receiver first samples a random element $u \in \mathbb{F} \setminus \{0\}$ and sends it to the sender. In response, the sender samples a random bit $\sigma \in \{0, 1\}$, sends $v_\sigma := ur + m$ and a random element $v_{1-\sigma} \in \mathbb{F}$ to the receiver, and uses the WI proof system to prove the following statement: there exists $i^* \in \{0, 1, 2, 3\}$ such that for each $i \in \{0, 1, 2, 3\} \setminus \{i^*\}$, c_i is a commitment to a pair (m_i, r_i) with $r_i u + m_i \in \{v_0, v_1\} \wedge C(m_i) = 1$.

WI can be proved via the same simple hybrid argument as in Construction 3. Soundness can be proved as in Construction 3 by using that, with high probability, the committed messages $(m_0, r_0), (m_1, r_1), (m_2, r_2), (m_3, r_3)$ of the commitments c_0, c_1, c_2, c_3 satisfy $\{m_i\}_{i \in \{0,1,2,3\} \setminus \{i^*\}} \subseteq \{m^\dagger, m^\ddagger\}$ for some i^* and $m^\dagger, m^\ddagger$. Indeed, with high probability over the choice of u , we have $r_i u + m_i \neq r_j u + m_j$ for every i, j such that $m_i \neq m_j$ (i.e., the map $m_i \mapsto r_i u + m_i$ is injective). Thus, given that the soundness of the WI proof guarantees $\{r_i u + m_i\}_{i \in \{0,1,2,3\} \setminus \{i^*\}} \subseteq \{v_0, v_1\}$ for some i^* , we have $\{m_i\}_{i \in \{0,1,2,3\} \setminus \{i^*\}} \subseteq \{m^\dagger, m^\ddagger\}$ for some $m^\dagger, m^\ddagger$.

Next, consider the following construction, where we eliminate i^* from the witness of the WI proof. (This is achieved by using a standard trick used in prior work such as ZAPs [DN07].) In the following, we instantiate the WI proof system with Blum’s 3-round ZK proof for Graph Hamiltonicity (or any special honest-verifier ZK proof) Π . For simplicity, assume that Π has a 1-bit verifier challenge message and soundness error $1/2$. Let the three messages of Π be denoted by π_1, π_2, π_3 .

Construction 5.

- **Commit phase and open phase:** Same as Construction 4.
- **Prove phase.**
 1. The receiver samples a random $u \in \mathbb{F} \setminus \{0\}$ and sends it to the sender.
 2. The sender samples a random bit $\sigma \in \{0, 1\}$, sends $v_\sigma := ur + m$ and a random element $v_{1-\sigma} \in \mathbb{F}$ to the receiver, and for each $i \in \{0, 1, 2, 3\}$ runs Π on the following statement to obtain its first message $\pi_{1,i}$: c_i is a commitment to a pair (m_i, r_i) such that $r_i u + m_i \in \{v_0, v_1\} \wedge C(m_i) = 1$.

ments are ill-formed. This assumption can be justified by using an NIC such that in the commit phase the receiver can efficiently check whether a commitment can be opened to some message. (Such an NIC can be obtained by using an injective one-way function with efficiently recognizable range [Blu82]).

3. The receiver samples a random challenge $\pi_{2,i} \in \{0, 1\}$ for each $i \in \{0, 1, 2, 3\}$.
4. The sender chooses a random $\Delta \in \{0, 1\}$ and sets $\hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta$ for each $i \in \{0, 1, 2, 3\}$. Then, for each $i \in \{0, 1, 2, 3\}$, the sender treats $\hat{\pi}_{2,i}$ as a verifier challenge message of Π and runs Π to obtain its third message $\pi_{3,i}$.

Soundness and WI can be proved by appropriately modifying the analysis of Construction 4.

- Soundness is proved via a union bound over $\Delta \in \{0, 1\}$. Note that to reuse the analysis of Construction 4, it suffices to show that the statement proven by Π is false for at most one index $i^* \in \{0, 1, 2, 3\}$. Thus, suppose that the statement is false for at least two indices. Then, for any fixed $\Delta \in \{0, 1\}$, the sender succeeds in all instances of Π with probability at most $(1/2)^2$ by the soundness of Π . Therefore, by a union bound over Δ , the probability that there exists some $\Delta \in \{0, 1\}$ for which the sender succeeds in all instances of Π is at most $2 \cdot (1/2)^2 = 1/2$.
- WI is proved by using that Π is special honest-verifier ZK. Recall that special honest-verifier ZK guarantees that for any $\pi_2^* \in \{0, 1\}$, one can simulate a proof (π_1, π_2^*, π_3) in a way indistinguishable from a real proof. Now, the key point of Construction 5 is that the sender can choose Δ so that the execution of Π can be replaced with a simulation for at most one index $i^* \in \{0, 1, 2, 3\}$: first choose a random $\hat{\pi}_{2,i^*}$ and simulate $(\pi_{1,i^*}, \hat{\pi}_{2,i^*}, \pi_{3,i^*})$, and then set $\Delta := \pi_{2,i^*} \oplus \hat{\pi}_{2,i^*}$ based on the challenge π_{2,i^*} sampled by the receiver. Thus, WI can be shown via the same simple hybrid argument as in Construction 4.

Finally, we obtain a black-box construction by using the MPC-in-the-head technique. In particular, the sender computes each c_i and $\pi_{1,i}, \pi_{3,i}$ in Construction 5 using the following simple special honest-verifier ZK commit-and-prove scheme.

- Instead of letting c_i be an NIC commitment to (m, r) , let $c_i = (c_{i,1}, \dots, c_{i,\nu})$ be NIC commitments to secret shares $s_{i,1}, \dots, s_{i,\nu}$ of (m, r) .
- Instead of using Π to generate $\pi_{1,i}$ and $\pi_{3,i}$, use the MPC-in-the-head technique to generate them.

Specifically, the MPC-in-the-head technique is used with the following functionality: the parties' private inputs are the secret shares $s_{i,1}, \dots, s_{i,\nu}$, and the functionality reconstructs a pair (m_i, r_i) from the shares and outputs 1 if and only if $r_i u + m_i \in \{v_0, v_1\} \wedge C(m_i) = 1$. Moreover, in the last round, the sender opens not only the views but also the shares of the parties selected by the receiver (i.e., $\{s_{i,j}, \text{view}_{i,j}\}_{j \in Q}$), and the receiver checks that the secret input in each $\text{view}_{i,j}$ is $s_{i,j}$.

The analysis proceeds as in Construction 5, with some minor adjustments for soundness. First, unlike Π , the MPC-in-the-head technique requires the challenge message to be longer than 1 bit, so we need to increase the number of parallel executions for a union bound.⁷ We also need to note that the MPC-in-the-head technique only guarantees that the committed shares are “close” to shares that satisfy the statement. (For example, if only one share is switched to an incorrect one, a random subset is likely to miss the incorrect share.) To ensure that a convincing proof still implies that the commitment can be opened to a message satisfying the statement, we relax the open phase verification condition: even when a small fraction of the committed shares are erroneous, an opening is accepted as long as the shares can be error-corrected. For this reason, in our actual scheme, shares are generated via Shamir's secret sharing rather than simple additive secret sharing.

⁷Specifically, we need to increase it from $4 = 3 \cdot 1 + 1$ to $3 \cdot O(\ell) + 1$, where ℓ is the challenge message length in the MPC-in-the-head technique. (It is observed by Chen et al. [CRT25] that, by cleverly using a 4-wise independent hash function, we can set $\ell = O(\log \nu)$ when the goal is constant soundness error.)

2.2 Non-Interactive Black-Box WI Commit-and-Prove

To remove interaction, we apply the Fiat–Shamir transformation where the hash function is instantiated by a targeted HSG. Since our interactive construction has two types of verifier challenges (u and π_2), we use the targeted HSG in a tree structure in the Fiat–Shamir transformation. That is, first a list of first-type challenges $\{u^{(\alpha)}\}_{\alpha \in [L]}$ is generated by using the targeted HSG, and then for each $\alpha \in [L]$, a list of second-type challenges $\{\pi_2^{(\alpha, \beta)}\}_{\beta \in [L]}$ is generated by using the targeted HSG again. (Here L is the list size of the HSG.) The soundness analysis follows naturally from existing analyses (in particular, Chen et al. [CRT25]) and uses the notion of *round-by-round soundness* [CCH⁺19], which was originally developed for soundly implementing the Fiat–Shamir transformation using *correlation-intractable hash functions* [CGH04]. The analysis of WI is achieved by appropriately setting the parameters of the secret sharing and the MPC so that not too many shares and views are opened. (As a minor issue, the linear polynomial $r_i u + m_i$ used in the simplified description above is replaced in the formal construction by a polynomial f_i of sufficiently large degree so that $f_i(u^{(1)}), \dots, f_i(u^{(L)})$ do not reveal m_i .)

Finally, we give minor comments on parameter selection. Specifically, the parameters in our construction must be carefully set under the following constraints.

- The message length must not be too large. This is because the list size of the HSG is determined by the HSG input length. (If the message length is too large, the value of L becomes large and WI may fail.)
- The verifier randomness must not be too long. This is because existing targeted HSGs [CT22a] can generate only pseudorandom strings of limited length.

See Section 4 for the formal construction and the parameter choices underlying these constraints.

3 Preliminaries

3.1 Notations and Conventions

We use λ to denote the security parameter. For any binary strings $a, b \in \{0, 1\}^*$, we write $a \parallel b$ and (a, b) for their concatenation. For any $n \in \mathbb{N}$, we use $[n]$ to denote the set $\{1, \dots, n\}$. For any sequence $a = (a_1, \dots, a_n)$ and subset $Q \subseteq [n]$, we use $a|_Q$ to denote the subsequence $(a_i)_{i \in Q}$.⁸ When it is clear from the context, we abuse notation and often use set notation for sequences. That is, the sequence $(a_1, \dots, a_n)$ and subsequence $a|_Q$ are often denoted as $\{a_i\}_{i \in [n]}$ and $\{a_i\}_{i \in Q}$, respectively. We use poly to denote an unspecified polynomial, negl to denote an unspecified negligible function, and PPT as an abbreviation for “probabilistic polynomial-time.” For a set S , we write $s \leftarrow S$ to denote sampling s uniformly from S . For any probabilistic algorithm Algo and an input x , we write $y \leftarrow \text{Algo}(x)$ to denote running $\text{Algo}(x)$ with uniform randomness and obtaining output y . We use $\perp$ to denote a special error symbol outside $\{0, 1\}^*$. Whenever an honest algorithm aborts, we regard each component of its output as $\perp$. Unless otherwise specified, any honest function/algorithm/Turing machine/circuit is assumed to output $\perp$ on any input containing $\perp$. These conventions do not restrict adversarial algorithms.

For any $\ell \in \mathbb{N}$, $\delta \in [0, 1]$, and string $u = (u_1, \dots, u_\ell) \in \Sigma^\ell$ over an alphabet Σ , we say that a string v is δ -close to u if $v = (v_1, \dots, v_\ell) \in \Sigma^\ell$ and $|\{i \mid u_i \neq v_i\}| \leq \delta \cdot \ell$. We say that v is δ -close to a set of strings S if v is δ -close to some $u \in S$. We say that v is δ -far from u (resp., S) if v is not δ -close to u (resp., S).

The reader is assumed to be familiar with basic notions in cryptography, such as computational indistinguishability and one-way functions. (For basic definitions, see standard textbooks such as [Gol01].) In this paper, all security notions are defined with respect to uniform adversaries.

⁸For simplicity, we often regard $a|_Q$ as implicitly including the information about Q .

3.2 Non-interactive Commitment

Definition 1 (Non-interactive commitment). *A pair of Turing machines (Com, Rec) , where Com is probabilistic and Rec is deterministic, is called a non-interactive commitment scheme if it satisfies the following.*

Correctness. *For every $\lambda \in \mathbb{N}$ and $m \in \{0, 1\}^*$,*

$$\Pr \left[\text{Rec}(1^\lambda, c, m, d) = 1 \mid (c, d) \leftarrow \text{Com}(1^\lambda, m) \right] = 1.$$

Perfect binding. *For any $\lambda \in \mathbb{N}$, $c \in \{0, 1\}^*$, and $(m_0, m_1) \in (\{0, 1\}^*)^2$ such that $m_0 \neq m_1$, there is no $(d_0, d_1) \in (\{0, 1\}^*)^2$ satisfying $\text{Rec}(1^\lambda, c, m_0, d_0) = 1 \wedge \text{Rec}(1^\lambda, c, m_1, d_1) = 1$.*

Computational hiding. *For every $\lambda \in \mathbb{N}$ and every pair of PPT Turing machines $(\mathcal{A}_1, \mathcal{A}_2)$,*

$$\Pr \left[\begin{array}{l} |m_0| = |m_1| \\ \wedge \mathcal{A}_2(\text{st}_{\mathcal{A}}, c) = b \end{array} \mid \begin{array}{l} (m_0, m_1, \text{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda) \\ b \leftarrow \{0, 1\} \\ (c, d) \leftarrow \text{Com}(1^\lambda, m_b) \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda).$$

The first part of the output of Com is called the *commitment*, and the second part is called the *decommitment*. A (potentially ill-formed) commitment c is called *valid* if it can be opened to some message $m \in \{0, 1\}^*$, i.e., $\exists m, d \in \{0, 1\}^*$ s.t. $\text{Rec}(1^\lambda, c, m, d) = 1$; otherwise, it is called *invalid*. When a commitment c is valid, its *committed value* is defined as the unique message m to which it can be opened. When a commitment c is invalid, its committed value is defined as $\perp$.

In this paper, we additionally require a stronger binding property for non-interactive commitment schemes.⁹ Informally, we require that any invalid commitment admits an efficiently verifiable certificate of invalidity. The formal definition is as follows.

Definition 2 (Strong perfect binding). *A non-interactive commitment scheme (Com, Rec) is said to satisfy strong perfect binding if it satisfies the following instead of the standard perfect binding requirement.*

1. **(Both valid and invalid commitments can be opened.)** *For any $\lambda \in \mathbb{N}$ and $c \in \{0, 1\}^*$, there exist $m \in \{0, 1\}^* \cup \{\perp\}$ and $d \in \{0, 1\}^*$ such that $\text{Rec}(1^\lambda, c, m, d) = 1$.*
2. **(Both valid and invalid commitments are binding.)** *For any $\lambda \in \mathbb{N}$, $c \in \{0, 1\}^*$, and $(m_0, m_1) \in (\{0, 1\}^* \cup \{\perp\})^2$ such that $m_0 \neq m_1$, there is no $(d_0, d_1) \in (\{0, 1\}^*)^2$ satisfying $\text{Rec}(1^\lambda, c, m_0, d_0) = 1 \wedge \text{Rec}(1^\lambda, c, m_1, d_1) = 1$.*

Non-interactive commitment schemes with strong perfect binding can be constructed from an injective one-way function with efficiently recognizable range.¹⁰

3.3 Non-Interactive Witness-Indistinguishable Commit and Prove

We define non-interactive witness-indistinguishable (WI) commit-and-prove schemes. Our definition is based on that of Khurana et al. [KOS18] for interactive WI commit-and-prove schemes. For simplicity, we restrict attention to the case where committed messages have length λ .

Definition 3. *A non-interactive witness-indistinguishable (WI) commit-and-prove scheme consists of four Turing machines $(\text{Com}, \text{Rec}, \text{Prv}, \text{Vrfy})$ that satisfy the following.*

⁹See Footnote 6 in Section 2.

¹⁰Let f be an injective one-way function and let hc be the hardcore bit of f . Then, it suffices to consider the commitment scheme [Blu82] in which a commitment to a message $m \in \{0, 1\}$ is $c = (f(r), m \oplus \text{hc}(r))$ for random $r \in \{0, 1\}^\lambda$. Rec can check the invalidity of c by checking whether the first component of c is outside the range of f . (The certificate d can be an empty string.)

- **Commitment correctness.** For every $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, and $m \in \{0, 1\}^\lambda$,

$$\Pr \left[\text{Rec}(1^\lambda, 1^T, c, m, d) = 1 \mid (c, d) \leftarrow \text{Com}(1^\lambda, 1^T, m) \right] = 1.$$

- **Proof correctness.** For every $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, $m \in \{0, 1\}^\lambda$, and a circuit C such that $|C| = T$ and $C(m) = 1$,

$$\Pr \left[\text{Vrfy}(1^\lambda, 1^T, C, c, \pi) = 1 \mid \begin{array}{l} (c, d) \leftarrow \text{Com}(1^\lambda, 1^T, m) \\ \pi \leftarrow \text{Prv}(1^\lambda, 1^T, C, c, m, d) \end{array} \right] = 1.$$

- **Perfect binding.** For any $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, $c \in \{0, 1\}^*$, and $(m_0, m_1) \in (\{0, 1\}^\lambda)^2$ such that $m_0 \neq m_1$, there is no $(d_0, d_1) \in (\{0, 1\}^*)^2$ satisfying $\text{Rec}(1^\lambda, 1^T, c, m_0, d_0) = 1 \wedge \text{Rec}(1^\lambda, 1^T, c, m_1, d_1) = 1$.

- **Computational hiding.** For every pair of PPT Turing machines $(\mathcal{A}_1, \mathcal{A}_2)$, every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, and every $\lambda \in \mathbb{N}$,

$$\Pr \left[\begin{array}{l} |m_0| = |m_1| \\ \wedge \mathcal{A}_2(\text{st}_{\mathcal{A}}, c) = b \end{array} \mid \begin{array}{l} (m_0, m_1, \text{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, 1^{T(\lambda)}) \\ b \leftarrow \{0, 1\} \\ (c, d) \leftarrow \text{Com}(1^\lambda, 1^{T(\lambda)}, m_b) \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda).$$

- **Soundness.** For every PPT Turing machine $\mathcal{A}$, every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, and every $\lambda \in \mathbb{N}$,

$$\Pr \left[\text{SUCCESS}(\lambda, T, C, c, \pi) \mid (C, c, \pi) \leftarrow \mathcal{A}(1^\lambda, 1^{T(\lambda)}) \right] \leq \text{negl}(\lambda),$$

where $\text{SUCCESS}(\lambda, T, C, c, \pi)$ is the event that all of the following hold.

- For every $m \in \{0, 1\}^\lambda$ such that $C(m) = 1$ and every $d \in \{0, 1\}^*$, it holds $\text{Rec}(1^\lambda, 1^{T(\lambda)}, c, m, d) \neq 1$.
- $\text{Vrfy}(1^\lambda, 1^{T(\lambda)}, C, c, \pi) = 1$.
- **Witness indistinguishability.** For every triple of PPT Turing machines $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$, every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, and every $\lambda \in \mathbb{N}$,

$$\Pr \left[\begin{array}{l} |m_0| = |m_1| \\ \wedge C(m_0) = C(m_1) = 1 \\ \wedge \mathcal{A}_3(\text{st}_{\mathcal{A},2}, \pi) = b \end{array} \mid \begin{array}{l} (m_0, m_1, \text{st}_{\mathcal{A},1}) \leftarrow \mathcal{A}_1(1^\lambda, 1^{T(\lambda)}) \\ b \leftarrow \{0, 1\} \\ (c, d) \leftarrow \text{Com}(1^\lambda, 1^{T(\lambda)}, m_b) \\ (C, \text{st}_{\mathcal{A},2}) \leftarrow \mathcal{A}_2(\text{st}_{\mathcal{A},1}, c) \\ \pi \leftarrow \text{Prv}(1^\lambda, 1^{T(\lambda)}, C, c, m_b, d) \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda).$$

Remark 1 (Difference from the definition of Khurana et al. [KOS18]). Unlike the definition of Khurana et al., the soundness definition above only requires that the commitment c can be opened to a message m satisfying the statement, and it does not require that such an m be efficiently extractable. $\diamond$

3.4 Shamir's Secret Sharing

We recall Shamir's secret sharing [Sha79]. In this paper, we use the following well-known useful fact: shares produced by Shamir's secret sharing can be viewed as a Reed–Solomon code-word, which allows error correction even when fewer than $1/3$ of the shares are corrupted [MS81].

Theorem 1 (Shamir's secret sharing scheme). *There exists a pair of polynomial-time Turing machines (Share, Recon) (where Share is probabilistic and Recon is deterministic) satisfying the following.*

- **Correctness.** For any $n \in \mathbb{N}$, $t \in \mathbb{N}$, and $m \in \{0, 1\}^*$ such that $t < n$,

$$\Pr [\text{Recon}(s_1, \dots, s_n) = m \mid (s_1, \dots, s_n) \leftarrow \text{Share}(1^n, 1^t, m)] = 1.$$

- **Robust reconstruction.** Fix any $n \in \mathbb{N}$ and $t \in \mathbb{N}$ such that $t < n/3$. For each $m \in \{0, 1\}^*$, let S_m denote the range of $\text{Share}(1^n, 1^t, m)$. Then, for any $s = (s_1, \dots, s_n) \in (\{0, 1\}^* \cup \{\perp\})^n$, the following hold.

- If there exists $m \in \{0, 1\}^*$ such that s is (t/n) -close to S_m , then $\text{Recon}(s_1, \dots, s_n)$ outputs m .¹¹
- Otherwise (i.e., s is (t/n) -far from S_m for every $m \in \{0, 1\}^*$), $\text{Recon}(s_1, \dots, s_n)$ outputs $\perp$.

Furthermore, S_m is efficiently checkable, i.e., for any $s = (s_1, \dots, s_n)$ one can decide in polynomial time whether $s \in S_m$.

- **Privacy.** There exists a PPT Turing machine $\mathcal{S}$ (called the simulator) such that for any $n \in \mathbb{N}$, $t \in \mathbb{N}$, $A \subset [n]$, and $m \in \{0, 1\}^*$ such that $t < n$ and $|A| \leq t$, the distributions $\{s|_A \mid s := (s_1, \dots, s_n) \leftarrow \text{Share}(1^n, 1^t, m)\}$ and $\mathcal{S}(1^n, 1^t, 1^{|m|}, A)$ are identically distributed.

3.5 Targeted Hitting-Set Generator

We recall the definition of targeted hitting-set generators (HSGs) [Gol11, CT22a, CT23b]. The following description is taken from [CRT25].

Definition 4 (avoider). We say that a function $D : \{0, 1\}^m \rightarrow \{0, 1\}$ is an $(1 - \epsilon)$ -dense avoider for a set $S \subseteq \{0, 1\}^m$ if

$$\Pr [D(r) = 1 \mid r \leftarrow \{0, 1\}^m] \geq 1 - \epsilon \quad \wedge \quad \forall s \in S, D(s) = 0.$$

Definition 5 (targeted HSGs). Let G be an algorithm that takes input $x \in \{0, 1\}^n$ and outputs $p = p(n)$ strings of length $m = m(n)$, and let $X = \{X_n\}_{n \in \mathbb{N}}$ be an ensemble of distributions (where X_n is over $\{0, 1\}^n$). We say that G is a targeted hitting-set generator that is (ϵ, δ) -pseudorandom over X for a class $\mathcal{C}$ of uniform algorithms (or (ϵ, δ) -targeted HSG over X for $\mathcal{C}$ in short) if for every Turing machine $M \in \mathcal{C}$ and every sufficiently large $n \in \mathbb{N}$,

$$\Pr [M(x, \cdot) \text{ is a } (1 - \epsilon)\text{-dense avoider for } G(x) \mid x \leftarrow X_n] \leq \delta(n).$$

In this paper, we use the targeted HSG from [CRT25] (which is based on [CT21, Proposition 6.2] and [CT22b, Theorem 7.1]). Roughly speaking, this targeted HSG is obtained by assuming the existence of a function that is “non-batch-computable” in the following sense: there exists a function $f : \{0, 1\}^n \rightarrow \{0, 1\}^k$ such that each output bit of f can be computed in time T (and hence the k -bit output of f can trivially be computed in time $k \cdot T$), but the k -bit output of f cannot be computed in time $k^\delta T$ for some small $\delta > 0$. This non-batch-computability is required to hold when the input x to f is sampled from any polynomial-time samplable distribution. In fact, the actual assumption is even stronger: even approximating the k -bit output of f , namely computing the output of f on a 0.99 fraction of the k coordinates, is assumed to be impossible. (For more about non-batch-computability, see, e.g., [CT21, Section 1.2].)

Theorem 2 ([CRT24, the first half of Corollary 7.16]). For a sufficiently small $\gamma > 0$, assume that there is $f : \{0, 1\}^n \rightarrow \{0, 1\}^{k=n^\gamma}$ such that each output bit of f is computable in time $T(n) = \text{poly}(n)$, but the following holds. For every algorithm Rec running in time $T \cdot k^{0.01}$ and making queries to a

¹¹This implies that for any s , there exists at most one m such that s is (t/n) -close to S_m .

SAT oracle $\mathcal{O}$, every polynomial-time samplable distribution $X = \{X_n\}_{n \in \mathbb{N}}$, and every sufficiently large $n \in \mathbb{N}$, the probability over $x \sim X_n$ that the following holds is negligible:

$$\Pr [\text{Rec}^{\mathcal{O}}(x)_i = f(x)_i] \geq 0.99,$$

where the probability above is over the random coins of Rec and over $i \in [k]$.

Then, there is a $(1/2, N^{-\omega(1)})$ -targeted HSG running in polynomial time that is pseudorandom for $\text{DTIME}[N]^{\text{SAT}}$ over all polynomial-time samplable distributions, and on inputs of length N outputs $N^{O(\epsilon)}$ strings of length N^ϵ , where $\epsilon = \eta \cdot \gamma$ for a constant $\eta > 0$.¹²

3.6 Secure Computation

We assume the reader is familiar with MPC protocols. In this paper, we consider MPC protocols in the honest-majority setting, under a synchronous communication model with secure point-to-point channels and a broadcast channel. When an MPC protocol is run among ν parties $P_1, \dots, P_\nu$, each party P_i is assigned a common public input x and a local private input y_i , and the parties jointly evaluate a (deterministic single-output) ν -party functionality $\mathcal{F}$ to obtain an output $z := \mathcal{F}(x, y_1, \dots, y_\nu)$ for all parties. The running time of each party P_i is $\text{poly}(\nu, T)$, where T is the circuit size of $\mathcal{F}$. The view of each party P_i consists of its local input, its randomness, and the messages received by P_i during the execution.

We consider malicious security, and as in prior MPC-in-the-head literature [IKOS09], we use the following property-based security rather than real/ideal-based security.

Definition 6 (Correctness). We say that an MPC protocol Π realizes a ν -party functionality $\mathcal{F}$ with perfect correctness if for all inputs $(x, y_1, \dots, y_\nu)$, the probability that the output of some party is different from $\mathcal{F}(x, y_1, \dots, y_\nu)$ is 0.¹³

Definition 7 (Privacy). Let $1 < \tau < \nu$. We say that an MPC protocol Π realizes a ν -party functionality $\mathcal{F}$ with perfect τ -privacy if there exists a PPT simulator S such that for any inputs $x, y_1, \dots, y_\nu$ and every set of corrupted parties $T \subseteq [\nu]$ with $|T| \leq \tau$, the joint view $\text{view}_T(x, y_1, \dots, y_\nu)$ of parties in T is distributed identically to $S(T, x, (y_i)_{i \in T}, \mathcal{F}(x, y_1, \dots, y_\nu))$.

Definition 8 (Robustness). We say that an MPC protocol Π realizes a ν -party functionality $\mathcal{F}$ with perfect τ -robustness if it is perfectly correct in the presence of a semi-honest adversary as in Definition 6, and furthermore for any computationally unbounded malicious adversary corrupting a set T of at most τ parties, and for any inputs $(x, y_1, \dots, y_\nu)$, the following robustness property holds. If there is no $(y'_1, \dots, y'_\nu)$ such that $|\{i \mid y_i \neq y'_i\}| \leq |T|$ and $\mathcal{F}(x, y'_1, \dots, y'_\nu) = 1$, then the probability that some uncorrupted party outputs 1 in an execution of Π in which the inputs of the honest parties are consistent with $(x, y_1, \dots, y_\nu)$ is 0.¹⁴

Finally, as in prior MPC-in-the-head literature [IKOS09], we introduce consistency between the views of two parties.

Definition 9 (Consistent views). We say that a pair of views $(\text{view}_i, \text{view}_j)$ is consistent (with respect to the MPC protocol Π and some public input x) if the outgoing messages implicit in view_i (with respect to Π and x) are identical to the incoming messages reported in view_j and vice versa.

¹²More precisely, “pseudorandom for $\text{DTIME}[N]^{\text{SAT}}$ ” means the following: for every algorithm D that gets input $z \in \{0, 1\}^N$ and random coins $r \in \{0, 1\}^{N^\epsilon}$ and runs in linear time $O(N)$ while making oracle queries to SAT, and every polynomial-time samplable distribution $Z = \{Z_N\}_{N \in \mathbb{N}}$, the probability over $z \sim Z_N$ that $D(z, \cdot)$ is a $(1/2)$ -dense avoider for the output list of the generator on z is negligible [CRT24, Footnote 53].

¹³The probability is over the independent choices of the random inputs $r_1, \dots, r_\nu$ to the parties.

¹⁴This definition of robustness is slightly stronger than that of Ishai et al. [IKOS09], but it is still clearly implied by real/ideal-based security.

4 Non-Interactive Witness-Indistinguishable Commit and Prove

In this section, we prove our main result.

Theorem 3. *Assume the existence of the following primitives.*

- A non-interactive commitment scheme $\text{NIC} = (\text{NIC.Com}, \text{NIC.Rec})$ satisfying strong perfect binding.
- A $(1/2, \text{negl})$ -targeted HSG HSG for $\text{DTIME}[N]^{\text{SAT}}$ over all polynomial-time samplable distributions, where on inputs of length N , the HSG runs in time $\text{poly}(N)$ and outputs $\text{poly}(N^\epsilon)$ strings, each of length N^ϵ for a sufficiently small constant $\epsilon > 0$.¹⁵

Then, there exists a non-interactive witness-indistinguishable commit-and-prove scheme that makes only black-box use of NIC.

Theorem 3 yields the following corollary (cf. Section 3.2 and Theorem 2).

Corollary 1. *Assume the following.*

- There is an injective one-way function with efficiently recognizable range.
- For a sufficiently small $\gamma > 0$, there is $f : \{0, 1\}^n \rightarrow \{0, 1\}^{k=n^\gamma}$ such that each output bit of f is computable in time $T(n) = \text{poly}(n)$, but the following holds. For every algorithm Rec running in time $T \cdot k^{0.01}$ and making queries to a SAT oracle $\mathcal{O}$, every polynomial-time samplable distribution $X = \{X_n\}_{n \in \mathbb{N}}$, and every sufficiently large $n \in \mathbb{N}$, the probability over $x \sim X_n$ that the following holds is negligible: $\Pr[\text{Rec}^{\mathcal{O}}(x)_i = f(x)_i] \geq 0.99$, where the probability is over the random coins of Rec and over $i \in [k]$.

Then, there exists a non-interactive witness-indistinguishable commit-and-prove scheme that makes only black-box use of the injective one-way function.

We prove Theorem 3 in three steps. In the first two steps, we construct an interactive black-box WI commit-and-prove scheme (specifically, first a special honest-verifier ZK commit-and-prove scheme, and then a WI one based on it). In the final step, we apply the Fiat–Shamir transformation to obtain a non-interactive construction.

4.1 Step 1. Interactive SHVZK Commit and Prove

In what follows, we use the following information-theoretic primitives besides NIC and HSG.

- Shamir’s secret sharing scheme ($\text{Share}, \text{Recon}$).
- An MPC protocol Π satisfying the following. For any $\nu, \tau \in \mathbb{N}$ such that $\tau < \nu/3$, Π enables ν parties to jointly compute the following functionality $\mathcal{F}$ with perfect τ -privacy and perfect τ -robustness.
 - When the common public input is a circuit C and the ν private inputs are $s_1, \dots, s_\nu$, $\mathcal{F}$ outputs 1 if and only if all of the following hold.
 - * $\text{Recon}(s_1, \dots, s_\nu)$ outputs some m such that $C(m) = 1$.
 - * $(s_1, \dots, s_\nu)$ is in the range of $\text{Share}(1^\nu, 1^\tau, m)$.

An MPC protocol satisfying the above can be realized, for example, by the BGW protocol [BGW88].

¹⁵For the required size of ϵ , see the proof of Lemma 8.

We define a commit-and-prove scheme $\text{SHVZKCP} = (\text{SHVZKCP.Com}, \text{SHVZKCP.Rec}, \text{SHVZKCP.Priv}_1, \text{SHVZKCP.Vrfy}_1, \text{SHVZKCP.Priv}_2, \text{SHVZKCP.Vrfy}_2)$ as follows. (For later use, we require SHVZKCP to satisfy strong binding just like NIC . Thus, SHVZKCP.Rec verifies not only decommitments for valid commitments but also certificates of invalidity for invalid commitments.) At a high level, SHVZKCP is a simple special honest-verifier ZK commit-and-prove scheme based on the MPC-in-the-head technique described in [Section 2.1](#). The only non-trivial ingredient is a trick due to Chen et al. [[CRT25](#)], which uses a 4-wise independent hash function to reduce the verifier's randomness to logarithmic length while retaining constant soundness error. (This logarithmic randomness bound is needed for the later Fiat–Shamir transformation using a targeted HSG; see [Section 2.2](#) and [Section 4.3](#).)

Special honest-verifier ZK commit-and-prove SHVZKCP .

- $\text{SHVZKCP.Com}(1^\lambda, 1^T, m)$.
If $|m| \neq \lambda$, abort. Otherwise, do the following.
 1. Compute $(s_1, \dots, s_\nu) \leftarrow \text{Share}(1^\nu, 1^\tau, m)$ for $\tau := \max(\lambda, T)$ and $\nu := 3\tau + 1$.
 2. Compute $(c_i, d_i) \leftarrow \text{NIC.Com}(1^\lambda, s_i)$ for each $i \in [\nu]$.
 3. Output $c := (c_1, \dots, c_\nu)$ and $d := (s_1, d_1, \dots, s_\nu, d_\nu)$.
- $\text{SHVZKCP.Rec}(1^\lambda, 1^T, c, m, d)$.
Parse c and d as $(c_1, \dots, c_\nu)$ and $(s_1, d_1, \dots, s_\nu, d_\nu)$, respectively. Then, output 1 if and only if all of the following hold.
 - $\text{NIC.Rec}(1^\lambda, c_i, s_i, d_i) = 1$ for every $i \in [\nu]$.
 - $\text{Recon}(s_1, \dots, s_\nu) = m$.
- $\text{SHVZKCP.Priv}_1(1^\lambda, 1^T, C, c, m, d)$.
If $|m| \neq \lambda$ or $|C| \neq T$, abort. Otherwise, do the following.
 1. Parse c and d as $(c_1, \dots, c_\nu)$ and $(s_1, d_1, \dots, s_\nu, d_\nu)$, respectively.
 2. Run the MPC protocol Π in the head with parameters ν, τ , common public input C , and ν private inputs $s_1, \dots, s_\nu$. Let the views of the ν parties be denoted by $(\text{view}_1, \dots, \text{view}_\nu)$.
 3. For each $i \in [\nu]$, compute $(c'_i, d'_i) \leftarrow \text{NIC.Com}(1^\lambda, \text{view}_i)$.
 4. Output $\pi_1 := (c'_1, \dots, c'_\nu)$ and $\text{st} := (s_1, d_1, \text{view}_1, d'_1, \dots, s_\nu, d_\nu, \text{view}_\nu, d'_\nu)$.
- $\text{SHVZKCP.Vrfy}_1(1^\lambda, 1^T)$.
 1. Sample a 4-wise independent hash function $h : [\nu] \rightarrow [8\tau/q]$, where $q := \Theta(\sqrt{\tau})$ is a sufficiently large parameter.^a
 2. Output $\pi_2 := h$.
- $\text{SHVZKCP.Priv}_2(\text{st}, \pi_2)$.
 1. Parse st as $(s_1, d_1, \text{view}_1, d'_1, \dots, s_\nu, d_\nu, \text{view}_\nu, d'_\nu)$ and parse π_2 as a 4-wise independent hash function $h : [\nu] \rightarrow [8\tau/q]$.
 2. Define a family of 4-wise independent random variables $\{X_i\}_{i \in [\nu]}$ as follows.

$$X_i := \begin{cases} 1 & \text{if } h(i) = 1 \\ 0 & \text{otherwise} \end{cases}.$$
 3. Let $Q := \{i \in [\nu] \mid X_i = 1\}$. If $|Q| > q$, output $\pi_3 := \perp$. Otherwise, output $\pi_3 := \{s_i, d_i, \text{view}_i, d'_i\}_{i \in Q}$.

- $\text{SHVZKCP.Vrfy}_2(1^\lambda, 1^T, C, c, \pi_1, \pi_2, \pi_3)$.

If $|C| \neq T$, output 0. Also, define Q as in SHVZKCP.Priv_2 , and if $|Q| > q$ then output 1. In all other cases, parse c, π_1 , and π_3 as $(c_1, \dots, c_\nu), (c'_1, \dots, c'_\nu)$, and $\{s_i, d_i, \text{view}_i, d'_i\}_{i \in Q}$, respectively, and output 1 if and only if all of the following hold.^b

- $\text{NIC.Rec}(1^\lambda, c_i, s_i, d_i) = 1$ and $\text{NIC.Rec}(1^\lambda, c'_i, \text{view}_i, d'_i) = 1$ for every $i \in Q$.
- For each $i \in Q$, the secret input of the i -th party in view_i equals s_i .
- For each $i \in Q$, the output of the i -th party in view_i is 1.^c
- For each $i, j \in Q$, view_i and view_j are consistent.

^aFor the required size of q , see the proof of Lemma 3.

^bFor each i with $\text{view}_i = \perp$, we treat view_i as an arbitrary default view and define its secret input, output, and consistency with other parties' views accordingly.

^cThat is, the i -th party outputs 1 when the common public input is C and its view (including its local secret input, randomness, and incoming messages) is view_i .

It is immediate from the construction that SHVZKCP makes only black-box use of NIC, as it accesses NIC only through the input/output interfaces of NIC.Com and NIC.Rec . We now establish the required properties of SHVZKCP, beginning with correctness.

Lemma 1.

- **Commitment correctness.** For every $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, and $m \in \{0, 1\}^\lambda$,

$$\Pr \left[\text{SHVZKCP.Rec}(1^\lambda, 1^T, c, m, d) = 1 \mid (c, d) \leftarrow \text{SHVZKCP.Com}(1^\lambda, 1^T, m) \right] = 1.$$

- **Proof correctness.** For every $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, $m \in \{0, 1\}^\lambda$, and a circuit C such that $|C| = T$ and $C(m) = 1$,

$$\Pr \left[\text{SHVZKCP.Vrfy}_2(1^\lambda, 1^T, C, c, \pi_1, \pi_2, \pi_3) = 1 \mid \begin{array}{l} (c, d) \leftarrow \text{SHVZKCP.Com}(1^\lambda, 1^T, m) \\ (\pi_1, \text{st}) \leftarrow \text{SHVZKCP.Priv}_1(1^\lambda, 1^T, C, c, m, d) \\ \pi_2 \leftarrow \text{SHVZKCP.Vrfy}_1(1^\lambda, 1^T) \\ \pi_3 \leftarrow \text{SHVZKCP.Priv}_2(\text{st}, \pi_2) \end{array} \right] = 1.$$

Moreover, SHVZKCP.Vrfy_1 is a public-coin algorithm,¹⁶ and $|\pi_2| = O(\log(\lambda + T))$. Also, there exists a universal constant $K > 0$ (independent of λ and T) such that $|c| + |\pi_1| \leq \nu^K$ for $\nu = 3 \max(\lambda, T) + 1$.

Proof. Commitment correctness follows from the correctness of NIC and that of Share. Proof correctness follows from these and the correctness of Π . The verifier's randomness complexity is $O(\log \nu) = O(\log(\lambda + T))$ since it suffices to choose a 4-wise independent hash function $h : [\nu] \rightarrow [8\tau/q]$. Finally, $|c| + |\pi_1| = |(c_1, \dots, c_\nu)| + |(c'_1, \dots, c'_\nu)| \leq \text{poly}(\lambda, T, \nu) \leq \text{poly}(\nu)$. $\square$

Next, we show strong perfect binding and computational hiding.

Lemma 2.

- **Strong perfect binding.**

1. For any $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, and $c \in \{0, 1\}^*$, there exist $m \in \{0, 1\}^* \cup \{\perp\}$ and $d \in \{0, 1\}^*$ such that $\text{SHVZKCP.Rec}(1^\lambda, 1^T, c, m, d) = 1$.
2. For any $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, $c \in \{0, 1\}^*$, and $(m_0, m_1) \in (\{0, 1\}^* \cup \{\perp\})^2$ such that $m_0 \neq m_1$, there is no $(d_0, d_1) \in (\{0, 1\}^*)^2$ satisfying $\text{SHVZKCP.Rec}(1^\lambda, 1^T, c, m_0, d_0) = 1 \wedge \text{SHVZKCP.Rec}(1^\lambda, 1^T, c, m_1, d_1) = 1$.

¹⁶That is, SHVZKCP.Vrfy_1 samples a string uniformly at random and simply outputs it.

- **Computational hiding.** For every pair of PPT Turing machines (A_1, A_2) , every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, and every $\lambda \in \mathbb{N}$,

$$\Pr \left[\begin{array}{l} |m_0| = |m_1| \\ \wedge A_2(\text{st}_A, c) = b \end{array} \mid \begin{array}{l} (m_0, m_1, \text{st}_A) \leftarrow A_1(1^\lambda, 1^{T(\lambda)}) \\ b \leftarrow \{0, 1\} \\ (c, d) \leftarrow \text{SHVZKCP.Com}(1^\lambda, 1^{T(\lambda)}, m_b) \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda).$$

Proof. Strong perfect binding follows from the strong perfect binding of NIC and the fact that Recon is deterministic. Computational hiding follows from the computational hiding of NIC. $\square$

Next, we show soundness. As mentioned in the technical overview (Section 2.2), we need a strengthened variant called *round-by-round soundness* [CCH⁺19]. Informally, it guarantees that, given a partial transcript (C, c, π_1, π_2) , we can decide whether the sender is “doomed” in a way that satisfies the following properties.

- When the statement (C, c) is false (i.e., c cannot be opened to any message m such that $C(m) = 1$), then for any π_1 , the sender is doomed in the partial transcript (C, c, π_1, π_2) with high probability over the choice of π_2 .
- If the sender is doomed in a partial transcript (C, c, π_1, π_2) , then for any π_3 , the receiver rejects in the full transcript $(C, c, \pi_1, \pi_2, \pi_3)$.

The formal statement appears below.

Lemma 3 (Round-by-round soundness). *There exists a Boolean function Doomed satisfying the following.*

1. For every λ, T, C , and c such that $T = \text{poly}(\lambda)$, if the commitment c cannot be opened to any m such that $C(m) = 1$, (i.e., if for every m such that $C(m) = 1$ and every d , $\text{SHVZKCP.Rec}(1^\lambda, 1^T, c, m, d) \neq 1$), then

$$\forall \pi_1 : \Pr \left[\text{Doomed}(\lambda, T, C, c, \pi_1, \pi_2) = 1 \mid \pi_2 \leftarrow \text{SHVZKCP.Vrfy}_1(1^\lambda, 1^T) \right] \geq \frac{1}{2}. \quad (1)$$

2. For any λ, T, C, c, π_1 , and π_2 such that $T = \text{poly}(\lambda)$, if $\text{Doomed}(\lambda, T, C, c, \pi_1, \pi_2) = 1$, then

$$\forall \pi_3 : \text{SHVZKCP.Vrfy}_2(1^\lambda, 1^T, C, c, \pi_1, \pi_2, \pi_3) = 0.$$

Furthermore, Doomed can be computed in time $\text{poly}(\nu)$ with oracle access to $\text{NTIME}[\text{poly}(\nu)]$, where $\nu = 3 \max(\lambda, T) + 1$.

In the proof of Lemma 3 below, we use the following elementary facts.

Fact 1 ([CRT24, Fact A.1 (slightly augmented)]). Let $X_1, \dots, X_n$ be pairwise independent Bernoulli random variables with the same mean $p \in (0, 1]$, and let $\mu := \mathbb{E}[\sum_{i=1}^n X_i]$ be the expectation of their sum. Then,

$$\Pr \left[\left| \sum_{i=1}^n X_i - \mu \right| \geq \mu \right] \leq \frac{1}{np}.$$

In particular,

$$\Pr [X_1 = \dots = X_n = 0] \leq \frac{1}{np}.$$

$\diamond$

Fact 1 follows immediately from Chebyshev’s inequality.

Fact 2. If the size of a minimum vertex cover of a graph G is greater than t , then G has a matching of size greater than $t/2$. $\diamond$

Fact 2 holds since if the maximum matching contains at most $t/2$ edges, the vertices of this matching form a vertex cover of size at most t .

Proof of Lemma 3. Define Doomed as follows.

$\text{Doomed}(\lambda, T, C, c, \pi_1, \pi_2)$.

1. Parse c and π_1 as $(c_1, \dots, c_\nu)$ and $(c'_1, \dots, c'_\nu)$, respectively. For each $i \in [\nu]$, define s_i and view_i to be the committed values of c_i and c'_i , respectively.^a Define Q as in SHVZKCP.Prv_2 .
2. If $|Q| > q$, output 0. Otherwise, output 1 if and only if $\{s_i, \text{view}_i\}_{i \in Q}$ fails to satisfy one of the last three conditions in SHVZKCP.Vrfy_2 . That is, output 1 if and only if $|Q| \leq q$ and moreover one of the following holds.^b
 - For some $i \in Q$, the secret input of the i -th party in view_i differs from s_i .
 - For some $i \in Q$, the output of the i -th party in view_i differs from 1.
 - For some $i, j \in Q$, view_i and view_j are inconsistent.

^aIf c_i (resp., c'_i) is an invalid commitment, we define $s_i := \perp$ (resp., $\text{view}_i := \perp$).

^bAs in SHVZKCP.Vrfy_2 , for each i with $\text{view}_i = \perp$ we treat view_i as an arbitrary default view and define its secret input, output, and consistency with other parties' views accordingly.

To see that Doomed can be computed in time $\text{poly}(\nu)$ with oracle access to $\text{NTIME}[\text{poly}(\nu)]$, observe the following. First, given s_i and view_i for each $i \in [\nu]$, the rest of the computation of Doomed is clearly computable in time $\text{poly}(\lambda, T, \nu) \leq \text{poly}(\nu)$. Second, since s_i and view_i are the committed values of c_i and c'_i , respectively, they can be computed in time $\text{poly}(\nu)$ given oracle access to $\text{NTIME}[\text{poly}(\nu)]$.

The second part of the lemma is immediate from the definition of Doomed and the perfect binding of NIC , so we focus on the first part.

Fix any λ, T, C , and c , and assume that the commitment c cannot be opened to any m such that $C(m) = 1$. Fix any π_1 . As in Doomed , parse c and π_1 as $(c_1, \dots, c_\nu)$ and $(c'_1, \dots, c'_\nu)$, and define s_i and view_i to be the committed values of c_i and c'_i , respectively, for each $i \in [\nu]$. Also, for each $i \in [\nu]$, let s'_i be the secret input of the i -th party in view_i . (If $\text{view}_i = \perp$, then s'_i is the secret input of an arbitrary default view.) Let $s := (s_1, \dots, s_\nu)$ and $s' := (s'_1, \dots, s'_\nu)$.

The proof proceeds in three steps.

1. Show that when the commitment c cannot be opened to any m such that $C(m) = 1$, we have $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu) \neq 1 \vee |\{i \in [\nu] : s_i \neq s'_i\}| > \tau/2$, where $\tilde{\mathcal{F}}$ is defined so that $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu)$ outputs 1 if and only if all of the following hold.
 - $\text{Recon}(s'_1, \dots, s'_\nu)$ outputs some m such that $C(m) = 1$.
 - $(s'_1, \dots, s'_\nu)$ is $(\tau/2\nu)$ -close to the range of $\text{Share}(1^\nu, 1^\tau, m)$.
2. Show that when $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu) \neq 1$, we have (1).
3. Show that when $|\{i \in [\nu] : s_i \neq s'_i\}| > \tau/2$, we have (1).

Details follow.

First, we show that when the commitment c cannot be opened to any m such that $C(m) = 1$, we have $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu) \neq 1 \vee |\{i \in [\nu] : s_i \neq s'_i\}| > \tau/2$ (i.e., either $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu) \neq 1$ or s and s' are $(\tau/2\nu)$ -far). We actually prove the contrapositive: if $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu) = 1$ and s and s' are $(\tau/2\nu)$ -close, then the commitment c can be opened to some m such that $C(m) = 1$. By the definition of $\tilde{\mathcal{F}}$, when $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu) = 1$, we have the following.

- $\text{Recon}(s')$ outputs some m such that $C(m) = 1$.
- s' is $(\tau/2\nu)$ -close to the range of $\text{Share}(1^\nu, 1^\tau, m)$.

Since we assumed that s and s' are $(\tau/2\nu)$ -close, it follows that s is (τ/ν) -close to the range of $\text{Share}(1^\nu, 1^\tau, m)$. Therefore, by the robust reconstruction of Shamir's secret sharing, $\text{Recon}(s)$ outputs m . Summarizing, we have the following.

- $\text{Recon}(s)$ outputs some m such that $C(m) = 1$.

Therefore, letting d_i be a decommitment that opens c_i to s_i for each i ,¹⁷ we can use the decommitment $d = (s_1, d_1, \dots, s_\nu, d_\nu)$ to open the commitment c to some m such that $C(m) = 1$.

Next, we show that when $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu) \neq 1$, we have (1). (This step closely follows the proof in [CRT24, Appendix A].) Recall that in Doomed, a family of 4-wise independent random variables $\{X_i\}_{i \in [\nu]}$ and a set $Q := \{i \in [\nu] \mid X_i = 1\}$ are defined as in SHVZKCP.Prv_2 based on π_2 . (By definition, $\mathbb{E}[X_i] = \Pr[X_i = 1] = q/8\tau$ and $\mathbb{E}[\sum_{i \in [\nu]} X_i] = \nu \cdot q/8\tau = (3\tau + 1) \cdot q/8\tau \leq q/2$.) To show that Doomed outputs 1 with probability at least $1/2$ over uniformly random π_2 , it suffices to show that the following event occurs with probability at least $1/2$: the resulting set Q has size at most q and contains either an inconsistent pair of views or a party whose output is 0. For the former, the probability that $|Q| > q$ is at most

$$\Pr \left[\sum_{i \in [\nu]} X_i \geq 2 \cdot \mathbb{E} \left[\sum_{i \in [\nu]} X_i \right] \right] \leq \frac{1}{\nu \cdot q/8\tau} = \frac{1}{(3\tau + 1) \cdot q/8\tau} \leq \frac{8}{3q}.$$

(The first inequality follows from Fact 1.) To show the latter, consider the following *inconsistency graph* G : G has ν vertices, and there is an edge between vertex i and vertex j if and only if view_i and view_j are inconsistent.

- Case 1: the size of a minimum vertex cover of G is at most $\tau/2$. In this case, $(\text{view}_1, \dots, \text{view}_\nu)$ can be regarded as the views of an MPC execution with at most $\tau/2$ corrupted parties. Moreover, by the definition of $\tilde{\mathcal{F}}$ and the robust reconstruction of Share, the condition $\tilde{\mathcal{F}}(C, s'_1, \dots, s'_\nu) \neq 1$ implies that there is no $(s''_1, \dots, s''_\nu)$ such that $|\{i \mid s'_i \neq s''_i\}| \leq \tau/2$ and $\mathcal{F}(C, s''_1, \dots, s''_\nu) = 1$. Therefore, by the perfect τ -robustness of the MPC protocol Π , at least $\nu - \tau/2 > 5\tau/2$ views in $\{\text{view}_i\}_{i \in [\nu]}$ output 0. Therefore, the probability that none of these views is selected by Q is at most $16/5q$ by Fact 1.
- Case 2: the size of a minimum vertex cover of G is greater than $\tau/2$. In this case, by Fact 2, G has a matching M of size greater than $\tau/4$. For each edge $e = (i, j) \in M$, let $Y_e = X_i \cdot X_j$. (That is, Y_e is the indicator of the event that both i and j are selected by Q .) Thus, $\{Y_e\}_{e \in M}$ is a sequence of at least $\tau/4$ pairwise independent Bernoulli random variables with mean $\mathbb{E}[Y_e] = \mathbb{E}[X_i] \cdot \mathbb{E}[X_j] = (q/8\tau)^2$. Hence, by Fact 1, the probability that $Y_e = 0$ for every $e \in M$ is at most $256\tau/q^2$.

Therefore, when $q = \Theta(\sqrt{\tau})$ is sufficiently large, in either case the event described above occurs with probability at least $1/2$. Hence, (1) holds.

Finally, we show that when s and s' are $(\tau/2\nu)$ -far, we have (1). To show that Doomed outputs 1 with probability at least $1/2$, it suffices to show that the following event occurs with probability at least $1/2$: the set Q has size at most q and contains an index i with $s_i \neq s'_i$. For the former, the probability that $|Q| > q$ is at most $8/3q$ as above. For the latter, the probability that none of the at least $\tau/2$ indices with $s_i \neq s'_i$ is selected by Q is at most $16/q$ by Fact 1. Therefore, when $q = \Theta(\sqrt{\tau})$ is sufficiently large, (1) holds. $\square$

Finally, we show strong honest-verifier ZK. We consider a strengthened variant in which the receiver rewinds the sender a bounded number of times in a tree structure. That is, for a single commitment c , the receiver specifies multiple statements $\{C^{(\alpha)}\}_{\alpha \in [L]}$ to obtain multiple first-round prover messages $\{\pi_1^{(\alpha)}\}_{\alpha \in [L]}$, and for each $(C^{(\alpha)}, \pi_1^{(\alpha)})$, the receiver specifies multiple verifier challenge messages $\{\pi_2^{(\alpha, \beta)}\}_{\beta \in [L]}$ to obtain multiple third-round prover messages $\{\pi_3^{(\alpha, \beta)}\}_{\beta \in [L]}$. (This strengthened variant is motivated by our plan to later instantiate the Fiat-Shamir transformation using a targeted HSG in a tree structure, as sketched in Section 2.2.) We note that our variant is strong honest-verifier ZK in the sense that the cheating receiver

¹⁷Such d_i exists by the strong perfect binding of NIC.

specifies verifier challenge messages $\{\pi_2^{(\alpha,\beta)}\}_{\beta \in [L]}$ at the beginning of the experiment. (In contrast, for later convenience we allow the cheating receiver to specify statements $\{C^{(\alpha)}\}_{\alpha \in [L]}$ adaptively based on the commitment c .)

Lemma 4 (SHVZK against bounded rewinding adversaries). *For every pair of PPT Turing machines $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, the following holds. Let $L : \mathbb{N} \rightarrow \mathbb{N}$ be any function such that $L(\lambda) = O(\nu^\delta)$ for $\nu = 3 \max(\lambda, T(\lambda)) + 1$ and $\delta < 1/4$. Then, there exists a pair of PPT Turing machines $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ (called the simulator) such that the output of the following two experiments, $\text{REAL}(\lambda)$ and $\text{IDEAL}(\lambda)$, are computationally indistinguishable.*

• $\text{REAL}(\lambda)$:

1. Compute $(m, \{\pi_2^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]}, \text{st}_{\mathcal{A},1}) \leftarrow \mathcal{A}_1(1^\lambda, 1^T, 1^L)$, where $T := T(\lambda)$ and $L := L(\lambda)$.
2. Compute $(c, d) \leftarrow \text{SHVZKCP.Com}(1^\lambda, 1^T, m)$.
3. Compute $(\{C^{(\alpha)}\}_{\alpha \in [L]}, \text{st}_{\mathcal{A},2}) \leftarrow \mathcal{A}_2(\text{st}_{\mathcal{A},1}, c)$.
4. Compute $(\pi_1^{(\alpha)}, \text{st}^{(\alpha)}) \leftarrow \text{SHVZKCP.Prv}_1(1^\lambda, 1^T, C^{(\alpha)}, c, m, d)$ for each $\alpha \in [L]$.
5. Compute $\pi_3^{(\alpha,\beta)} \leftarrow \text{SHVZKCP.Prv}_2(\text{st}^{(\alpha)}, \pi_2^{(\alpha,\beta)})$ for each $\alpha, \beta \in [L]$.
6. If $|m| \neq \lambda$ or $\exists \alpha \in [L]$ s.t. $|C^{(\alpha)}| \neq T \vee C^{(\alpha)}(m) \neq 1$, output $\perp$. Otherwise, output $(\text{st}_{\mathcal{A},2}, c, \{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]})$.

• $\text{IDEAL}(\lambda)$:

1. Compute $(m, \{\pi_2^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]}, \text{st}_{\mathcal{A},1}) \leftarrow \mathcal{A}_1(1^\lambda, 1^T, 1^L)$, where $T := T(\lambda)$ and $L := L(\lambda)$.
2. Compute $(c, \text{st}_{\mathcal{S}}) \leftarrow \mathcal{S}_1(1^\lambda, 1^T, 1^L, \{\pi_2^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]})$.
3. Compute $(\{C^{(\alpha)}\}_{\alpha \in [L]}, \text{st}_{\mathcal{A},2}) \leftarrow \mathcal{A}_2(\text{st}_{\mathcal{A},1}, c)$.
4. Compute $(\{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]}) \leftarrow \mathcal{S}_2(\text{st}_{\mathcal{S}}, \{C^{(\alpha)}\}_{\alpha \in [L]})$.
5. If $|m| \neq \lambda$ or $\exists \alpha \in [L]$ s.t. $|C^{(\alpha)}| \neq T \vee C^{(\alpha)}(m) \neq 1$, output $\perp$. Otherwise, output $(\text{st}_{\mathcal{A},2}, c, \{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]})$.

Proof. Fix any pair of PPT Turing machines $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. Consider the following simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$. Let $\mathcal{S}_{\text{Share}}$ be the simulator whose existence is guaranteed by the privacy of Shamir's secret sharing, and let $\mathcal{S}_{\Pi}$ be the simulator whose existence is guaranteed by the perfect τ -privacy of the MPC protocol Π .

Simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$.

- $\mathcal{S}_1(1^\lambda, 1^T, 1^L, \{\pi_2^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]})$.
 1. For each $\alpha, \beta \in [L]$, define the set $Q^{(\alpha,\beta)}$ based on $\pi_2^{(\alpha,\beta)}$ as in SHVZKCP.Prv_2 . Also, for each $\alpha \in [L]$, let $Q^{(\alpha)} := \hat{Q}^{(\alpha,1)} \cup \dots \cup \hat{Q}^{(\alpha,L)}$, where each $\hat{Q}^{(\alpha,\beta)}$ is defined as $\hat{Q}^{(\alpha,\beta)} := Q^{(\alpha,\beta)}$ when $|Q^{(\alpha,\beta)}| \leq q$ and $\hat{Q}^{(\alpha,\beta)} := \emptyset$ otherwise. Finally, let $Q := Q^{(1)} \cup \dots \cup Q^{(L)}$.
 2. Compute $\{s_i\}_{i \in Q} \leftarrow \mathcal{S}_{\text{Share}}(1^\nu, 1^\tau, 1^\lambda, Q)$ for $\tau := \max(\lambda, T)$ and $\nu := 3\tau + 1$. Also, for each $i \in [\nu] \setminus Q$, let s_i be an arbitrary default share.
 3. Compute $(c_i, d_i) \leftarrow \text{NIC.Com}(1^\lambda, s_i)$ for each $i \in [\nu]$.
 4. Output $c := (c_1, \dots, c_\nu)$ and $\text{st}_{\mathcal{S}} := (1^\lambda, 1^T, 1^L, \{Q^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]}, s_1, d_1, \dots, s_\nu, d_\nu)$.
- $\mathcal{S}_2(\text{st}_{\mathcal{S}}, \{C^{(\alpha)}\}_{\alpha \in [L]})$.
 1. Parse $\text{st}_{\mathcal{S}}$ as $(1^\lambda, 1^T, 1^L, \{Q^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]}, s_1, d_1, \dots, s_\nu, d_\nu)$, and define $Q^{(\alpha)}$ for each $\alpha \in [L]$ as in $\mathcal{S}_1$.

2. For each $\alpha \in [L]$, compute $\pi_1^{(\alpha)}$ as follows. Compute $\{\text{view}_i^{(\alpha)}\}_{i \in Q^{(\alpha)}} \leftarrow \mathcal{S}_\Pi(Q^{(\alpha)}, C^{(\alpha)}, \{s_i\}_{i \in Q^{(\alpha)}}, 1)$. For each $i \in [\nu] \setminus Q^{(\alpha)}$, let $\text{view}_i^{(\alpha)}$ be an arbitrary default view of Π . For each $i \in [\nu]$, compute $(c_i^{(\alpha)}, d_i^{(\alpha)}) \leftarrow \text{NIC.Com}(1^\lambda, \text{view}_i^{(\alpha)})$. Let $\pi_1^{(\alpha)} := (c_1^{(\alpha)}, \dots, c_\nu^{(\alpha)})$.
3. For each $\alpha, \beta \in [L]$, compute $\pi_3^{(\alpha, \beta)}$ as follows. If $|Q^{(\alpha, \beta)}| > q$, let $\pi_3^{(\alpha, \beta)} := \perp$. Otherwise, let $\pi_3^{(\alpha, \beta)} := \{s_i, d_i, \text{view}_i^{(\alpha)}, d_i^{(\alpha)}\}_{i \in Q^{(\alpha, \beta)}}$.
4. Output $(\{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha, \beta)}\}_{\alpha, \beta \in [L]})$.

Consider the following hybrid experiments.

- Hybrid H_0 is identical to $\text{REAL}(\lambda)$.
- Hybrid H_1 differs from H_0 only in that for each $\alpha \in [L]$, $\{\text{view}_i^{(\alpha)}\}_{i \in [\nu] \setminus Q^{(\alpha)}}$ is defined as in $\mathcal{S}_2$ (i.e., as arbitrary default views of Π). The views $\{\text{view}_i^{(\alpha)}\}_{i \in [\nu] \setminus Q^{(\alpha)}}$ are used only to compute commitments $\{c_i^{(\alpha)}\}_{i \in [\nu] \setminus Q^{(\alpha)}}$, and these commitments are never opened. Thus, by the computational hiding of NIC, the output of H_1 is computationally indistinguishable from that of H_0 .
- Hybrid H_2 differs from H_1 only in that for each $\alpha \in [L]$, $\{\text{view}_i^{(\alpha)}\}_{i \in Q^{(\alpha)}}$ is defined as in $\mathcal{S}_2$ (i.e., computed using $\mathcal{S}_\Pi$). The total number of views selected by $Q^{(\alpha)}$ is at most $q \cdot L = \Theta(\sqrt{\tau}) \cdot o(\tau^{1/4}) < \tau$ for sufficiently large λ . Thus, by the perfect τ -privacy of Π , the output of H_2 is identically distributed as that of H_1 .
- Hybrid H_3 differs from H_2 only in that $\{s_i\}_{i \in [\nu] \setminus Q}$ is defined as in $\mathcal{S}_1$ (i.e., as arbitrary default shares). The shares $\{s_i\}_{i \in [\nu] \setminus Q}$ are used only to compute commitments $\{c_i\}_{i \in [\nu] \setminus Q}$, and these commitments are never opened. Thus, by the computational hiding of NIC, the output of H_3 is computationally indistinguishable from that of H_2 .
- Hybrid H_4 differs from H_3 only in that $\{s_i\}_{i \in Q}$ is defined as in $\mathcal{S}_1$ (i.e., computed using $\mathcal{S}_{\text{Share}}(1^\nu, 1^\tau, 1^\lambda, Q)$). The total number of shares selected by Q is at most $q \cdot L^2 = \Theta(\sqrt{\tau}) \cdot o(\tau^{1/2}) < \tau$ for sufficiently large λ . Thus, by the privacy of Shamir's secret sharing, the output of H_4 is identically distributed as that of H_3 .
- Hybrid H_5 is identical to $\text{IDEAL}(\lambda)$. By the construction of $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, the output of H_5 is identically distributed as that of H_4 .

□

4.2 Step 2. Interactive WI Commit and Prove

Using $\text{SHVZKCP} = (\text{SHVZKCP.Com}, \text{SHVZKCP.Rec}, \text{SHVZKCP.Prv}_1, \text{SHVZKCP.Vrfy}_1, \text{SHVZKCP.Prv}_2, \text{SHVZKCP.Vrfy}_2)$ constructed above, we define a commit-and-prove scheme $\text{WICP} = (\text{WICP.Com}, \text{WICP.Rec}, \text{WICP.Vrfy}_0, \text{WICP.Prv}_1, \text{WICP.Vrfy}_1, \text{WICP.Prv}_2, \text{WICP.Vrfy}_2)$ as follows. At a high level, WICP is obtained by running several instances of SHVZKCP in parallel, as described in [Section 2.1](#). For notational simplicity, we assume throughout this step that, when run with security parameter λ , SHVZKCP can commit to messages of length $\lambda_{\text{SHVZKCP}} := \lambda(B+1)$, where $B := \max(\lambda^{1/\epsilon}, T)$.¹⁸ (Recall that ϵ is the parameter of HSG as stated in [Theorem 3](#).)

¹⁸This can be achieved by internally instantiating the construction from [Section 4.1](#) with security parameter λ_{SHVZKCP} . We nevertheless continue to write 1^λ as the first input to the SHVZKCP algorithms below.

Witness-indistinguishable commit-and-prove WICP.

- $\text{WICP.Com}(1^\lambda, 1^T, m)$.

If $|m| \neq \lambda$, abort. Otherwise, do the following.

1. Let $\mathbb{F}$ be a finite field such that $|\mathbb{F}| = 2^\lambda$, and view m as an element of $\mathbb{F}$. Let $\tau' = O(\log(\lambda + T))$ be a sufficiently large positive integer,^a and let $\nu' := 3\tau' + 1$. Let $T' = \text{poly}(\lambda, T)$ denote the size of the circuit C' defined later in WICP.Priv_1 ; under the fixed construction of C' , T' is uniquely determined by λ and T . Let $B := \max(\lambda^{1/\epsilon}, T)$.
2. Sample $r_1, \dots, r_B \in \mathbb{F}$ uniformly at random.
3. For each $i \in [\nu']$, compute $(c_i, d_i) \leftarrow \text{SHVZKCP.Com}(1^\lambda, 1^{T'}, \hat{m}_i)$, where $\hat{m}_i := m \parallel r_1 \parallel \dots \parallel r_B$ is the concatenation of the binary representations of m and $r_1, \dots, r_B$.
4. Output $c := (c_1, \dots, c_{\nu'})$ and $d := (\hat{m}_1, d_1, \dots, \hat{m}_{\nu'}, d_{\nu'})$.

- $\text{WICP.Rec}(1^\lambda, 1^T, c, m, d)$.

Parse c and d as $(c_1, \dots, c_{\nu'})$ and $(\hat{m}_1, d_1, \dots, \hat{m}_{\nu'}, d_{\nu'})$, and let k be any index such that $\hat{m}_k$ is the most frequent value among $(\hat{m}_1, \dots, \hat{m}_{\nu'}) \in (\{0, 1\}^{\lambda(B+1)} \cup \{\perp\})^{\nu'}$ (with ties broken canonically).^b Then, output 1 if and only if all of the following hold.

- $\text{SHVZKCP.Rec}(1^\lambda, 1^{T'}, c_i, \hat{m}_i, d_i) = 1$ for every $i \in [\nu']$.
- There exist $r_1, \dots, r_B \in \mathbb{F}$ such that $\hat{m}_k = m \parallel r_1 \parallel \dots \parallel r_B$.

- $\text{WICP.Vrfy}_0(1^\lambda, 1^T)$.

1. Sample $u \in \mathbb{F} \setminus \{0\}$ uniformly at random.
2. Output $\pi_0 := u$.

- $\text{WICP.Priv}_1(1^\lambda, 1^T, C, c, m, d, \pi_0)$.

If $|m| \neq \lambda$ or $|C| \neq T$, abort. Otherwise, do the following.

1. Parse c , d , and π_0 as $(c_1, \dots, c_{\nu'})$, $(\hat{m}_1, d_1, \dots, \hat{m}_{\nu'}, d_{\nu'})$, and u , respectively, and abort if $u \notin \mathbb{F} \setminus \{0\}$. Check that for some $r_1, \dots, r_B \in \mathbb{F}$ each $\hat{m}_i$ satisfies $\hat{m}_i = m \parallel r_1 \parallel \dots \parallel r_B$, and define the degree- B polynomial $f : \mathbb{F} \rightarrow \mathbb{F}$ by $f(x) = r_B x^B + \dots + r_1 x + m$.
2. Sample $\sigma \in \{0, 1\}$ uniformly at random, let $v_\sigma := f(u) \in \mathbb{F}$, sample $v_{1-\sigma} \in \mathbb{F}$ uniformly at random, and define C' as the following circuit.
 - C' is given (u, v_0, v_1) as hardwired input. Then, given $m \parallel r_1 \parallel \dots \parallel r_B$ as input, C' outputs $C(m)$ if the degree- B polynomial $f(x) = r_B x^B + \dots + r_1 x + m$ satisfies $f(u) \in \{v_0, v_1\}$, and outputs 0 otherwise.

By the definition of T' , we have $|C'| = T'$.

3. For each $i \in [\nu']$, compute $(\pi_{1,i}, \text{st}_i) \leftarrow \text{SHVZKCP.Priv}_1(1^\lambda, 1^{T'}, C', c_i, \hat{m}_i, d_i)$.
4. Output $\pi_1 := (v_0, v_1, \pi_{1,1}, \dots, \pi_{1,\nu'})$ and $\text{st} := (\text{st}_1, \dots, \text{st}_{\nu'})$.

- $\text{WICP.Vrfy}_1(1^\lambda, 1^T)$.

1. Choose $\pi_{2,1}, \dots, \pi_{2,\nu'} \in \{0, 1\}^\ell$ uniformly at random, where $\ell = O(\log(\lambda + T))$ is the output length of $\text{SHVZKCP.Vrfy}_1(1^\lambda, 1^{T'})$.
2. Output $\pi_2 := (\pi_{2,1}, \dots, \pi_{2,\nu'})$.

- $\text{WICP.Priv}_2(\text{st}, \pi_2)$.

1. Parse st and π_2 as $(\text{st}_1, \dots, \text{st}_{\nu'})$ and $(\pi_{2,1}, \dots, \pi_{2,\nu'})$, respectively.
 2. Choose $\Delta \in \{0, 1\}^\ell$ uniformly at random, and for each $i \in [\nu']$ let $\hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta$.
 3. For each $i \in [\nu']$, compute $\pi_{3,i} \leftarrow \text{SHVZKCP.Prv}_2(\text{st}_i, \hat{\pi}_{2,i})$.
 4. Output $\pi_3 := (\Delta, \pi_{3,1}, \dots, \pi_{3,\nu'})$.
- $\text{WICP.Vrfy}_2(1^\lambda, 1^T, C, c, \pi_0, \pi_1, \pi_2, \pi_3)$.
- If $|C| \neq T$, output 0. Otherwise, do the following.
1. Parse c, π_0, π_1, π_2 , and π_3 as $(c_1, \dots, c_{\nu'}), u, (v_0, v_1, \pi_{1,1}, \dots, \pi_{1,\nu'}), (\pi_{2,1}, \dots, \pi_{2,\nu'}),$ and $(\Delta, \pi_{3,1}, \dots, \pi_{3,\nu'})$, respectively. Define C' as in WICP.Prv_1 with (u, v_0, v_1) hardwired, and let $T' := |C'|$. Also, compute $(\hat{\pi}_{2,1}, \dots, \hat{\pi}_{2,\nu'})$ as in WICP.Prv_2 .
 2. Output 1 if and only if, for every $i \in [\nu']$, $\text{SHVZKCP.Vrfy}_2(1^\lambda, 1^{T'}, C', c_i, \pi_{1,i}, \hat{\pi}_{2,i}, \pi_{3,i}) = 1$.

^aFor the required size of τ' , see the proof of Lemma 7.

^bNote that $\hat{m}_k$ may be $\perp$.

It is immediate from the construction that WICP makes only black-box use of SHVZKCP. Since SHVZKCP in turn makes only black-box use of NIC, it follows that WICP also makes only black-box use of NIC. We now establish the required properties of WICP, beginning with correctness.

Lemma 5.

- **Commitment correctness.** For every $\lambda \in \mathbb{N}, T \in \mathbb{N}$, and $m \in \{0, 1\}^\lambda$,

$$\Pr \left[\text{WICP.Rec}(1^\lambda, 1^T, c, m, d) = 1 \mid (c, d) \leftarrow \text{WICP.Com}(1^\lambda, 1^T, m) \right] = 1.$$

- **Proof correctness.** For every $\lambda \in \mathbb{N}, T \in \mathbb{N}, m \in \{0, 1\}^\lambda$, and a circuit C such that $|C| = T$ and $C(m) = 1$,

$$\Pr \left[\text{WICP.Vrfy}_2(1^\lambda, 1^T, C, c, \pi_0, \pi_1, \pi_2, \pi_3) = 1 \mid \begin{array}{l} (c, d) \leftarrow \text{WICP.Com}(1^\lambda, 1^T, m) \\ \pi_0 \leftarrow \text{WICP.Vrfy}_0(1^\lambda, 1^T) \\ (\pi_1, \text{st}) \leftarrow \text{WICP.Prv}_1(1^\lambda, 1^T, C, c, m, d, \pi_0) \\ \pi_2 \leftarrow \text{WICP.Vrfy}_1(1^\lambda, 1^T) \\ \pi_3 \leftarrow \text{WICP.Prv}_2(\text{st}, \pi_2) \end{array} \right] = 1.$$

Moreover, WICP.Vrfy_0 and WICP.Vrfy_1 are public-coin algorithms, and $|\pi_0| = \lambda$ and $|\pi_2| = O((\log(\lambda + T))^2)$. Also, there exists a universal constant $\eta \geq 1$ (independent of ϵ, λ , and T) such that

$$\lambda + |C| + |c| + |\pi_0| + |\pi_1| \leq B^\eta \quad (2)$$

for $B := \max(\lambda^{1/\epsilon}, T)$.

Proof. Commitment correctness follows directly from the commitment correctness of SHVZKCP. Proof correctness follows from the perfect proof correctness of SHVZKCP and the fact that SHVZKCP.Vrfy_1 is a public-coin algorithm. Moreover, by construction, $|\pi_0| = \lambda$, and $|\pi_2| = |\pi_{2,1}| + \dots + |\pi_{2,\nu'}| = \nu' \cdot O(\log(\lambda + T)) = O((\log(\lambda + T))^2)$. Furthermore, since $|C'| = \text{poly}(\lambda, T, B)$, we have $|\pi_1| = \text{poly}(\lambda, T, B, \nu')$, which implies (2). $\square$

Next, we show perfect binding and computational hiding.

Lemma 6.

- **Perfect binding.** For any $\lambda \in \mathbb{N}, T \in \mathbb{N}, c \in \{0, 1\}^*$, $(m_0, m_1) \in (\{0, 1\}^\lambda)^2$ such that $m_0 \neq m_1$, there is no $(d_0, d_1) \in (\{0, 1\}^*)^2$ satisfying $\text{WICP.Rec}(1^\lambda, 1^T, c, m_0, d_0) = 1 \wedge \text{WICP.Rec}(1^\lambda, 1^T, c, m_1, d_1) = 1$.

- **Computational hiding.** For every pair of PPT Turing machines $(\mathcal{A}_1, \mathcal{A}_2)$, every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, and every $\lambda \in \mathbb{N}$,

$$\Pr \left[\begin{array}{l} |m_0| = |m_1| \\ \wedge \mathcal{A}_2(\text{st}_{\mathcal{A}}, c) = b \end{array} \mid \begin{array}{l} (m_0, m_1, \text{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, 1^{T(\lambda)}) \\ b \leftarrow \{0, 1\} \\ (c, d) \leftarrow \text{WICP.Com}(1^\lambda, 1^{T(\lambda)}, m_b) \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda).$$

Proof. Perfect binding follows directly from the strong perfect binding of SHVZKCP. Indeed, suppose there exist $d_0 = (\hat{m}_{0,1}, d_{0,1}, \dots, \hat{m}_{0,\nu'}, d_{0,\nu'})$ and $d_1 = (\hat{m}_{1,1}, d_{1,1}, \dots, \hat{m}_{1,\nu'}, d_{1,\nu'})$ such that $\text{SHVZKCP.Rec}(1^\lambda, 1^{T'}, c_i, \hat{m}_{b,i}, d_{b,i}) = 1$ for every $b \in \{0, 1\}$ and $i \in [\nu']$. Then, by the strong perfect binding of SHVZKCP, we have $\hat{m}_{0,i} = \hat{m}_{1,i}$ for every $i \in [\nu']$, and hence the most frequent value among $(\hat{m}_{0,1}, \dots, \hat{m}_{0,\nu'})$ and $(\hat{m}_{1,1}, \dots, \hat{m}_{1,\nu'})$ is identical. Since any valid opening to m_b requires this most frequent value to have prefix m_b , this implies $m_0 = m_1$.

Computational hiding follows directly from the computational hiding of SHVZKCP. $\square$

Next, we show soundness. As before, we consider round-by-round soundness.

Lemma 7 (Round-by-round soundness). *There exist Boolean functions $\text{Doomed}_1, \text{Doomed}_2$ such that the following hold.*

1. For any λ, T, C , and c such that $T = \text{poly}(\lambda)$, if the commitment c cannot be opened to any m such that $C(m) = 1$, (i.e., if for every m such that $C(m) = 1$ and every d , $\text{WICP.Rec}(1^\lambda, 1^T, c, m, d) \neq 1$), then

$$\Pr \left[\text{Doomed}_1(\lambda, T, C, c, \pi_0) = 1 \mid \pi_0 \leftarrow \text{WICP.Vrfy}_0(1^\lambda, 1^T) \right] \geq \frac{1}{2}. \quad (3)$$

2. For any λ, T, C, c , and π_0 such that $T = \text{poly}(\lambda)$, if $\text{Doomed}_1(\lambda, T, C, c, \pi_0) = 1$, then

$$\forall \pi_1 : \Pr \left[\text{Doomed}_2(\lambda, T, C, c, \pi_0, \pi_1, \pi_2) = 1 \mid \pi_2 \leftarrow \text{WICP.Vrfy}_1(1^\lambda, 1^T) \right] \geq \frac{1}{2}. \quad (4)$$

3. For any $\lambda, T, C, c, \pi_0, \pi_1$, and π_2 , such that $T = \text{poly}(\lambda)$, if $\text{Doomed}_2(\lambda, T, C, c, \pi_0, \pi_1, \pi_2) = 1$, then

$$\forall \pi_3 : \text{WICP.Vrfy}_2(1^\lambda, 1^T, C, c, \pi_0, \pi_1, \pi_2, \pi_3) = 0. \quad (5)$$

Furthermore, Doomed_1 and Doomed_2 can be computed in time $\text{poly}(B)$ with oracle access to $\text{NTIME}[\text{poly}(B)]$.

Proof. We define Doomed_1 and Doomed_2 as follows. Let $\text{Doomed}_{\text{SHVZKCP}}$ be the Boolean function whose existence is guaranteed by the round-by-round soundness of SHVZKCP (Lemma 3).

Doomed₁ and Doom₂.

- $\text{Doomed}_1(\lambda, T, C, c, \pi_0)$.
 1. Parse c and π_0 as $(c_1, \dots, c_{\nu'})$ and u , respectively. Let the committed values of $(c_1, \dots, c_{\nu'})$ be $\hat{m}_1, \dots, \hat{m}_{\nu'} \in \{0, 1\}^{\lambda(B+1)} \cup \{\perp\}$, and let $k \in [\nu']$ be any index such that $\hat{m}_k$ is the most frequent value among $(\hat{m}_1, \dots, \hat{m}_{\nu'})$ (with ties broken canonically). Also, parse each $\hat{m}_i$ as $m_i \parallel r_{i,1} \parallel \dots \parallel r_{i,B}$ and define the degree- B polynomial $f_i : \mathbb{F} \rightarrow \mathbb{F}$ by $f_i(x) = r_{i,B}x^B + \dots + r_{i,1}x + m_i$.^a
 2. Output 1 if and only if all of the following hold.
 - $C(m_k) \neq 1$.
 - For every $i, j \in [\nu']$, if $\hat{m}_i \neq \hat{m}_j$, then $f_i(u) \neq f_j(u)$.
- $\text{Doomed}_2(\lambda, T, C, c, \pi_0, \pi_1, \pi_2)$.

1. Parse c , π_0 , π_1 , and π_2 as $(c_1, \dots, c_{\nu'})$, u , $(v_0, v_1, \pi_{1,1}, \dots, \pi_{1,\nu'})$, and $(\pi_{2,1}, \dots, \pi_{2,\nu'})$, respectively. Define C' as in WICP.Prv_1 with (u, v_0, v_1) hardwired, and let $T' := |C'|$.
2. Output 1 if and only if the following holds: for every $\Delta \in \{0, 1\}^\ell$, there exists $j \in [\nu']$ such that $\hat{\pi}_{2,j} := \pi_{2,j} \oplus \Delta$ satisfies $\text{Doomed}_{\text{SHVZKCP}}(\lambda, T', C', c_j, \pi_{1,j}, \hat{\pi}_{2,j}) = 1$.

^aIf $\hat{m}_i = \perp$, let $m_i := \perp$ and $f_i := \perp$. (In this case, we view f_i as a function that outputs $\perp$ on all inputs.)

To see that Doomed_1 and Doomed_2 can be computed in time $\text{poly}(B)$ with oracle access to $\text{NTIME}[\text{poly}(B)]$, observe the following. For Doomed_1 , given $\hat{m}_i$ for each $i \in [\nu']$, the remaining computation is clearly computable in time $\text{poly}(\lambda, T, B, \nu') \leq \text{poly}(B)$. Moreover, since each $\hat{m}_i$ is the committed value of c_i , it can be computed in time $\text{poly}(B)$ given oracle access to $\text{NTIME}[\text{poly}(B)]$. For Doomed_2 , aside from computing $\text{Doomed}_{\text{SHVZKCP}}$, the computation is clearly computable in time $\text{poly}(2^\ell, \nu') = \text{poly}(\lambda, T, \nu') \leq \text{poly}(B)$. Moreover, $\text{Doomed}_{\text{SHVZKCP}}$ can be computed in time $\text{poly}(\nu)$ with oracle access to $\text{NTIME}[\text{poly}(\nu)]$, where $\nu = 3 \max(\lambda_{\text{SHVZKCP}}, T') + 1 = \text{poly}(\lambda, T, B) \leq \text{poly}(B)$.

First part. First, we show (3). Fix any λ , T , C , and c , and assume that the commitment c cannot be opened to any m such that $C(m) = 1$. As in Doomed_1 , define $\hat{m}_1, \dots, \hat{m}_{\nu'}$ and k , parse each $\hat{m}_i$ as $m_i \parallel r_{i,1} \parallel \dots \parallel r_{i,B}$, and define f_i . The goal is to show that, for uniformly random $u \in \mathbb{F} \setminus \{0\}$, the following event occurs with probability at least $1/2$.

- $C(m_k) \neq 1$.
- For every $i, j \in [\nu']$, if $\hat{m}_i \neq \hat{m}_j$, then $f_i(u) \neq f_j(u)$.

From the construction of WICP.Rec , when the commitment c cannot be opened to any m such that $C(m) = 1$, we always have $C(m_k) \neq 1$.¹⁹ Also, for any $i, j \in [\nu']$, if $\hat{m}_i \neq \hat{m}_j$, then f_i and f_j are distinct degree- B polynomials (or $\perp$), so

$$\Pr [\exists i, j \in [\nu'] \text{ s.t. } \hat{m}_i \neq \hat{m}_j \wedge f_i(u) = f_j(u) \mid u \leftarrow \mathbb{F} \setminus \{0\}] \leq \nu'^2 \cdot \frac{B}{|\mathbb{F}| - 1} = \nu'^2 \cdot \frac{\max(\lambda^{1/\epsilon}, T)}{2^\lambda - 1}.$$

Since $T = \text{poly}(\lambda)$ and $\nu' = O(\log(\lambda + T))$, this upper bound is at most $1/2$ for sufficiently large λ . Therefore, we have (3).

Second part. Next we show (4). Fix any λ , T , C , c , and π_0 such that $\text{Doomed}_1(\lambda, T, C, c, \pi_0) = 1$, and also fix π_1 . Parse π_0 and π_1 as u and $(v_0, v_1, \pi_{1,1}, \dots, \pi_{1,\nu'})$, respectively. Define C' as in WICP.Prv_1 with (u, v_0, v_1) hardwired, and let $T' := |C'|$. As in Doomed_1 , define $\hat{m}_1, \dots, \hat{m}_{\nu'}$ and k , parse each $\hat{m}_i$ as $m_i \parallel r_{i,1} \parallel \dots \parallel r_{i,B}$, and define f_i .

As a preliminary step, we show that when $\text{Doomed}_1(\lambda, T, C, c, \pi_0) = 1$, the index set I^* defined as $I^* := \{i \in [\nu'] \mid C'(\hat{m}_i) \neq 1\}$ satisfies $|I^*| \geq \tau' + 1$. (Intuitively, I^* is the set of indices where the statement proven by SHVZKCP is false.) By the definition of Doomed_1 , if $\text{Doomed}_1(\lambda, T, C, c, \pi_0) = 1$ then both of the following hold.

- $C(m_k) \neq 1$ (i.e., most frequent value does not satisfy C).
- For every $i, j \in [\nu']$, if $\hat{m}_i \neq \hat{m}_j$, then $f_i(u) \neq f_j(u)$ (i.e., the map $\hat{m}_i \mapsto f_i(u)$ is injective).

To show that $|I^*| \geq \tau' + 1$, we assume $|I^*| \leq \tau'$ and derive a contradiction. By the definition of I^* , we have $C'(\hat{m}_i) = 1$ for every $i \in [\nu'] \setminus I^*$. Hence, by the definition of C' , we have $C(m_i) = 1 \wedge f_i(u) \in \{v_0, v_1\}$ for every $i \in [\nu'] \setminus I^*$. Since the map $\hat{m}_i \mapsto f_i(u)$ is injective, it follows that

¹⁹This is because if $C(m_k) = 1$, then by opening $c = (c_1, \dots, c_{\nu'})$ to $(\hat{m}_1, \dots, \hat{m}_{\nu'})$, we can open c to some m such that $C(m) = 1$.

there exist $\hat{m}^\dagger, \hat{m}^\ddagger$ such that we have $C(m_i) = 1 \wedge \hat{m}_i \in \{\hat{m}^\dagger, \hat{m}^\ddagger\}$ for every $i \in [\nu'] \setminus I^*$. Since $|\nu' \setminus I^*| \geq \nu' - \tau' = 2\tau' + 1$, by an averaging argument there exists $\hat{m} \in \{\hat{m}^\dagger, \hat{m}^\ddagger\}$ such that $|\{i \in [\nu'] \setminus I^* \mid \hat{m}_i = \hat{m}\}| \geq \tau' + 1$. Thus, the most frequent value $\hat{m}_k$ satisfies $\hat{m}_k \in \{\hat{m}^\dagger, \hat{m}^\ddagger\}$,²⁰ and therefore $C(m_k) = 1$, a contradiction.

With this in hand, we show (4) using $|I^*| \geq \tau' + 1$. From the definition of Doomed_2 , our goal is to show that the following event occurs with probability at least $1/2$ over the choice of $\{\pi_{2,i}\}_{i \in [\nu']}$: for every $\Delta \in \{0, 1\}^\ell$, there exists $j \in [\nu']$ such that $\hat{\pi}_{2,j} := \pi_{2,j} \oplus \Delta$ satisfies $\text{Doomed}_{\text{SHVZKCP}}(\lambda, T', C', c_j, \pi_{1,j}, \hat{\pi}_{2,j}) = 1$. We consider a union bound over $\Delta \in \{0, 1\}^\ell$. Fix any $\Delta \in \{0, 1\}^\ell$, and let $\text{EVENT}(\Delta)$ be the event that $\hat{\pi}_{2,j} := \pi_{2,j} \oplus \Delta$ satisfies $\text{Doomed}_{\text{SHVZKCP}}(\lambda, T', C', c_j, \pi_{1,j}, \hat{\pi}_{2,j}) = 0$ for every $j \in I^*$. Note that, by the definition of I^* , for each $j \in I^*$ the statement proven by SHVZKCP is false (i.e., the commitment c_j cannot be opened to any $\hat{m}_j$ such that $C'(\hat{m}_j) = 1$). Then, since $\{\hat{\pi}_{2,i}\}_{i \in [\nu']} = \{\pi_{2,i} \oplus \Delta\}_{i \in [\nu']}$ is uniformly distributed when $\{\pi_{2,i}\}_{i \in [\nu]}$ is uniformly distributed, we have the following from the round-by-round soundness of SHVZKCP.

$$\Pr \left[\text{EVENT}(\Delta) \mid \{\pi_{2,i}\}_{i \in [\nu]} \leftarrow (\{0, 1\}^\ell)^{\nu'} \right] \leq 2^{-(\tau'+1)}.$$

Since $\ell = O(\log(\lambda + T))$, it follows that when $\tau' = O(\log(\lambda + T))$ is sufficiently large, the following follows from a union bound.

$$\Pr \left[\exists \Delta \in \{0, 1\}^\ell \text{ s.t. } \text{EVENT}(\Delta) \mid \{\pi_{2,i}\}_{i \in [\nu]} \leftarrow (\{0, 1\}^\ell)^{\nu'} \right] \leq 2^\ell \cdot 2^{-(\tau'+1)} \leq \frac{1}{2}.$$

Therefore (4) holds.

Third part. Finally, we show (5). This follows immediately from the construction of WICP.Vrfy_2 and the round-by-round soundness of SHVZKCP. $\square$

Finally, we show witness indistinguishability. As before, we consider a strengthened variant in which the receiver rewinds the sender a bounded number of times in a tree structure. Below, the upper bound on the number of rewindings is carefully chosen so that later we can use the Fiat-Shamir transformation with the targeted HSG assumed in [Theorem 3](#).

Lemma 8 (WI against bounded rewinding adversaries). *For every tuple of PPT Turing machines $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4)$ and every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, the following holds. Let $\eta \geq 1$ be any constant that satisfies (2) in [Lemma 5](#), let $N : \mathbb{N} \rightarrow \mathbb{N}$ be the function defined as $N(\lambda) = B^\eta$ for $B := \max(\lambda^{1/\epsilon}, T(\lambda))$, and let $L : \mathbb{N} \rightarrow \mathbb{N}$ be a function such that $L(\lambda) = \text{poly}(N(\lambda)^\epsilon)$. Then, when ϵ is sufficiently small, for every λ and for $T := T(\lambda)$ and $L := L(\lambda)$, it holds*

$$\Pr \left[\begin{array}{l} |m_0| = |m_1| \\ \wedge C(m_0) = C(m_1) = 1 \\ \wedge \forall \alpha, \alpha' \in [L] \text{ with } \alpha \neq \alpha' : \pi_0^{(\alpha)} \neq \pi_0^{(\alpha')} \\ \wedge \mathcal{A}_4(\text{st}_{\mathcal{A},3}, \{\pi_3^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]}) = b \end{array} \mid \begin{array}{l} (m_0, m_1, \text{st}_{\mathcal{A},1}) \leftarrow \mathcal{A}_1(1^\lambda, 1^T, 1^L) \\ b \leftarrow \{0, 1\} \\ (c, d) \leftarrow \text{WICP.Com}(1^\lambda, 1^T, m_b) \\ (C, \{\pi_0^{(\alpha)}\}_{\alpha \in [L]}, \text{st}_{\mathcal{A},2}) \leftarrow \mathcal{A}_2(\text{st}_{\mathcal{A},1}, c) \\ \forall \alpha \in [L] : (\pi_1^{(\alpha)}, \text{st}^{(\alpha)}) \leftarrow \text{WICP.Priv}_1(1^\lambda, 1^T, C, c, m_b, d, \pi_0^{(\alpha)}) \\ (\{\pi_2^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]}, \text{st}_{\mathcal{A},3}) \leftarrow \mathcal{A}_3(\text{st}_{\mathcal{A},2}, \{\pi_1^{(\alpha)}\}_{\alpha \in [L]}) \\ \forall \alpha, \beta \in [L] : \pi_3^{(\alpha,\beta)} \leftarrow \text{WICP.Priv}_2(\text{st}^{(\alpha)}, \pi_2^{(\alpha,\beta)}) \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda).$$

Proof. As a warm-up, we first consider the case $L = 1$. Without loss of generality, we assume that $\mathcal{A}_1$ always outputs m_0, m_1 such that $|m_0| = |m_1| = \lambda \wedge m_0 \neq m_1$, and that $\mathcal{A}_2$ always outputs $(C, \{\pi_0^{(\alpha)}\}_{\alpha \in [L]})$ such that $|C| = T \wedge C(m_0) = C(m_1) = 1$ and $\pi_0^{(\alpha)} \in \mathbb{F} \setminus \{0\}$ for every $\alpha \in [L]$. In this case, our goal is to bound the following probability. (For later convenience, we relabel m_0, m_1

²⁰Note that if $|I^*| \leq \tau'$, then $|\{\hat{m}_1, \dots, \hat{m}_{\nu'}\} \setminus \{\hat{m}^\dagger, \hat{m}^\ddagger\}| \leq \tau'$.

as m, m' in the following.)

$$\Pr \left[\begin{array}{l} |m| = |m'| \\ \wedge C(m) = C(m') = 1 \\ \wedge \mathcal{A}_4(\text{st}_{\mathcal{A},3}, \pi_3) = m^* \end{array} \middle| \begin{array}{l} (m, m', \text{st}_{\mathcal{A},1}) \leftarrow \mathcal{A}_1(1^\lambda, 1^T, 1^L) \\ m^* \leftarrow \{m, m'\} \\ (c, d) \leftarrow \text{WICP.Com}(1^\lambda, 1^T, m^*) \\ (C, \pi_0, \text{st}_{\mathcal{A},2}) \leftarrow \mathcal{A}_2(\text{st}_{\mathcal{A},1}, c) \\ (\pi_1, \text{st}) \leftarrow \text{WICP.Priv}_1(1^\lambda, 1^T, C, c, m^*, d, \pi_0) \\ (\pi_2, \text{st}_{\mathcal{A},3}) \leftarrow \mathcal{A}_3(\text{st}_{\mathcal{A},2}, \pi_1) \\ \pi_3 \leftarrow \text{WICP.Priv}_2(\text{st}, \pi_2) \end{array} \right]. \quad (6)$$

We bound the probability in (6) via hybrid experiments $H_0, \dots, H_{4(\nu'-1)+6}$ defined below. Let $\text{SHVZKCP.S} = (\text{SHVZKCP.S}_1, \text{SHVZKCP.S}_2)$ be the simulator whose existence is guaranteed by the strong honest-verifier zero-knowledge of SHVZKCP (for the case $L = 1$). We begin by defining H_0 and H_1 .

- Hybrid H_0 is the experiment in (6) with $m^* = m$. Inlining WICP.Com , WICP.Priv_1 , and WICP.Priv_2 , we can describe H_0 as follows.
 1. Compute $(m, m', \text{st}_{\mathcal{A},1}) \leftarrow \mathcal{A}_1(1^\lambda, 1^T, 1^L)$.
 2. Compute c as follows: sample $(r_1, \dots, r_B) \leftarrow \mathbb{F}^B$; for each $i \in [\nu']$, let $\hat{m}_i := m \parallel r_1 \parallel \dots \parallel r_B$ and compute $(c_i, d_i) \leftarrow \text{SHVZKCP.Com}(1^\lambda, 1^T, \hat{m}_i)$; let $c := (c_1, \dots, c_{\nu'})$.
 3. Compute $(C, u, \text{st}_{\mathcal{A},2}) \leftarrow \mathcal{A}_2(\text{st}_{\mathcal{A},1}, c)$.
 4. Compute π_1 as follows: sample $\sigma \leftarrow \{0, 1\}$ and let $v_\sigma := f(u)$ and $v_{1-\sigma} \leftarrow \mathbb{F}$ using the polynomial $f(x) = r_B x^B + \dots + r_1 x + m$; define C' based on (u, v_0, v_1) ; compute $(\pi_{1,i}, \text{st}_i) \leftarrow \text{SHVZKCP.Priv}_1(1^\lambda, 1^T, C', c_i, \hat{m}_i, d_i)$ for each $i \in [\nu']$; let $\pi_1 := (v_0, v_1, \pi_{1,1}, \dots, \pi_{1,\nu'})$.
 5. Compute $((\pi_{2,1}, \dots, \pi_{2,\nu'}), \text{st}_{\mathcal{A},3}) \leftarrow \mathcal{A}_3(\text{st}_{\mathcal{A},2}, \pi_1)$.
 6. Compute π_3 as follows: sample $\Delta \leftarrow \{0, 1\}^\ell$; let $\hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta$ for each $i \in [\nu']$; compute $\pi_{3,i} \leftarrow \text{SHVZKCP.Priv}_2(\text{st}_i, \hat{\pi}_{2,i})$ for each $i \in [\nu']$; let $\pi_3 := (\Delta, \pi_{3,1}, \dots, \pi_{3,\nu'})$.
 7. Compute $\text{out} \leftarrow \mathcal{A}_4(\text{st}_{\mathcal{A},3}, \pi_3)$.

The output of H_0 is $\mathcal{A}_4$'s output out.

- Hybrid H_1 differs from H_0 only in the following ways.
 - In Step 2, additionally sample $(r'_1, \dots, r'_B) \leftarrow \mathbb{F}^B$.
 - In Step 4, instead of setting $v_{1-\sigma} \leftarrow \mathbb{F}$, let $v_{1-\sigma} := f'(u)$ using the polynomial $f'(x) = r'_B x^B + \dots + r'_1 x + m'$.

The output of H_1 is identically distributed as that of H_0 . This is because when $u \neq 0$, the value $f'(u) = r'_B u^B + \dots + r'_1 u + m'$ for uniformly chosen $(r'_1, \dots, r'_B)$ is uniformly distributed in the adversary's view.

Next, for each $k \in [\nu']$, we define $H_{4(k-1)+2}$, $H_{4(k-1)+3}$, $H_{4(k-1)+4}$, and $H_{4(k-1)+5}$.

- For each $k \in [\nu']$, Hybrid $H_{4(k-1)+2}$ differs from $H_{4(k-1)+1}$ only in the following ways.
 - In Step 2, additionally sample $\hat{\pi}_{2,k} \leftarrow \{0, 1\}^\ell$.
 - In Step 6, instead of sampling $\Delta \leftarrow \{0, 1\}^\ell$ and setting $\hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta$ for each $i \in [\nu']$, let $\Delta := \pi_{2,k} \oplus \hat{\pi}_{2,k}$ and set $\hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta$ for each $i \in [\nu'] \setminus \{k\}$.

The output of $H_{4(k-1)+2}$ is identically distributed as that of $H_{4(k-1)+1}$. This is because for any $\{\pi_{2,i}\}_{i \in [\nu']}$, the distributions

$$\left\{ (\{\hat{\pi}_{2,i}\}_{i \in [\nu]}, \Delta) \mid \begin{array}{l} \Delta \leftarrow \{0, 1\}^\ell \\ \forall i \in [\nu] : \hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta \end{array} \right\}$$

and

$$\left\{ (\{\hat{\pi}_{2,i}\}_{i \in [\nu']}, \Delta) \mid \begin{array}{l} \hat{\pi}_{2,k} \leftarrow \{0, 1\}^\ell \\ \Delta := \pi_{2,k} \oplus \hat{\pi}_{2,k} \\ \forall i \in [\nu'] \setminus \{k\} : \hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta \end{array} \right\}$$

are identically distributed.

- For each $k \in [\nu']$, Hybrid $H_{4(k-1)+3}$ differs from $H_{4(k-1)+2}$ only in the following ways.
 - In Step 2, instead of computing $(c_k, d_k) \leftarrow \text{SHVZKCP.Com}(1^\lambda, 1^{T'}, \hat{m}_k)$, compute $(c_k, \text{st}_S) \leftarrow \text{SHVZKCP.S}_1(1^\lambda, 1^{T'}, 1^L, \hat{\pi}_{2,k})$.
 - In Steps 4 and 6, instead of computing $(\pi_{1,k}, \text{st}_k) \leftarrow \text{SHVZKCP.Prv}_1(1^\lambda, 1^{T'}, C', c_k, \hat{m}_k, d_k)$ and $\pi_{3,k} \leftarrow \text{SHVZKCP.Prv}_2(\text{st}_k, \hat{\pi}_{2,k})$, compute $(\pi_{1,k}, \pi_{3,k}) \leftarrow \text{SHVZKCP.S}_2(\text{st}_S, C')$ in Step 4.

By the special honest-verifier zero-knowledge of SHVZKCP, the output of $H_{4(k-1)+3}$ is computationally indistinguishable from that of $H_{4(k-1)+2}$.

- For each $k \in [\nu']$, Hybrid $H_{4(k-1)+4}$ differs from $H_{4(k-1)+3}$ only in the following ways.
 - In Step 2, instead of computing $(c_k, \text{st}_S) \leftarrow \text{SHVZKCP.S}_1(1^\lambda, 1^{T'}, 1^L, \hat{\pi}_{2,k})$, let $\hat{m}'_k := m' \parallel r'_1 \parallel \dots \parallel r'_B$ and compute $(c_k, d_k) \leftarrow \text{SHVZKCP.Com}(1^\lambda, 1^{T'}, \hat{m}'_k)$.
 - In Step 4, instead of computing $(\pi_{1,k}, \pi_{3,k}) \leftarrow \text{SHVZKCP.S}_2(\text{st}_S, C')$, compute $(\pi_{1,k}, \text{st}_k) \leftarrow \text{SHVZKCP.Prv}_1(1^\lambda, 1^{T'}, C', c_k, \hat{m}'_k, d_k)$ and $\pi_{3,k} \leftarrow \text{SHVZKCP.Prv}_2(\text{st}_k, \hat{\pi}_{2,k})$ in Steps 4 and 6.

As above, by the special honest-verifier zero-knowledge of SHVZKCP, the output of $H_{4(k-1)+4}$ is computationally indistinguishable from that of $H_{4(k-1)+3}$.

- For each $k \in [\nu']$, Hybrid $H_{4(k-1)+5}$ differs from $H_{4(k-1)+4}$ only in the following ways.
 - In Step 2, do not sample $\hat{\pi}_{2,k} \leftarrow \{0, 1\}^\ell$.
 - In Step 6, instead of setting $\Delta := \pi_{2,k} \oplus \hat{\pi}_{2,k}$ and $\hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta$ for each $i \in [\nu'] \setminus \{k\}$, sample $\Delta \leftarrow \{0, 1\}^\ell$ and let $\hat{\pi}_{2,i} := \pi_{2,i} \oplus \Delta$ for each $i \in [\nu']$.

As above, the distributions of $(\{\hat{\pi}_{2,i}\}_{i \in [\nu']}, \Delta)$ are identical between $H_{4(k-1)+5}$ and $H_{4(k-1)+4}$, so the output of $H_{4(k-1)+5}$ is identically distributed as that of $H_{4(k-1)+4}$.

Finally, we define $H_{4(\nu'-1)+6}$.

- Hybrid $H_{4(\nu'-1)+6}$ differs from $H_{4(\nu'-1)+5}$ only in the following ways.
 - In Step 2, do not sample $(r_1, \dots, r_B) \leftarrow \mathbb{F}^B$.
 - In Step 4, instead of setting $v_\sigma := f(u)$ using the polynomial $f(x) = r_B x^B + \dots + r_1 x + m$, let $v_\sigma \leftarrow \mathbb{F}$.

As above, when $u \neq 0$, the value $f(u)$ is uniformly distributed in the adversary's view, and thus the output of $H_{4(\nu'-1)+6}$ is identically distributed as that of $H_{4(\nu'-1)+5}$.

Hybrid $H_{4(\nu'-1)+6}$ is identical to the experiment in (6) with $m^* = m'$ (the roles of σ and $1 - \sigma$ are swapped in Step 4). Therefore the probability in (6) is at most $1/2 + \text{negl}(\lambda)$.

Next, we consider the general case where $L(\lambda) = \text{poly}(N(\lambda)^\epsilon)$. We use a natural extension of the hybrid experiments from the case $L(\lambda) = 1$. (For example, as H_1 , we consider an experiment in which we set $v_{1-\sigma}^{(\alpha)} := f'(u^{(\alpha)})$ instead of $v_{1-\sigma}^{(\alpha)} \leftarrow \mathbb{F}$ for each $\alpha \in [L]$.²¹) The indistinguishability among these hybrid experiments can be shown as above; it suffices to note the following points.

²¹As another example, as $H_{4(k-1)+2}$, we consider an experiment in which in Step 2 we additionally sample $\{\hat{\pi}_{2,k}^{(\alpha,\beta)}\}_{\alpha,\beta \in [L]} \leftarrow (\{0, 1\}^\ell)^{L^2}$, and in Step 6 we set $\Delta^{(\alpha,\beta)} := \pi_{2,k}^{(\alpha,\beta)} \oplus \hat{\pi}_{2,k}^{(\alpha,\beta)}$ for each $\alpha, \beta \in [L]$ and set $\hat{\pi}_{2,i}^{(\alpha,\beta)} := \pi_{2,i}^{(\alpha,\beta)} \oplus \Delta^{(\alpha,\beta)}$ for each $i \in [\nu'] \setminus \{k\}$.

- To show that the output of H_1 is identically distributed as that of H_0 , we use the fact that for any distinct $u^{(1)}, \dots, u^{(L)} \in \mathbb{F} \setminus \{0\}$ and uniformly chosen $(r'_1, \dots, r'_B)$, the values $f'(u^{(1)}), \dots, f'(u^{(L)})$ are uniformly distributed when $L \leq B$. (When ϵ is sufficiently small, we indeed have $L = \text{poly}(N^\epsilon) = \text{poly}(B^{\epsilon\eta}) \leq B$.)
- To show that the output of $H_{4(k-1)+3}$ is computationally indistinguishable from that of $H_{4(k-1)+2}$, we use the fact that the special honest-verifier zero-knowledge of SHVZKCP holds when $L(\lambda) = O(\nu^\delta)$ for $\nu = 3 \max(\lambda_{\text{SHVZKCP}}, T'(\lambda)) + 1$ and $\delta < 1/4$. (When ϵ is sufficiently small, we indeed have $L(\lambda) = \text{poly}(N(\lambda)^\epsilon) = \text{poly}(B^{\epsilon\eta}) = O(\nu^\delta)$ for $\delta < 1/4$.²²)

□

4.3 Step 3. Non-interactive WI Commit and Prove

Finally, using $\text{WICP} = (\text{WICP.Com}, \text{WICP.Rec}, \text{WICP.Vrfy}_0, \text{WICP.Priv}_1, \text{WICP.Vrfy}_1, \text{WICP.Priv}_2, \text{WICP.Vrfy}_2)$ constructed above, we define a non-interactive WI commit-and-prove scheme $\text{NIWICP} = (\text{NIWICP.Com}, \text{NIWICP.Rec}, \text{NIWICP.Priv}, \text{NIWICP.Vrfy})$ as follows. Let $\eta \geq 1$ be a sufficiently large constant chosen so that it satisfies (2) in Lemma 5.²³

Non-interactive witness-indistinguishable commit-and-prove NIWICP.

- $\text{NIWICP.Com}(1^\lambda, 1^T, m)$.
 1. Output $(c, d) \leftarrow \text{WICP.Com}(1^\lambda, 1^T, m)$.
- $\text{NIWICP.Rec}(1^\lambda, 1^T, c, m, d)$.
 1. Output $b := \text{WICP.Rec}(1^\lambda, 1^T, c, m, d)$.
- $\text{NIWICP.Priv}(1^\lambda, 1^T, C, c, m, d)$.
 1. Let $N := B^\eta$ for $B := \max(\lambda^{1/\epsilon}, T)$. Let L denote the number of strings output by HSG on inputs of length N .
 2. Let $x_0 := (1^\lambda, C, c)$, and use padding so that $|x_0| = N$.
 3. Compute $\{\pi_0^{(\alpha)}\}_{\alpha \in [L]} := \text{HSG}(x_0)$.
 4. For each $\alpha \in [L]$, compute $(\pi_1^{(\alpha)}, \text{st}^{(\alpha)}) \leftarrow \text{WICP.Priv}_1(1^\lambda, 1^T, C, c, m, d, \pi_0^{(\alpha)})$.^a
 5. For each $\alpha \in [L]$, let $x_1^{(\alpha)} := (1^\lambda, C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)})$, use padding so that $|x_1^{(\alpha)}| = N$, and compute $\{\pi_2^{(\alpha, \beta)}\}_{\beta \in [L]} := \text{HSG}(x_1^{(\alpha)})$.
 6. For each $\alpha, \beta \in [L]$, compute $\pi_3^{(\alpha, \beta)} \leftarrow \text{WICP.Priv}_2(\text{st}^{(\alpha)}, \pi_2^{(\alpha, \beta)})$.^b
 7. Output $\pi := (\{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha, \beta)}\}_{\alpha, \beta \in [L]})$.
- $\text{NIWICP.Vrfy}(1^\lambda, 1^T, C, c, \pi)$.

If $|C| \neq T$, output 0. Otherwise, do the following.

 1. Parse π as $(\{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha, \beta)}\}_{\alpha, \beta \in [L]})$, and compute $\{\pi_0^{(\alpha)}\}_{\alpha \in [L]}$ and $\{\pi_2^{(\alpha, \beta)}\}_{\alpha, \beta \in [L]}$ as in NIWICP.Priv .
 2. Output 1 if and only if $\text{WICP.Vrfy}_2(1^\lambda, 1^T, C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}, \pi_2^{(\alpha, \beta)}, \pi_3^{(\alpha, \beta)}) = 1$ holds for every $\alpha, \beta \in [L]$.

^aFormally speaking, each $\pi_0^{(\alpha)}$ is truncated to an appropriate length so that it can be fed into WICP.Priv_1 .

^bAgain, formally speaking, each $\pi_2^{(\alpha, \beta)}$ is truncated to an appropriate length.

²²To see the last equality, observe that given $T'(\lambda) = |C'| = B^{\Theta(1)}$, we have $\nu = 3 \max(\lambda_{\text{SHVZKCP}}, T'(\lambda)) + 1 = 3 \max(\lambda(B+1), B^{\Theta(1)}) + 1 = B^{\Theta(1)}$.

²³For the required size of η , see the proof of Lemma 11.

It is immediate from the construction that NIWICP makes only black-box use of WICP and thus only black-box use of NIC.

Lemma 9.

- **Commitment correctness.** For every $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, and $m \in \{0, 1\}^\lambda$,

$$\Pr \left[\text{NIWICP.Rec}(1^\lambda, 1^T, c, m, d) = 1 \mid (c, d) \leftarrow \text{NIWICP.Com}(1^\lambda, 1^T, m) \right] = 1.$$

- **Proof correctness.** For every $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, $m \in \{0, 1\}^\lambda$, and a circuit C such that $|C| = T$ and $C(m) = 1$,

$$\Pr \left[\text{NIWICP.Vrfy}(1^\lambda, 1^T, C, c, \pi) = 1 \mid \begin{array}{l} (c, d) \leftarrow \text{NIWICP.Com}(1^\lambda, 1^T, m) \\ \pi \leftarrow \text{NIWICP.Prv}(1^\lambda, 1^T, C, c, m, d) \end{array} \right] = 1.$$

Proof. Both claims follow immediately from the corresponding properties of WICP. (To see that the strings generated by HSG are long enough to serve as $\pi_0^{(\alpha)}$ and $\pi_2^{(\alpha, \beta)}$, observe that HSG outputs strings of length $N^\epsilon \geq \lambda \geq \max(|\pi_0^{(\alpha)}|, |\pi_2^{(\alpha, \beta)}|)$.²⁴) $\square$

Lemma 10.

- **Perfect binding.** For any $\lambda \in \mathbb{N}$, $T \in \mathbb{N}$, $c \in \{0, 1\}^*$, $(m_0, m_1) \in (\{0, 1\}^\lambda)^2$ such that $m_0 \neq m_1$, there is no $(d_0, d_1) \in (\{0, 1\}^*)^2$ satisfying $\text{NIWICP.Rec}(1^\lambda, 1^T, c, m_0, d_0) = 1 \wedge \text{NIWICP.Rec}(1^\lambda, 1^T, c, m_1, d_1) = 1$.
- **Computational hiding.** For every pair of PPT Turing machines $(\mathcal{A}_1, \mathcal{A}_2)$, every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, and every $\lambda \in \mathbb{N}$,

$$\Pr \left[\begin{array}{l} |m_0| = |m_1| \\ \wedge \mathcal{A}_2(\text{st}_{\mathcal{A}}, c) = b \end{array} \mid \begin{array}{l} (m_0, m_1, \text{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, 1^{T(\lambda)}) \\ b \leftarrow \{0, 1\} \\ (c, d) \leftarrow \text{NIWICP.Com}(1^\lambda, 1^{T(\lambda)}, m_b) \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda).$$

Proof. Both claims follow immediately from the corresponding properties of WICP. $\square$

Lemma 11 (Soundness). For every PPT Turing machine $\mathcal{A}$, every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, and every $\lambda \in \mathbb{N}$,

$$\Pr \left[\text{SUCCESS}(\lambda, T, C, c, \pi) \mid (C, c, \pi) \leftarrow \mathcal{A}(1^\lambda, 1^{T(\lambda)}) \right] \leq \text{negl}(\lambda), \quad (7)$$

where $\text{SUCCESS}(\lambda, T, C, c, \pi)$ is the event that all of the following hold.

- For every $m \in \{0, 1\}^\lambda$ such that $C(m) = 1$ and every $d \in \{0, 1\}^*$, it holds $\text{NIWICP.Rec}(1^\lambda, 1^{T(\lambda)}, c, m, d) \neq 1$.
- $\text{NIWICP.Vrfy}(1^\lambda, 1^{T(\lambda)}, C, c, \pi) = 1$.

Proof. The proof proceeds in three steps.

First, we prove the following.

$$\Pr \left[\text{CHEAT}_1 \mid \begin{array}{l} (C, c, (\{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha, \beta)}\}_{\alpha, \beta \in [L]})) \leftarrow \mathcal{A}(1^\lambda, 1^{T(\lambda)}) \\ x_0 := (1^\lambda, C, c) \\ \{\pi_0^{(\alpha)}\}_{\alpha \in [L]} := \text{HSG}(x_0) \end{array} \right] \leq \text{negl}(\lambda), \quad (8)$$

where CHEAT_1 is the event that all of the following hold.

²⁴Recall from Lemma 5 that $|\pi_0^{(\alpha)}| = \lambda$ and $|\pi_2^{(\alpha, \beta)}| = O((\log(\lambda + T))^2)$.

- For every $m \in \{0,1\}^\lambda$ such that $C(m) = 1$ and every $d \in \{0,1\}^*$, it holds $\text{NIWICP.Rec}(1^\lambda, 1^{T(\lambda)}, c, m, d) \neq 1$.
- For every $\alpha \in [L]$, it holds $\text{Doomed}_1(\lambda, T(\lambda), C, c, \pi_0^{(\alpha)}) = 0$.

Equation (8) follows immediately from our assumption that HSG is a $(1/2, \text{negl})$ -targeted HSG for $\text{DTIME}[N]^{\text{SAT}}$ over all polynomial-time samplable distributions. Indeed, for any $(\lambda, T(\lambda), C, c)$, if $\text{NIWICP.Rec}(1^\lambda, 1^{T(\lambda)}, c, m, d) \neq 1$ for every $m \in \{0,1\}^\lambda$ such that $C(m) = 1$ and every $d \in \{0,1\}^*$, then by the round-by-round soundness of WICP, we have

$$\Pr \left[\text{Doomed}_1(\lambda, T(\lambda), C, c, \pi_0) = 1 \mid \pi_0 \leftarrow \text{WICP.Vrfy}_0(1^\lambda, 1^{T(\lambda)}) \right] \geq \frac{1}{2}.$$

Moreover, since Doomed_1 can be computed in time $\text{poly}(B)$ with oracle access to $\text{NTIME}[\text{poly}(B)]$, it follows that when η is sufficiently large, we have $\text{Doomed}_1 \in \text{DTIME}[N]^{\text{SAT}}$, i.e., Doomed_1 is computable in linear time in the input length given oracle access to SAT. (Recall that $N := B^\eta$.) Therefore, $\text{Doomed}_1(\lambda, T(\lambda), C, c, \cdot)$ is a $1/2$ -dense avoider for $\text{HSG}(x_0)$ with negligible probability over the choice of $x_0 = (1^\lambda, C, c)$, and thus, we have (8).

Next, we prove the following.

$$\Pr \left[\begin{array}{l} \exists k \in [L] \text{ s.t. } \text{CHEAT}_{2,k} \\ (C, c, (\{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha, \beta)}\}_{\alpha, \beta \in [L]})) \leftarrow \mathcal{A}(1^\lambda, 1^{T(\lambda)}) \\ x_0 := (1^\lambda, C, c) \\ \{\pi_0^{(\alpha)}\}_{\alpha \in [L]} := \text{HSG}(x_0) \\ \forall \alpha \in [L] : x_1^{(\alpha)} := (1^\lambda, C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}) \\ \forall \alpha \in [L] : \{\pi_2^{(\alpha, \beta)}\}_{\beta \in [L]} := \text{HSG}(x_1^{(\alpha)}) \end{array} \right] \leq \text{negl}(\lambda), \quad (9)$$

where for each $k \in [L]$, $\text{CHEAT}_{2,k}$ is the event that all of the following hold.

- $\text{Doomed}_1(\lambda, T(\lambda), C, c, \pi_0^{(k)}) = 1$.
- For every $\beta \in [L]$, it holds $\text{Doomed}_2(\lambda, T(\lambda), C, c, \pi_0^{(k)}, \pi_1^{(k)}, \pi_2^{(k, \beta)}) = 0$.

As above, Equation (9) follows immediately from our assumption that HSG is a $(1/2, \text{negl})$ -targeted HSG for $\text{DTIME}[N]^{\text{SAT}}$ over all polynomial-time samplable distributions. Indeed, choose $k \in [L]$ uniformly at random and apply the targeted HSG to the resulting polynomial-time samplable distribution of $x_1^{(k)}$. Whenever $\text{CHEAT}_{2,k}$ holds for some k , the uniformly chosen index satisfies $\text{CHEAT}_{2,k}$ with probability at least $1/L$. Since $L = N^{O(\epsilon)}$ and the failure probability of the targeted HSG is $N^{-\omega(1)}$, this polynomial loss preserves negligibility, proving (9).

We are now ready to prove (7). From (8) and (9), the following holds.

$$\Pr \left[\begin{array}{l} \text{CHEAT}_3 \\ (C, c, (\{\pi_1^{(\alpha)}\}_{\alpha \in [L]}, \{\pi_3^{(\alpha, \beta)}\}_{\alpha, \beta \in [L]})) \leftarrow \mathcal{A}(1^\lambda, 1^{T(\lambda)}) \\ x_0 := (1^\lambda, C, c) \\ \{\pi_0^{(\alpha)}\}_{\alpha \in [L]} := \text{HSG}(x_0) \\ \forall \alpha \in [L] : x_1^{(\alpha)} := (1^\lambda, C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}) \\ \forall \alpha \in [L] : \{\pi_2^{(\alpha, \beta)}\}_{\beta \in [L]} := \text{HSG}(x_1^{(\alpha)}) \end{array} \right] \leq \text{negl}(\lambda), \quad (10)$$

where CHEAT_3 is the event that all of the following hold.

- For every $m \in \{0,1\}^\lambda$ such that $C(m) = 1$ and every $d \in \{0,1\}^*$, it holds $\text{NIWICP.Rec}(1^\lambda, 1^{T(\lambda)}, c, m, d) \neq 1$.
- For every $\alpha, \beta \in [L]$, it holds $\text{Doomed}_2(\lambda, T(\lambda), C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}, \pi_2^{(\alpha, \beta)}) = 0$.

Moreover, by the round-by-round soundness of WICP, for each $\alpha, \beta \in [L]$, it holds that if $\text{Doomed}_2(\lambda, T(\lambda), C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}, \pi_2^{(\alpha, \beta)}) = 1$, then

$$\forall \pi_3^{(\alpha, \beta)} : \text{WICP.Vrfy}_2(1^\lambda, 1^{T(\lambda)}, C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}, \pi_2^{(\alpha, \beta)}, \pi_3^{(\alpha, \beta)}) = 0.$$

(Equivalently, for each $\alpha, \beta \in [L]$, we have $\text{WICP.Vrfy}_2(1^\lambda, 1^{T(\lambda)}, C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}, \pi_2^{(\alpha, \beta)}, \pi_3^{(\alpha, \beta)}) = 1$ for some $\pi_3^{(\alpha, \beta)}$ only when $\text{Doomed}_2(\lambda, T(\lambda), C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}, \pi_2^{(\alpha, \beta)}) = 0$.) Then, since by construction we have $\text{NIWICP.Vrfy}(1^\lambda, 1^{T(\lambda)}, C, c, \pi) = 1$ only when $\text{WICP.Vrfy}_2(1^\lambda, 1^{T(\lambda)}, C, c, \pi_0^{(\alpha)}, \pi_1^{(\alpha)}, \pi_2^{(\alpha, \beta)}, \pi_3^{(\alpha, \beta)}) = 1$ for every $\alpha, \beta \in [L]$, (7) follows from (10). $\square$

Lemma 12 (Witness indistinguishability). *For every triple of PPT Turing machines $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$, every polynomial $T : \mathbb{N} \rightarrow \mathbb{N}$, and every $\lambda \in \mathbb{N}$,*

$$\Pr \left[\begin{array}{l} |m_0| = |m_1| \\ \wedge C(m_0) = C(m_1) = 1 \\ \wedge \mathcal{A}_3(\text{st}_{\mathcal{A},2}, \pi) = b \end{array} \middle| \begin{array}{l} (m_0, m_1, \text{st}_{\mathcal{A},1}) \leftarrow \mathcal{A}_1(1^\lambda, 1^{T(\lambda)}) \\ b \leftarrow \{0, 1\} \\ (c, d) \leftarrow \text{NIWICP.Com}(1^\lambda, 1^{T(\lambda)}, m_b) \\ (C, \text{st}_{\mathcal{A},2}) \leftarrow \mathcal{A}_2(\text{st}_{\mathcal{A},1}, c) \\ \pi \leftarrow \text{NIWICP.Priv}(1^\lambda, 1^{T(\lambda)}, C, c, m_b, d) \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda). \quad (11)$$

Proof. By construction, this follows immediately from the witness indistinguishability of WICP against bounded rewinding adversaries. Indeed, HSG outputs $\text{poly}(N^\epsilon)$ strings (where $N := B^\eta$ for $B := \max(\lambda^{1/\epsilon}, T)$), and it can be assumed without loss of generality that these strings are distinct, so we can use Lemma 8 to obtain (11). $\square$

This completes the proof of Theorem 3.

Statement on the Use of Generative AI

Limited or no AI contribution.²⁵ Generative AI tools were used solely to assist with language editing, document organization, formatting, and minor technical checks. They were not used to generate research results, mathematical claims, or proofs. All AI-assisted material included in the manuscript was reviewed and verified by the author, who takes full responsibility for the content of this article.

References

- [BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.
- [Blu82] Manuel Blum. Coin flipping by telephone. In *Proc. IEEE Spring COMPCOM*, pages 133–137, 1982.
- [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 Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1082–1090. ACM Press, June 2019.
- [CDMW09] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 387–402. Springer, Berlin, Heidelberg, March 2009.
- [CDMW18] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. A black-box construction of non-malleable encryption from semantically secure encryption. *Journal of Cryptology*, 31(1):172–201, January 2018.

²⁵This designation follows the [IACR Generative AI Policy](#) (revised September 22, 2026).

- [CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *Journal of the ACM*, 51(4):557–594, July 2004.
- [CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503. ACM Press, May 2002.
- [CRT24] Lijie Chen, Ron D. Rothblum, and Roei Tell. Fiat-Shamir in the Plain Model from Derandomization (Or: Do Efficient Algorithms Believe that $NP = PSPACE$?). Electronic Colloquium on Computational Complexity (ECCC), TR24-116, Revision #1, 2024. A full version of [CRT25].
- [CRT25] Lijie Chen, Ron D. Rothblum, and Roei Tell. Fiat-Shamir in the plain model from derandomization (or: Do efficient algorithms believe that $NP = PSPACE$?). In Michal Koucký and Nikhil Bansal, editors, *57th ACM STOC*, pages 977–985. ACM Press, June 2025.
- [CT21] Lijie Chen and Roei Tell. Hardness vs Randomness, Revised: Uniform, Non-Black-Box, and Instance-Wise. Electronic Colloquium on Computational Complexity (ECCC), TR21-080, 2021. A full version of [CT22a].
- [CT22a] Lijie Chen and Roei Tell. Hardness vs randomness, revised: Uniform, non-black-box, and instance-wise. In *62nd FOCS*, pages 125–136. IEEE Computer Society Press, February 2022.
- [CT22b] Lijie Chen and Roei Tell. When Arthur has Neither Random Coins nor Time to Spare: Superfast Derandomization of Proof Systems. Electronic Colloquium on Computational Complexity (ECCC), TR22-057, Revision #2, 2022. A full version of [CT23b].
- [CT23a] Lijie Chen and Roei Tell. Guest column: New ways of studying the $BPP = P$ conjecture. *SIGACT News*, 54(2):44 – 69, June 2023.
- [CT23b] Lijie Chen and Roei Tell. When arthur has neither random coins nor time to spare: Superfast derandomization of proof systems. In Barna Saha and Rocco A. Servedio, editors, *55th ACM STOC*, pages 60–69. ACM Press, June 2023.
- [DN07] Cynthia Dwork and Moni Naor. Zaps and their applications. *SIAM Journal on Computing*, 36(6):1513–1543, 2007.
- [FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO’86*, volume 263 of *LNCS*, pages 186–194. Springer, Berlin, Heidelberg, August 1987.
- [GLOV12] Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *53rd FOCS*, pages 51–60. IEEE Computer Society Press, October 2012.
- [GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.
- [Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*, volume 1. Cambridge University Press, Cambridge, UK, 2001.
- [Gol11] Oded Goldreich. Two Comments on Targeted Canonical Derandomizers. Electronic Colloquium on Computational Complexity (ECCC), TR11-047, 2011.

- [GOSV14] Vipul Goyal, Rafail Ostrovsky, Alessandra Scafuro, and Ivan Visconti. Black-box non-black-box zero knowledge. In David B. Shmoys, editor, *46th ACM STOC*, pages 515–524. ACM Press, May / June 2014.
- [HVW22] Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, and Mor Weiss. ZK-PCPs from leakage-resilient secret sharing. *Journal of Cryptology*, 35(4):23, October 2022.
- [IKOS09] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM Journal on Computing*, 39(3):1121–1152, 2009.
- [IW97] Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *29th ACM STOC*, pages 220–229. ACM Press, May 1997.
- [Khu21] Dakshita Khurana. Non-interactive distributional indistinguishability (NIDI) and non-malleable commitments. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part III*, volume 12698 of *LNCS*, pages 186–215. Springer, Cham, October 2021.
- [KOS18] Dakshita Khurana, Rafail Ostrovsky, and Akshayaram Srinivasan. Round optimal black-box “commit-and-prove”. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 286–313. Springer, Cham, November 2018.
- [LP12] Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 461–478. Springer, Berlin, Heidelberg, August 2012.
- [MS81] R. J. McEliece and D. V. Sarwate. On sharing secrets and reed-solomon codes. *Communications of the ACM*, 24(9):583 – 584, September 1981.
- [NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.
- [Sha79] Adi Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, November 1979.
- [Tel25] Roei Tell. Hardness vs Randomness in the 2020s: a practical guide for the working cryptographer, 2025. An invited talk given at TCC 2025. <https://sites.google.com/site/roeitell/videos>.