Overcoming Padding in Cryptography

via Hardness of Certifying Random Strings

Yao-Ching Hsieh[*] University of Washington
Abhishek Jain[†] NTT Research & JHU
Jiatu Li[‡] MIT
Surya Mathialagan[§] NTT Research

Abstract

Modern cryptography relies on a provable security paradigm where the security of a primitive is established from the intractability of one or more computational problems. Hybrid arguments are central to this approach. Somewhat unexpectedly, many security proofs in the literature that use this methodology require hybrids with description-size larger than the real scheme. To argue security, the real scheme is artificially *padded* to match this size. This so-called "superfluous padding" phenomenon is surprisingly commonplace and poses a barrier to achieving *succinct* cryptographic primitives where the size of the scheme parameters does not grow with the input length.

We present a framework to eliminate padding in cryptography. We demonstrate several applications, including rate-1 iO, succinct (and extractable) witness encryption and succinct non-interactive arguments with short CRS for subclasses of NP, and multi-input functional encryption for functions with unbounded arity.

To obtain our results, we establish a new connection between cryptography and proof complexity. Central to our approach is the notion of *proof-complexity generators* (PCGs) [Krajíček'01, Alekhnovich, Ben-Sasson, Razborov, Wigderson'04], which, informally speaking, formalizes the hardness of certifying random (incompressible) strings with propositional proof systems. Our applications are derived from a master theorem that lifts cryptographic security on "compressible instances" to security on *any* instance under the existence of PCGs. Due to its broad scope, this theorem might find future applications.

[*] ychsieh@cs.washington.edu
[†] abhishek.jain@ntt-research.com
[‡] jiatuli@mit.edu
[§] surya.mathialagan@ntt-research.com

Contents

I Overview of Results **1**

1 Introduction **1**
1.1 Our Results . . . 2
1.2 Applications . . . 5
1.3 On the Existence of Proof Complexity Generators . . . 10
1.4 On the Theory APC_1 . . . 13
1.5 Future Directions . . . 14
1.6 Related Work . . . 15

2 Technical Overview **16**
2.1 The Master Theorem and Bounded Arithmetic . . . 16
2.2 Building Primitives assuming Compression . . . 17
2.3 Discussion: Non-Examples of our Master Theorem . . . 26
2.4 Organization of the Paper . . . 27

3 Preliminaries **28**
3.1 Bounded Arithmetic . . . 28
3.2 Time Bounded Kolmogorov Complexity . . . 29
3.3 Cryptographic Definitions and Tools . . . 30

4 Upgrading Security via Proof Complexity Generators **41**
4.1 Proof Complexity Generator against NP . . . 41
4.2 Falsifiable Security Games . . . 44
4.3 Master Theorem for Upgrading Security . . . 45
4.4 Typical Parameter Regimes . . . 47

II Applications **49**

5 Rate-1 Obfuscation and Applications **49**
5.1 Rate-1 iO for Circuits . . . 50
5.2 Rate-1 iO for EF-Equivalent Programs . . . 55
5.3 Applications to Witness Encryption . . . 58

6 Succinct Multi-Input Functional Encryption **61**
6.1 Relation-succinct NIWIPoK with PV proofs. . . . 61
6.2 Construction of succinct EF-MIFE . . . 63

7 Non-Adaptive EF-SNARGs with Short CRS **72**
7.1 Tools and Definitions from Prior Work . . . 72
7.2 Construction and analysis . . . 74

III Formalization in Bounded Arithmetic **79**

8 Proof Complexity Generators against Feasible Cryptography **79**
8.1 Notation and Basic Concepts . . . 79
8.2 Infinitely-Often Provability . . . 80
8.3 Proof Complexity Generators against Bounded Arithmetic . . . 81
8.4 Time-bounded Chaitin Incompleteness . . . 82

9 Upgrading Security via PCG against Feasible Cryptography **83**
9.1 Formalizations in Bounded Arithmetic . . . 83
9.2 Master Theorem with Bounded Arithmetic . . . 84
9.3 Typical Parameter Regimes . . . 87

10 Applications **87**
10.1 Cryptographic Primitives Used as Axioms . . . 88
10.2 Cryptographic Primitives Constructed from Axioms . . . 93
10.3 Formalization of Security Analysis . . . 97

A The Hardness Assumption of Korten and Santhanam **113**

B Missing Formalizations **114**
B.1 Circuit SAT Reduces to Hamiltonian Cycle . . . 114
B.2 Description Succinct Obfuscation . . . 115
B.3 Correctness of the Indistinguishability Game . . . 117
B.4 EF-Equivalence Proof for Lemma 6.16 . . . 118

C Justifications and Cryptanalysis of Proof Complexity Generators **120**

Part I
Overview of Results

1 Introduction

The success of modern cryptography is rooted in a provable security approach in which the security of a scheme is proven based on the intractability of one or more computational problems. Hybrid arguments play a central role in this paradigm. Informally speaking, this process involves modifying a distribution in a sequence of indistinguishable steps to reach a target distribution, where the indistinguishability of each step is predicated on the intractability of a computational problem. This technique is used in the security proofs of nearly all computationally-secure cryptographic primitives, including advanced primitives such as indistinguishability obfuscation (iO) [BGI+01].

Unsurprisingly, the "efficiency" of the hybrid argument has a direct bearing on the efficiency of the cryptographic scheme. A standard measure of efficiency is the number of hybrids: if the proof relies on the security of an assumption in (say) k hybrids, then the security parameter of that assumption must grow with $\log k$. Large values of k (say, exponential in some parameter of the scheme) necessitate a large security parameter, which adversely affects the efficiency of the scheme.

Security via padding. An orthogonal efficiency consideration is the *size* of each hybrid. If an intermediate hybrid is larger in size than the real scheme, then the real scheme must be *padded* to match this size in order to argue its security. This might seem unnatural: why should padding make a scheme secure? Surprisingly, this is a common phenomenon in cryptography. In the iO literature, it was initially referred to as "superfluous padding" [BM15] but was later shown to be necessary in some cases [Hol15]. In fact, this phenomenon is not limited to iO. Consider a security proof where an intermediate hybrid involves encryption of a larger string than in the real world. Since encryption security only holds for same-length messages, one must pad the encrypted string in the real world. This example is not fictitious – it, in fact, arises in recent constructions of succinct non-interactive arguments (SNARGs) [JKLV24, JKLM25, JJ25].

Succinctness beyond falsifiability. The first issue pertaining to the use of a large number of hybrids is common in the security proofs of cryptographic primitives whose security definitions are not efficiently falsifiable [Nao03].[1] For example, the number of hybrids in security proofs for iO [JLS21, JLS22, RVV24] is exponential in the input length.[2] This creates a barrier to achieving *succinct* iO where the size of the obfuscated program is independent of the input length.

Interestingly, even if we resolve the first issue or simply make peace with it, the second issue, namely the seeming necessity of padding, persists and creates a *second barrier* to achieving succinctness. Let us illustrate with two examples:

- *Rate-1 iO*: Even if we assume the existence of iO for Turing machines with unbounded input length, constructions of *rate-1* iO remain elusive. Here, by rate, we refer to the ratio between the size

[1] A hardness assumption is efficiently falsifiable if the challenger in the security experiment runs in polynomial time, and can decide in polynomial time whether an adversary wins.

[2] Recall that iO guarantees indistinguishability of obfuscations of any two programs of the same size that are functionally equivalent. Checking whether two programs are equivalent is, in general, inefficient.

(measured in the number of bits[3]) of the unobfuscated program and the obfuscated program. Consider the following natural template for rate-1 iO: to obfuscate a program P, we encrypt P using a rate-1 encryption scheme and obfuscate a Turing machine M that works as follows. Upon receiving as input the encryption of P and an input x for P, it decrypts P and outputs $P(x)$. It is easy to see that this candidate achieves rate-1 efficiency. However, it is not clear how to prove its security for two equivalent programs P_0, P_1. Prior works [BV15, AJS17, JLL23, JJMP25] prove security by creating an intermediate hybrid, where we encrypt *both* P_0 and P_1. However, this means that, in the real scheme, we must pad the program to twice its size. This results in a rate-1/2 scheme.

- *SNARGs with short CRS*: Under standard assumptions, recent works [JJ22, JKLV24] have constructed SNARGs for a subclass of $\mathsf{NP} \cap \mathsf{coNP}$ where non-membership can be proven in the Extended Frege (EF) propositional proof system (see [CR79]).[4] Although they achieve succinct proofs, their CRS is *not* succinct. In particular, their security proofs require a large CRS whose size grows with the size of the EF proof (which grows polynomially in the size of the coNP witness). This means that the CRS in the real scheme must be padded to this size, resulting in a non-succinct solution. Note that known barriers to achieving SNARGs with short CRS [GW11, CJ25] are not applicable to this setting.[5]

Later, we will discuss more examples where the seeming necessity of padding creates similar barriers to achieving succinct constructions.

We investigate whether such padding can be avoided or if it is always necessary. Note that a positive resolution to this question would open the door to achieving succinctness *beyond* the falsifiability barrier.

1.1 Our Results

We develop a new approach to avoid superfluous padding in the above examples, and beyond.

The key insight. Before presenting our results, let us provide some intuition as to why this might even be possible. Let us revisit the above template for constructing rate-1 iO. For any pair of equivalent programs P_0 and P_1, suppose that P_1 is *compressible* in the sense that it has a short description ρ of size $o(|P_0|)$. In this case, one can achieve rate-1 iO as follows: in the hybrid where prior works encrypt both P_0, P_1, we now encrypt ρ instead of P_1. The machine M is modified so that upon receiving encryption of P_0, ρ, it first "expands" ρ into P_1 and then works in the same manner as before. Crucially, now the real scheme only needs to use padding of size $|\rho|$ and therefore achieves rate-1 efficiency. Ignoring some details, it turns out that this approach can indeed be translated to a secure scheme.

But what if we do not have a compressed description of P_1? In this case, it is not clear why the above scheme with size $|\rho|$ padding should be secure. Our key observation is that if an adversary successfully breaks the scheme in this case, then it can be turned into an *efficient certificate that a string is incompressible* (i.e., random). This is a well-studied problem in proof complexity and is conjectured to be hard (see our specific formulation below). This observation forms the basis for our positive results.

[3] An alternative formulation of rate-1 iO where rate refers to the ratio between the runtime of the unobfuscated and the obfuscated programs is known to be impossible [LMOP26].

[4] Extended Frege is a propositional proof system in which one proves tautologies using standard logical inference rules together with *extension axioms* of the form $p \leftrightarrow \varphi$, where p is a fresh variable and φ is a Boolean formula.

[5] While the definition of SNARGs, similarly to iO, is not efficiently falsifiable in general, it is falsifiable when restricted to languages in $\mathsf{NP} \cap \mathsf{coNP}$.

1.1.1 Proof Complexity Generator Assumption

The main leverage behind our results is the assumption that *proof complexity generators* (aka. PCG) exist. This primitive was introduced independently by Krajíček [Kra01] and Alekhnovich, Ben-Sasson, Razborov, and Wigderson [ABSRW04], as a class of candidate hard tautologies towards strong proof complexity lower bounds.

Fix any propositional proof system P [CR79]. A family of stretching circuits $C_m : \{0,1\}^\ell \to \{0,1\}^m$ (where $\ell = \ell(m) < m$) is said to be a proof complexity generator if for *every* $y \in \{0,1\}^m$, the formula $\tau_y(C_m)$ formalizing "$C_m(x) \neq y$ for every $x \in \{0,1\}^\ell$" requires superpolynomial size P-proofs. Note that for at most 2^ℓ strings y, $\tau_y(C_m)$ is not a tautology and thus not provable from P, while for other (at least a $(1-2^{\ell-m})$-fraction of) strings, $\tau_y(C_m)$ are tautologies. Therefore, when $\ell \ll m$, a proof complexity generator offers an efficient method to sample hard tautologies against P. One may view a stretching circuit $C_m : \{0,1\}^\ell \to \{0,1\}^m$ as the decompression function of a compression scheme. Under this interpretation, a proof complexity generator is a compression scheme such that it is hard to prove a string is incompressible.

Our assumption claims that there is a proof complexity generator for *any* propositional proof system with a "simple soundness proof". Let $\ell = \ell(m) = o(m)$ be a function called *compression rate* and $t = t(m)$ be a function. For a mathematical theory $\mathcal{T}$, we assert that:

Assumption 1 (Proof Complexity Generators against $\mathcal{T}$)**.** There is a uniform family of circuits $C_m : \{0,1\}^\ell \to \{0,1\}^m$ (called t-secure proof complexity generators) such that for every polynomial time algorithm $V(y,\pi)$, the following two conditions cannot hold simultaneously:

- (*Completeness*). For infinitely many m, there exists a string $y \in \{0,1\}^m \setminus \mathsf{Range}(C_m)$ and a string $\pi \in \{0,1\}^t$ such that $V(y,\pi) = 1$.
- ($\mathcal{T}$*-provable Soundness*). The following sentence ϕ is provable in the mathematical theory $\mathcal{T}$: "for every m, every $y \in \mathsf{Range}(C_m)$, and every $\pi \in \{0,1\}^t$, $V(y,\pi) = 0$".

We will work with Assumption 1 with superpolynomial compression rate $\ell = m^{o(1)}$ and polynomial hardness (i.e. $t = \mathsf{poly}(m)$) or superpolynomial hardness (i.e. $t = m^{\omega(1)}$), as well as super-linear compression rate $\ell(m) = o(m)$ and subexponential hardness (i.e. $t = 2^{\ell^{\Omega(1)}}$). Note that both parameter regimes are not trivially false even for $\mathcal{T} = \mathsf{ZFC}$ (i.e., the Zermelo-Fraenkel set theory with the axiom of choice), as the only known unconditional algorithm V runs in $2^{O(\ell)}$ time by exhaustively enumerating over all possible outputs of C_m.

Intuitively, Assumption 1 states the *hardness of certifying incompressible strings*. Suppose that a verifier $V(y,\pi)$ is complete and $\mathcal{T}$-provably sound. Then for every string y, by the provable soundness property, an NP-witness π such that $V(y,\pi) = 1$ combined with the $\mathcal{T}$-proof of the sentence ϕ certify the incompressibility of y. This proof is convincing to anyone who can both understand the theory $\mathcal{T}$ and verify $V(y,\pi) = 1$, and thus for a weak mathematical theory $\mathcal{T}$, the proof is "simple" in a formal sense. The assumption rules out the case that infinitely many incompressible strings can be certified in this way.

It is important to note that this assumption is *not* in contradiction with the fact that almost every string is incompressible. It is possible to efficiently generate strings that are highly likely to be incompressible. However, as there is no known algorithm to verify the incompressibility of strings, it is hard to make the algorithm *errorless*, which is required by the soundness property. More in-depth discussions of the assumption are deferred to Section 1.3.

For most of our applications, it suffices to assume the *existence* of a proof complexity generator rather than an explicit construction of it. This is because the proof complexity generator is used only in security proofs, but not in the constructions. Nevertheless, we note that a universal candidate for PCG is known [RSW22] (see also Section 4.1.2 and 8.4), and it is used in one of our applications.

Instantiation. The assumption is most appealing when $\mathcal{T}$ is a weak yet well-studied theory. Following [HJLM26], the theory $\mathcal{T}$ will be chosen as Jeřábek's bounded arithmetic theory APC_1 [Jeř07a] augmented with the following non-logical axioms:

- a standard derandomization assumption $\mathsf{prBPP} = \mathsf{prP}$,
- the security of cryptographic primitives, including fully homomorphic encryption (FHE) [Gen09, GSW13], pseudorandom generators (PRG) [BM84, Yao82], batch arguments for NP (BARG) [CJJ21], as well as a recently developed primitive called fully-succinct EF-iO for Turing machines [JJMP25].[6]

For simplicity, we may call the theory "feasible cryptography", i.e., the extension of feasible mathematics APC_1 with axioms for cryptographic assumptions. For this instantiation, Assumption 1 suggests that it is hard to provably certify incompressible strings, even with help from (specific) cryptographic security axioms.

1.1.2 Master Theorem: Lifting Limited Security to Full Security

In this work, we present several applications of PCGs to cryptography. All our applications are built on a master theorem that utilizes Assumption 1. In the following, we first state our master theorem and then describe the end results in Section 1.2.

Consider the following generic template of falsifiable security games.

- Fix an NP language $\mathcal{U}$, a probabilistic polynomial-time algorithm $\mathcal{C}(x, w, A)$ called the *challenger*, and a threshold $\gamma \in (0, 1)$.
- On an input length n, an adversary of size t wins with advantage ε if there exists a string $x \in \mathcal{U}$ of length n, an NP witness w for $x \in \mathcal{U}$, and a circuit A of size at most t such that

$$\Pr\left[\mathcal{C}(x, w, A) = 1\right] > \gamma + \varepsilon.$$

We say that the game is secure if any polynomial-sized adversary wins with a negligible advantage.

The language $\mathcal{U}$ defines valid inputs to the security game. For example, in the security game for public-key encryption, the instance x is a pair of messages (m_0, m_1), and the language $\mathcal{U}$ consists of all valid pairs of messages with the same length. The challenger $\mathcal{C}(x, w, A)$ randomly picks $b \in \{0, 1\}$, encrypts m_b as ct, and accepts if $b = A(\mathsf{pk}, \mathsf{ct})$. The threshold γ is set to $1/2$. (Here, it suffices to define $\mathcal{U} \in \mathsf{P}$ and $w = \varepsilon$, while in more general cases, we may need to use an NP language.)

The master theorem. At a high level, our master theorem states that under Assumption 1, for any game of the above form such that:

- it is secure with an additional restriction that $x \| w$ is *compressible*: $x \| w$ has low time-bounded Kolmogorov complexity, and
- the security proof (with the additional restriction on compressibility) can be formalized in the mathematical theory $\mathcal{T}$,

then the game is secure (without the restriction on compressibility).

Theorem 1.1 (Master Theorem, informal; see Theorem 9.3)**.** *Let $\mathcal{T}$ be a "nice" mathematical theory, and assume $\mathsf{prBPP} = \mathsf{prP}$. Under Assumption 1 with compression rate $m^{o(1)}$ and polynomial security, there are*

[6] For technical reasons, we need to first fix constructions of these primitives and add the security axioms of the fixed constructions; see Section 1.5 for more discussion.

functions $\ell(n) = n^{o(1)}$, $t(n) = n^{O(1)}$ such that the following is true. Fix any security game of the above form. Suppose that it is secure on any input x and witness w such that $\boldsymbol{K}^t(x\|w) \leq \ell(|x| + |w|)$. That is:

- *For every polynomial s, there is a negligible function $\mathsf{negl}(\cdot)$ such that on any input length n, any $x \in \mathcal{U}$ of length n, any NP witness w for $x \in \mathcal{U}$ and any adversary A of size $s(n)$, if $\boldsymbol{K}^t(x\|w) \leq \ell(|x| + |w|)$, then*

$$\Pr\left[\mathcal{C}(x, w, A) = 1\right] \leq \gamma + \mathsf{negl}(n).$$

Moreover, suppose that the security is provable within the theory $\mathcal{T}$.

Then the security game is also secure (without the restriction on Kolmogorov complexity).

The master theorem can be interpreted as follows: For a cryptographic construction with security captured as a falsifiable security game, we can effectively assume the input x and its NP-witness w as *compressible* in the security proof provided that:

(i) The security proof can be formalized within the theory $\mathcal{T}$.

(ii) Assumption 1 holds, i.e., there is a PCG against PPS with $\mathcal{T}$ proof of soundness.

Here, $\mathcal{T}$ is "nice" if it is a sound extension of a bounded arithmetic theory APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$" [Jeř07a, HJLM26] to support basic reasoning for probabilistic polynomial time algorithms; see Section 1.4 for more details.

Remark 1. The formal master theorem (see Theorem 9.3) allows flexible parameter regimes on the security of PCGs, the security of the game, and the efficiency of the language $\mathcal{U}$ (that captures the falsifiability of the game). All three parameter settings we will use can be found in Section 9.3.

For instance, we will use the following version of the master theorem: Assuming 2^{n^δ}-secure proof complexity generators for $\ell(n) = o(n), t(n) = n^{O(1)}$ against feasible cryptography for a constant $\delta \in (0, 1)$, for any game with $\mathcal{U} \in \mathsf{DTIME}[T]$ for $T(n) = 2^{n^{\delta'}}$, $\delta' < \delta$, if it is provably secure on compressible instances, it is also secure. (Here, $\mathcal{U} \in \mathsf{DTIME}[T]$ means that there is no witness w, and thus an instance x is said to be compressible if $\boldsymbol{K}^t(x) \leq \ell(|x|)$.)

1.2 Applications

We identify several important cryptographic primitives where known constructions are non-succinct due to the use of superfluous padding. Using the master theorem (Theorem 1.1), we show how to "compress" the padding and obtain *succinct* solutions. Our results cover core cryptographic primitives both in the inefficiently falsifiable and efficiently falsifiable regimes, thus highlighting the versatility of our master theorem. Looking ahead, our results include rate-1 iO (Section 1.2.1), succinct (and extractable) witness encryption (Section 1.2.2), SNARGs with short CRS (Section 1.2.3), and succinct multi-input functional encryption (Section 1.2.4).

We now discuss our applications in more detail.

1.2.1 Rate-1 Obfuscation

Our first application is rate-1 iO in two different settings: general circuits, and Turing machines with proofs of equivalence in the Extended Frege (EF) propositional proof system. The first setting is not efficiently falsifiable, while the latter is efficiently falsifiable.

Rate-1 iO for circuits. As our first application, we show how to use our master theorem to achieve a rate-1 iO scheme for circuits. We say that an iO scheme achieves rate-1 if the obfuscation size of any circuit C measured in bits is $(1 + o(1))|C| + \mathsf{poly}(\lambda, n)$, where the additive term is a fixed polynomial.[7]

The best known result from prior work [BV15, AJS17, JLL23, JJMP25] achieves rate-1/2. The work of [JJMP25] additionally shows how to achieve rate better than 1/2 for specific circuits (e.g., null iO). We prove the following result:

Theorem 1.2 (Rate-1 iO for Circuits (Informal; see Theorem 5.1 and Theorem 10.22))**.** *Assuming the subexponential version of Assumption 1 as in Remark 1 with theory* $\mathcal{T} = \mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"$+$ *crypto, and*

- *subexponentially secure iO for circuits,*
- *subexponentially secure Learning with Errors (LWE),*

there exists a rate-1 iO scheme for circuits. That is, there exists a fixed polynomial p *such that for a circuit* $C : \{0,1\}^n \to \{0,1\}$*, the obfuscation size is* $(1 + o(1))|C| + p(n, \lambda)$*.*

Our construction follows the template discussed earlier where we first prove security for the setting where we have a compressed description of one of the circuits and then use our master theorem to lift the result to the general setting. We note that the above theorem relies on subexponential security of PCGs (see Remark 1) due to the fact that iO for general circuits is not an efficiently falsifiable notion. The results presented in the remainder of this section, in contrast, rely only on polynomial security of PCGs.

See Section 2.2.1 for an overview of our construction.

Rate-1 iO for uniform EF-equivalent Turing machines. We next investigate the problem of constructing rate-1 iO for Turing machines. Here, we say that an obfuscation scheme is rate-1 if the obfuscation size of a machine M is $(1 + o(1))|M| + \mathsf{poly}(\lambda)$. Importantly, we require that the obfuscation size be independent of the input length.

Following a recent line of work on obfuscating TMs with *unbounded* input length [JJ22, MDS25, JJMP25], we focus on the following notion of security that we refer to as EF-iO: obfuscations of two TMs are indistinguishable if the TMs are equivalent, and admit a proof of equivalence in the EF proof system that can be uniformly generated.[8] The key observation in this line of work is that EF proofs of equivalence can be used to avoid an exponential (in the input length) number of hybrids in the security proof which is necessary to handle unbounded input length.

Assuming iO for circuits (of "small" size independent of input length) and other standard assumptions, these works construct EF-iO for TMs with unbounded input length.[9] The size of the obfuscation in these constructions, however, grows with the size of the machine that generates the EF proof, which may grow polynomially in the size of the machines we wish to obfuscate. Thus, these schemes achieve poor rate.

We show how to remove this dependence on the size of the EF-proof, and in particular, achieve rate-1 EF-iO.

[7] We emphasize that the evaluation time of the obfuscation can be polynomial in the circuit size. Indeed, a recent work of [LMOP26] proves a negative result for iO schemes that achieve rate-1 in *runtime*.

[8] We say that π_{EF} is a uniform proof of equivalence for a pair of machines M_0, M_1 if on input n, it outputs an EF proof that for all $x \in \{0,1\}^n$, $M_0(x) = M_1(x)$ in time $\mathsf{poly}(n)$. We refer the reader to Definition 3.17 for more details.

[9] For general TMs, presently, no solutions are known based on standard assumptions. If we allow for non-standard assumptions, then solutions are known based on differing-input obfuscation [ABG+13, IPS15].

Theorem 1.3 (Rate-1 EF-iO (Informal; See Theorem 5.2 and Theorem 10.24))**.** *Assuming Assumption 1 with* $\mathcal{T} = \mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} +$ *crypto,* $\mathsf{prBPP} = \mathsf{prP}$*, an* EF*-iO scheme for Turing machines, and PRGs, there is a rate-1* EF*-iO scheme for Turing machines.*

We highlight two important observations behind our result. First, the aforementioned works use EF proofs only in the security proof, and therefore the dependence on their size stems from padding. Thus, we are able to apply our general methodology for compressing padding to this problem. Second, the security game for EF-iO is efficiently falsifiable. This allows us to use *polynomial*-hardness rather than $2^{\text{input-length}}$ hardness of the PCG. This is crucial to setting the parameter ℓ in Theorem 1.1 independent of any input bound.

A note on universality. We do not know how to prove a similar result as in Theorem 1.3 for *general* Turing machines (i.e., not just EF-equivalent ones) even assuming the existence of iO for TMs with unbounded input length. The main challenge in applying our techniques to this setting stems from the non-falsifiability of the iO definition for general TMs. Nevertheless, we are able to show that the construction in Theorem 1.3 is “universal” in the following sense: if there *exists* any rate-1 iO scheme for TMs whose correctness is provable in Cook’s Theory PV [Coo75], then the scheme in Theorem 1.3 is a rate-1 iO scheme for *all TMs.*

1.2.2 Succinct Witness Encryption

Witness encryption (WE) [GGSW13] allows encrypting a message to an NP statement x such that anyone holding a valid witness w can decrypt the ciphertext. We consider the problem of *succinct* witness encryption where the size of the ciphertext is only a polynomial in the security parameter, and in particular, independent of both the instance and the witness length.

For general NP languages, the best known constructions achieve ciphertext size that grows polynomially with the witness size [JJMP25]. To achieve shorter ciphertexts, recent works [JJ22, JJMP25] consider a subclass of $\mathsf{NP} \cap \mathsf{coNP}$ where every false instance admits an EF proof of non-membership in the language. An example of such a language is the Decisional Diffie-Hellman (DDH) language (see [JKLV24]). The ciphertext size in the schemes of [JJ22, JJMP25] grows with the size of the EF proof, which, in turn, grows with the size of the coNP witness and the instance. Once again, this is an artifact of padding where the ciphertexts in the real scheme are padded to grow with the size of the EF proof which is only used in the security proof.

We show how to avoid padding and achieve succinct WE for the above class of languages.

Theorem 1.4 (Succinct WE for languages with proofs of non-membership (Informal; See Theorem 5.12, and Theorem 10.24 for the underlying rate-1 EF-iO with PCG against bounded arithmetic))**.** *Assuming Assumption 1 with* $\mathcal{T} = \mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} +$ *crypto,* $\mathsf{prBPP} = \mathsf{prP}$*, an* EF*-iO scheme for Turing machines, and PRG, there exists a succinct witness encryption scheme with ciphertexts of size* $\mathsf{poly}(\lambda)$ *for languages with* EF *proofs of non-membership.*

As a corollary, we obtain succinct WE for the DDH language with $\mathsf{poly}(\lambda)$ size ciphertexts.

Extractable witness encryption. Standard WE guarantees semantic security only for ciphertexts encrypted with false statements. Next, we consider the stronger notion of *extractable* WE [GGSW13] which guarantees that a distinguisher for two ciphertexts encrypted with a true statement x can be used to recover a witness for x. This is a strong primitive, and is subject to known barriers for general NP languages [GGHW14]. However, for UP languages (i.e., where every true statement has a unique witness), extractable

WE is known from iO for circuits [BCP14, GGSW13]. These schemes, however, do not achieve succinct ciphertexts. In particular, the ciphertext size grows polynomially with the instance and witness size.

We consider a subclass of UP with relation R where every true instance x has an EF *proof of unambiguity*, i.e. a proof that "if $R(x, w_1) = R(x, w_2) = 1$, then $w_1 = w_2$". For example, languages with EF-provable extraction, such as the language of LWE samples $R_{\mathbf{A}} = \{(x = \mathbf{sA} + \mathbf{e}, w = (\mathbf{s}, \mathbf{e}))\}$ for a fixed public matrix $\mathbf{A}$ with trapdoor, naturally admit an EF proof of unambiguity.

Our next result is a construction of *succinct and extractable* WE for the above class of languages with total relations, i.e., where every statement is true. More generally, we construct a succinct WE for the above class of languages with an extraction guarantee for ciphertexts encrypted with true statements (but no semantic security guarantee for ciphertexts encrypted with false statements).[10]

Theorem 1.5 (Succinct Extractable WE for Total Languages with proof of unambiguity (Informal; See Theorem 5.12, and Theorem 10.24 for the underlying rate-1 EF-iO with PCG against bounded arithmetic))**.** *Assuming Assumption 1 with* $\mathcal{T} = \mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"$+$ *crypto,* $\mathsf{prBPP} = \mathsf{prP}$*, an* EF*-iO scheme for Turing machines, and PRG, there exists a succinct and extractable WE for total relations with* EF *proof of unambiguity, i.e., every* $x \in \mathcal{L}$ *has a proof of unambiguity.*

The key observation behind the above result is that for any true statement x with a unique witness w, the proof of unambiguity implies a proof of equivalence between the following two programs: $R(x, w') = 1$, and $R(x, w') = 1 \wedge w_i = w'_i$. We first combine this observation with the extraction technique of [BCP14] to build a non-succinct extractable WE scheme from EF-iO (as opposed to general purpose iO). The main reason this step yields a non-succinct solution is that the size of the proof (which we only use in the security proof) grows with the witness size. We then apply our general compression technique to remove this dependence on the witness size and obtain a *succinct* solution.

1.2.3 SNARGs with Short CRS via LWE

A SNARG [Mic94] is a non-interactive, computationally sound proof system for NP in the common reference string (CRS) model, in which the proof is much shorter than the corresponding witness.[11] We say that a SNARG scheme has a *short* CRS if the size of the CRS is independent of both the instance and witness length m.

SNARGs for NP with short CRS are subject to barriers both in the adaptive and non-adaptive setting [GW11, CJ25]. We consider a subclass of NP where these barriers do not apply: languages in $\mathsf{NP} \cap \mathsf{coNP}$ where every false instance admits an EF proof of non-membership in the language (e.g., DDH). Furthermore, to demonstrate the versatility of our techniques, we focus on constructions from assumptions *not* known to imply obfuscation.

The work of [JKLV24] constructed SNARGs for the above class of languages from the hardness of LWE. While their scheme achieves succinct proofs, the CRS grows with the size of the EF proof of non-membership; this proof, in turn, grows with the size of the instance and the coNP witness. We provide a construction of SNARGs for the same class of languages with a *short* CRS of size $\mathsf{poly}(\lambda)$.

Theorem 1.6 (SNARGs with Short CRS for EF-friendly languages (Informal; See Theorem 7.2 and Theorem 10.26))**.** *Assuming polynomial hardness of Assumption 1 with* $\mathcal{T} = \mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"$+$ *crypto,*

[10] A recent work of [JLP26] investigates a different, but related problem of SNARGs with knowledge (i.e., SNARKs) for UP languages with proofs of unambiguity. Besides the difference in notions, their setting is different from ours in that they use a large CRS. In contrast, our WE scheme enjoys full succinctness. See Section 1.6 for more details.

[11] We focus on SNARGs with non-adaptive soundness where the adversary chooses the statement before the CRS is fixed.

unleveled FHE, and BARGs, there is a SNARG scheme with short CRS for languages with EF *proofs of non-membership.*

Once again, we leverage the fact that the EF proof only appears in the security proof of [JKLV24] and thus, the large CRS is an artifact of padding. Furthermore, by leveraging the fact that the definition of SNARGs is falsifiable for the above class of languages, we are able to rely on the polynomial hardness of Assumption 1. See Section 2.2.4 for details.

1.2.4 Succinct Multi-Input Functional Encryption

Multi-input functional encryption (MIFE) [GGG+14] is a generalization of functional encryption [SW05, BSW11] where the decryption key is associated with a multi-input function. In an MIFE scheme for n-ary functions $\mathcal{F}$, one can separately encrypt an input x_i to each position i. Further, given the master key, one can generate secret keys associated with functions $f \in \mathcal{F}$. Given a secret key for a function f and a tuple of ciphertexts, one for each position i encrypting x_i, one can recover the output $f(x_1, \ldots, x_n)$. We consider the generalized setting in which each position i is associated with a different encryption key ek_i.

MIFE provides indistinguishability-based security similar to standard encryption except that for every function appearing in the key queries of the adversaries, we require its output to be the same on every "left" and "right" query. More specifically, for any query bound q, let $\{x^0_{i,q_i}\}$ and $\{x^1_{i,q_i}\}$ denote the left and right challenge messages, where $i \in [n]$. We require that the sets of ciphertexts $\{\mathsf{ct}^0_{i,q_i} = \mathsf{Enc}(\mathsf{ek}_i, x^0_{i,q_i})\}$ and $\{\mathsf{ct}^1_{i,q_i} = \mathsf{Enc}(\mathsf{ek}_i, x^1_{i,q_i})\}$ are indistinguishable given decryption keys $\{\mathsf{sk}_{f_q}\}$ and corrupted encryption keys $\{\mathsf{ek}_{i^*}\}$, as long as the following "non-triviality condition" holds: honest decryptions using sk_{f_q} over the challenge ciphertexts and honestly-generated ciphertexts using $\{\mathsf{ek}_{i^*}\}$ do not distinguish the two sets.

Succinct MIFE. In this work, we consider the problem of constructing MIFE schemes which are *succinct in the arity* n, i.e., the key and ciphertext sizes are independent of n. This notion only makes sense in the Turing machine setting, where the function f may have description size smaller than n. All prior constructions of MIFE [GGG+14, GJO16, MDS25] have key sizes that grow with $\mathsf{poly}(n)$, even when the supported functions are Turing machines.[12]

Similar to iO, the notion of MIFE is non-falsifiable in general. We focus on the falsifiable regime and consider the notion of EF-MIFE [MDS25] – a variant of MIFE for Turing machines where security is only guaranteed when the non-triviality condition additionally admits an EF proof. In this setting, the work of [MDS25] achieves a non-succinct solution where the size of encryption and decryption keys grows with the arity n. We show how to achieve a *succinct* solution:

Theorem 1.7 (Succinct EF-MIFE (Informal; See Theorem 6.1 and Theorem 10.19))**.** *Assuming Assumption 1 with* $\mathcal{T} = \mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"$+$ *crypto, a succinct* EF*-iO scheme for Turing machines, succinct NIWI, and PKE, there exists a succinct bounded-query* EF*-MIFE scheme for Turing machines, where the key and ciphertext sizes are independent of the arity, query bound, and* EF *proof size.*

Our construction achieves a selective notion of security against any polynomially-bounded number of queries; see Section 6 for details. We build upon the original scheme of [GGG+14], which requires padding the scheme parameters to the total size of the challenge message queries. We extend their construction to an EF-MIFE scheme for Turing machines and reduce the padding size to $\mathsf{poly}(\lambda)$ assuming that the adversary's challenge is compressible. Combined with our master theorem, we obtain a succinct EF-MIFE

[12] Under the non-standard assumption of differing-input obfuscation, succinct MIFE schemes are known [BGJS15].

scheme for Turing machines with bounded-query security. An overview of our construction is presented in Section 2.2.5.

1.3 On the Existence of Proof Complexity Generators

In this section, we discuss proof complexity generators in more detail. We first note that Assumption 1 is not known to be false even if we significantly strengthen it by removing the provability requirement, or equivalently, setting the theory $\mathcal{T}$ to be the theory of all true sentences. Below we state this *strong* formulation of PCGs:

Assumption 2 (*Strong* Proof Complexity Generators)**.** There is a uniform family of circuits $C_m : \{0,1\}^\ell \to \{0,1\}^m$ such that for every polynomial time algorithm $V(y,\pi)$, the following two conditions cannot hold simultaneously:

- (*Completeness*). For infinitely many m, there exists a string $y \in \{0,1\}^m \setminus \mathsf{Range}(C_m)$ and a string $\pi \in \{0,1\}^t$ such that $V(y,\pi) = 1$.
- (*Soundness*). For every m, every $y \in \mathsf{Range}(C_m)$, and every $\pi \in \{0,1\}^t$, $V(y,\pi) = 0$.

It is not hard to observe that a family C_m is exactly a proof complexity generator against any Cook-Reckhow proof system (see Section 4.1.1 for details). Therefore, this assumption strengthens Krajíček's conjecture on the existence of PCG against any PPS [Kra24] to the almost everywhere regime.

Assumption 2 appears to be a strong assumption; for instance, it immediately implies that $\mathsf{NP} \neq \mathsf{coNP}$ (and indeed $\mathsf{NP} \neq$ i.o.-coNP), as otherwise every tautology admits short propositional proofs [CR79]. However, to the best of our knowledge, it is consistent with our knowledge and almost all standard assumptions.

Evidence. Indeed, there has been exciting progress on proving weaker variants of Assumption 2. Proof complexity generators are closely related to the range avoidance problem (see [Kor25]):

- Given a circuit $C : \{0,1\}^n \to \{0,1\}^m$ for $m > n$, output a string $y \in \{0,1\}^m \setminus \mathsf{Range}(C)$.

Ren, Santhanam, and Wang [RSW22] proved that the range avoidance problem is hard against nondeterministic algorithms if and only if every propositional proof system admits a non-uniform proof complexity generator. Following [ILW23], Chen and Li [CL24] proved that the range avoidance problem is hard against nondeterministic algorithms under hardness of lattice problems against coNP. This was subsequently improved by Ren, Wang, and Zhong [RWZ26] and independently Ilango [Ila25b], building the same results under demi-bits generators (see [Rud97]).

These results provide strong evidence that for every propositional proof system P, there is a non-uniform PCG against it. Ilango [Ila25b] further proved (under demi-bits generators) that there is a *uniform* PCG against *any* propositional proof system in the CRS model (the CRS is given to both the generator C_m and the verifier V) with extremely high probability. Formally:

Theorem 1.8 ([Ila25b])**.** *For some $\ell = m^{\Omega(1)}$, $k = m^{O(1)}$, and $t = m^{\omega(1)}$, the following is true assuming Rudich's demi-bits [Rud97] secure on almost every input length: There is a uniform family of circuits $C_m : \{0,1\}^k \times \{0,1\}^\ell \to \{0,1\}^m$ such that for every polynomial time algorithm $V(y,\mathsf{crs},\pi)$, the following two conditions cannot hold simultaneously:*

- *(Completeness). For infinitely many m, over a random $\mathsf{crs} \in \{0,1\}^k$, with probability at least $2^{-\Omega(m)}$, there exists a string $y \in \{0,1\}^m$ and a proof $\pi \in \{0,1\}^t$ such that $V(y,\mathsf{crs},\pi) = 1$.*

- *(Soundness). For every string* $\mathsf{crs} \in \{0,1\}^k$*,* $y \in \{0,1\}^m$*, and* $\pi \in \{0,1\}^t$*, if* $V(y, \mathsf{crs}, \pi) = 1$*, then* $y \notin \mathsf{Range}(C_m(\mathsf{crs}, \cdot))$.

This PCG is insufficient for our purposes as the length k of the CRS grows with the output length m; it shows the hardness of certifying any incompressible string *relative to* a pre-determined CRS that is longer than the string. Our applications require Assumption 2, which is exactly Theorem 1.8 with no CRS. This suggests that, either Assumption 2 is true, or there is a somewhat surprising difference between the plain and CRS model in the context of proving incompressibility of strings.

Skepticism. On the other hand, some complexity theorists[13] expressed skepticism on the validity of Assumption 2. Indeed, one could consider a world where not only Assumption 2 is false, but for every PCG C_m, there is a *deterministic* algorithm that breaks it. Namely, there is a polynomial time algorithm A satisfying both:

- (*Completeness*). For infinitely many m, $A(1^m)$ outputs a string $y \in \{0,1\}^m \setminus \mathsf{Range}(C_m)$.
- (*Soundness*). For every m, $A(1^m)$ either outputs $\perp$ or outputs $y \notin \mathsf{Range}(C_m)$.

The problem of designing such an algorithm is referred to as solving the range avoidance problem for *uniform input circuits*. Note that this immediately breaks Assumption 2 by fixing $V(y, \pi)$ that ignores π and accepts if and only if $A(1^m) = y$.

The feasibility of the uniform range avoidance problem is an important open problem in complexity theory. Korten [Kor22] observed that many *explicit construction problems*, including circuit lower bounds, the construction of rigid matrices and strong extractors, reduce to solving the uniform range avoidance problem in polynomial time (see [Kor25] for a survey). This implies that designing such an algorithm A will automatically resolve many major open problems at once. However, this does not necessarily mean that such an algorithm should not exist – at this point, all consequences of such an algorithm appear to be believable.

Progress on Refuting Assumption 2. Despite shared skepticism, there has been limited discussion and progress on refuting Assumption 2, or showing the feasibility of the range avoidance problem for uniform input circuits.

- Ren, Santhanam, and Wang called for research on the existence of such an algorithm A [RSW22, Hypothesis 1.19], and proved that the existence of A is equivalent to the existence of algorithms generating strings with high time-bounded Kolmogorov complexity (i.e. $\boldsymbol{K}^t$ complexity). The same question is emphasized again by Ilango, Li, and Williams [ILW23] (and subsequently [RWZ26, Ila25b]) in light of the hardness of range avoidance (for non-uniform input circuits).

- In a recent work, Korten and Santhanam [KS25] proved that, under a natural hardness assumption, there is a deterministic algorithm that takes 1^m and generates an m-bit string with almost maximum $\boldsymbol{K}^t$-complexity. It is not hard to observe that the hardness assumption in [KS25] almost immediately refutes Assumption 2 (see Section A for details). Thus, while the result of Korten and Santhanam clarifies the world where Assumption 2 is false, it adds limited understanding to the validity of Assumption 2 itself.

[13] Based on personal communication with Lijie Chen and Rahul Santhanam, separately with Rahul Ilango, separately with Oliver Korten, and separately with Ryan Williams, who shares similar skepticism on Assumption 2.

A Random Oracle Heuristic. It is clear that by choosing a weak mathematical theory $\mathcal{T}$, Assumption 1 could be more plausible than Assumption 2. Here, we consider a natural argument[13] against Assumption 2, which shows the tractability of the uniform range avoidance problem via the Random Oracle Heuristic, and explain why the same argument may not apply to Assumption 1.

Remark 2 (Random Oracle Heuristic)**.** Fix a PCG with compression rate $\ell(m)$. Suppose that the algorithm A has access to a random oracle $O : \{0,1\}^m \to \{0,1\}^m$. On any input 1^m, it simply outputs $O(1^m)$. Note that it can produce a correct answer with probability $1 - 2^{\ell-m}$ when given 1^m. For $\ell \leq (1 - \Omega(1)) \cdot m$, the expected number of input lengths that it fails is

$$\sum_{m=1}^{\infty} 2^{\ell-m} \leq \sum_{m=1}^{\infty} 2^{-\Omega(m)} = O(1).$$

Therefore, by instantiating a complicated hash function,[14] the algorithm A may make mistakes on only $O(1)$ input lengths – it then suffices to output $\perp$ on small input lengths.

By choosing a specific theory $\mathcal{T}$ that is sufficiently weak, Assumption 1 seems to be resistant to the random oracle heuristic: When we replace the random oracle O by a concrete hash function h, it has to be chaotic enough to generate "seemingly random strings", yet it also has to be structured enough so the fact that $A^h(1^m) \notin \mathsf{Range}(C_m)$ is provable in a weak mathematical theory $\mathcal{T}$. These two requirements seem to be self-contradicting, and it might be too good to be true from a proof complexity point of view.

The Advantage of Bounded Arithmetic. Recall that we instantiate Assumption 1 with APC_1 (with additional axioms) – a weak yet well-studied theory in bounded arithmetic (see next subsection for details). Choosing a theory in bounded arithmetic rather than, say, ZFC, makes Assumption 1 more plausible, as we now argue.

For a bounded theory $\mathcal{T}$, there is a PPS $P_{\mathcal{T}}$ that polynomially simulates any PPS with $\mathcal{T}$-provable soundness. This observation is due to Cook [Coo75] (see also [Kra95, Section 15]), who proved that EF polynomially simulates any PPS with PV-provable soundness. For our theory of feasible cryptography (APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$" + cryptographic security axioms), $P_{\mathcal{T}}$ can be constructed by extending Jeřábek's WPHP-Frege system [Jeř05, Section 5.2] with appropriate "$\mathsf{prBPP} = \mathsf{prP}$" and cryptographic axiom schemes; this is rather standard and is left to interested readers.

Therefore, Assumption 1 instantiated with a bounded theory assumes PCG against a *single* PPS. Despite skepticism on uniform PCG against any PPS (that could potentially capitalize on any unknown mathematical structure), it would be quite surprising if there is no uniform PCG against a single PPS related to a mild extension of APC_1 with (deliberately designed) limited power.[15] Our results thus show a win-win between proof complexity and cryptography: Either (a mild extension of) APC_1 has surprising proof complexity strength, or our cryptographic constructions are secure.

[14] One crucial observation is that to instantiate it with a concrete hash function h, the running time of the hash function must exceed the running time of the proof complexity generators. This is because otherwise, we know unconditionally that the output of the hash function has low time-bounded Kolmogorov complexity, and is thus compressible by the $\boldsymbol{K}^t$ generator. Specifically, letting $t(m)$ be the running time of the hash function, the generator C_m takes as input the description of a Turing machine M, runs $M(1^m)$ for $t(m)$ steps and outputs $M(1^m)$. Then it follows that for $\ell = \omega(1)$, $A^h(1^m)$ is always in the range of C_m.

[15] Indeed, for weak PPS such as DNF-Resolution, there is a strong proof complexity generator based on lossless expanders [Raz15]; if such lossless expanders can be uniformly constructed, it will be a uniform PCG against DNF resolution. Progress has also been made to construct PCGs against AC^0-Frege [Kha22].

1.4 On the Theory APC_1

The theory APC_1 [Jeř07a] belongs to a collection of theories known as Bounded Arithmetic (see, e.g., [Kra95, Kra19, CN10, Oli25, Li25]) corresponding to complexity classes, ranging from AC^0 to PSPACE and beyond (see [CN10] for a comprehensive exposition).

One representative theory is Cook's theory PV_1 capturing *deterministic polynomial-time* reasoning [Coo75]; at a high level, it means that only polynomial time functions and induction on polynomial time predicates are allowed in mathematical proofs. This restriction makes it impossible to prove several seemingly natural statements:[16]

- Parikh's theorem [Par71] states that bounded theories, including PV_1, do not prove $\forall x\ \exists y\ y = 2^x$. This is because the length of y is exponential in the length of x, and thus it cannot be generated by any polynomial time algorithm.
- For a similar reason, PV_1 cannot prove the Weak Pigeonhole Principle (WPHP) under cryptographic CRHFs: for every circuit $C : \{0,1\}^n \to \{0,1\}^{n-1}$, there are $x \neq y$ such that $C(x) = C(y)$ [Jer07b].

On the other hand, PV_1 is known to formalize many interesting results in theoretical computer science, including the celebrated PCP theorem [Pic15] and correctness of many cryptographic primitives [JJ22, JKLM25, JJMP25, Mat25].

The theory PV_1 is insufficient for our purpose as it does not support mathematical tools to reason about *probabilistic polynomial time algorithms*. The standard solution to this issue is to work with Jeřábek's theory APC_1, which extends PV_1 with the dual Weak Pigeonhole Principle (dWPHP):

$$\forall \text{ circuit } C : \{0,1\}^n \to \{0,1\}^{n+1}\ \exists y \in \{0,1\}^{n+1}\ \forall x \in \{0,1\}^n\ C(x) \neq y.$$

This (mild) extension allows natural formalization of approximate acceptance probability of circuits and proves a vast majority of probability theory, making it possible to formalize security proofs of cryptographic constructions. Building on Jeřábek's results, a comprehensive toolkit for the purpose of formalizing cryptography is developed in [HJLM26], including a formalization of indistinguishability of distributions, search security games, and a list of inner theorems of $\mathsf{APC}_1 +$ "prBPP $=$ prP" related to these concepts.[17]

It is worth noting that APC_1 does not significantly exceed the power of PV_1. Both Parikh's theorem and the unprovability of WPHP hold for APC_1 as well [Jer07b]. Moreover, techniques used to show unprovability of complexity lower bounds in PV_1 [PS21] are known to adapt to APC_1 and even $\mathsf{APC}_1 +$ crypto [LO23]. Therefore, instantiation of Assumption 1 with $\mathcal{T} := \mathsf{APC}_1 +$ crypto seems more plausible than, say, $\mathcal{T} := \mathsf{ZFC} +$ crypto, as we have a very limited understanding of the power of ZFC.

Remark 3 (Formalization of axioms)**.** We also stress that as bounded theories are much weaker than usual mathematical theories, the way to formalize a statement may change its provability. For instance, Parikh's theorem shows that the existence of $x \mapsto 2^x$ cannot be proved in bounded arithmetic. However, the existence of $X \mapsto 2^{|X|}$, i.e., exponential function with unary-encoded input, is provable in PV_1. We stress that our formalization of security axioms is straightforward, and our results do not rely on any specific formalization details. Consequently, similar results should hold under other natural formalizations. This is crucial as we want our assumptions to capture intrinsic limitations of feasible proofs rather than the complication of a specific formalization.

[16] These unprovability results do not mean that PV_1 is not a "good" mathematical theory – the theory is designed to capture "polynomial-time reasoning" and the unprovability results are necessary consequences of the goal.

[17] Recently, a new theory called APX_1 has been introduced to explore the minimal theory for probabilistic polynomial time reasoning [CLOW26]. This theory is an extension of PV_1 and is (under plausible assumptions) strictly weaker than APC_1. At this point, we stick to the more well-established theory APC_1 for convenience.

1.5 Future Directions

In this work, we establish a new connection between proof complexity and cryptography by using PCGs to remove padding in cryptographic applications. We view the applications presented in this work as a preliminary demonstration of the applications of PCGs, and a more extensive exploration of the same is warranted. In the following, we discuss some future directions.

Weakening the assumptions. A natural question is whether one can use weaker assumptions from proof complexity to instantiate our master theorem. Our lifting theorem currently relies on the existence of uniform PCGs in the plain model. On the other hand, [Ila25b] recently constructed PCGs in the CRS model (where the CRS size grows with the PCG output length) under the weaker assumption of demi-bits generators [Rud97].[18] Exploring applications of this weaker notion within our framework is an interesting avenue of future research.

Rate-1 iO and succinct EF-MIFE. We highlight two questions left open by our results. First, although we construct a rate-1 iO scheme for all circuits, our result does not extend to all Turing machines. The main obstacle stems from the non-falsifiability of the iO definition, preventing an application of our master theorem in any meaningful parameter regime without violating succinctness.[19] The possibility of devising an alternative construction that overcomes this barrier remains open.

Second, our EF-MIFE scheme only guarantees security for an a priori bounded number of queries per input position. Interestingly, this bound is only reflected in the running time of our scheme, but not in the size of the parameters of the scheme. Presently, known schemes that handle an unbounded number of queries do so via an exponential number of hybrids resulting in non-succinct solutions that grow with the number of inputs. Thus, succinct EF-MIFE with security for an unbounded number of queries remains an open question.

On security axioms for feasible cryptography. In the instantiation of Assumption 1, we use security axioms for several primitives, including PRG, FHE, and EF-iO [JJMP25]. In particular, we need to fix a concrete construction of these primitives and add axioms that state the security of the fixed construction, rather than arbitrarily choose a construction. This is to prevent a contrived attack discussed in [HJLM26, Section 6.4.2] — certain constructions with planted loopholes could unconditionally falsify the unprovability assumption.

A natural direction, which we defer to future work, is to replace these axioms with more fundamental assumptions such as LWE or Diffie-Hellman. The rationale is that the proof complexity strength of these concrete assumptions is easier to study compared to security of concrete cryptographic constructions. To achieve this, we need to carefully formalize the security analysis of the cryptographic constructions in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$, possibly with minor modification to the schemes to avoid infeasible arguments. Indeed, a similar formalization has been carried out in [HJLM26]: the security proof of the SNARG construction in [JKLV24] (with certain modification) is formalized within $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. As a result, the unprovability assumption involves only fundamental assumptions.

We leave such a formalization to future works, with the consideration that it can be largely decoupled from our main contribution — the application of proof complexity generators. Given the tools developed

[18] Intuitively, a demi-bits generator asserts the hardness of certifying many incompressible strings (“average-case” hardness), while a PCG asserts the hardness of any single string (“best-case” hardness).

[19] Note that in the case of circuits, we are able to overcome this obstacle.

in [HJLM26], we see no fundamental barrier preventing us from formalizing the security of all used constructions (possibly with minor modification), so the theory $\mathcal{T}$ in Assumption 1 can be replaced by APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$" + LWE and other standard assumptions.

1.6 Related Work

On necessary padding. The work of Holmgren [Hol15] demonstrates an example use case of iO where the padding of the underlying circuit is *necessary*. In more detail, they define two samplers Samp_0 outputting (aux_0, C_0) and Samp_1 outputting (aux_1, C_1) where the following hold:

- distributions $(\mathsf{aux}_0, \mathsf{iO}(1^\lambda, C_0))$ and $(\mathsf{aux}_1, \mathsf{iO}(1^\lambda, C_1))$ are *distinguishable*,
- but $(\mathsf{aux}_0, \mathsf{iO}(1^\lambda, C_0||0^s))$ and $(\mathsf{aux}_1, \mathsf{iO}(1^\lambda, C_1||0^s))$ are *indistinguishable*, where $s = \mathsf{poly}(\lambda)$.

As we discuss in Section 2.3, our master theorem does *not* contradict this example. In short, since security cannot even be established in the compressed setting (iO without padding), our master theorem does not apply to this setting.

Extractable SNARGs for EF-UP. A recent work of [JLP26] constructs SNARGs with an argument of knowledge property (SNARK) for languages $\mathcal{L}$ with an EF proof of unambiguity. In addition to achieving extraction for true instances, they also prove a weak notion of "infinite-often soundness" for false instances via a non black-box use of the adversary. However, the CRS size in their scheme is large and grows at least with the length of the witness (in fact, it grows with the size of the adversary, which results in the weak notion of soundness).

Our result on extractable WE in Theorem 1.5 considers the same set of languages. Our focus is on achieving *succinct* ciphertext size, which is more analogous to the problem of SNARGs with short CRS. We do not show semantic security for ciphertexts encrypted with statements $x \notin \mathcal{L}$. It is plausible that the techniques of [JLP26] can be used to achieve semantic security in our setting; we leave this to future work.

High-rate obfuscation. Several works have studied the problem of minimizing the size of the obfuscation output by iO. Prior works [BV15, AJS17, JLL23, JJMP25] show that it is possible to achieve rate-$1/2$ obfuscation (with a $\mathsf{poly}(\lambda, n)$ additive overhead), where the size of the obfuscation is measured via the length of its description. The work of [JJMP25] additionally shows how to achieve rate-1 null iO.

The work of Ma et al. [MDS25] shows how to achieve *quasi-linear* EF-iO for circuits, where the obfuscation *runtime* is quasi-linear in the size of the original circuit as well as the EF-proof of equivalence. In contrast, our construction outputs an obfuscation whose *description* is rate-1 in the size of the original circuit or machine, and is independent of the proof size. We note, however, that the evaluation runtime of our obfuscation is not quasi-linear in the circuit size, but is polynomial in both the circuit size and the size of the EF-proof. The work of [LMOP26] shows that it is in fact impossible to achieve an iO scheme for circuits whose *runtime* incurs only an additive $O(s/\log s)$ overhead, where s is the size of the circuit.

Succinct cryptography from extractability assumptions. Another approach to achieving succinctness in cryptography is via the use of strong extractability-based assumptions such as differing-input obfuscation. For example, succinct iO and WE [ABG+13, IPS15] and succinct MIFE [BGJS15] are known from (public-coin) differing-input iO. However, such assumptions are subject to strong barriers [GGHW14, BSW16]. Similarly, extractability assumptions have been used in the context of SNARGs to achieve constructions

with short CRS [BCCT12, BCCT13]. Besides being non-standard (and often implausible), a common issue with the use of extractability assumptions is that the reductions are not fully explicit.

Compared to this line of work, our constructions do not rely on any extractability assumptions. Instead, we rely on standard, falsifiable cryptographic assumptions paired with a well-studied proof complexity assumption.

Applications of Proof Complexity to Cryptography. Our work adds to a growing list of recent works demonstrating applications of proof complexity to cryptography. Starting with [JJ22], a recent line of work [JKLV24, MDS25, JKLM25, JJMP25, JLP26] has demonstrated how to use propositional proofs to construct succinct cryptographic primitives such as SNARGs and iO for TMs.

While these works rely on upper bounds in proof complexity, recent works use lower bounds to overcome known barriers in cryptography. The work of [Ila25a] constructs a one-message proof system with no setup that achieves statistical soundness and a weakened form of zero-knowledge, assuming the conjecture of Krajíček and Pudlák [KP89] that no optimal proof system exists. The zero-knowledge guarantee is "effective" in the sense that the existence of a simulator cannot be disproved within a fixed theory such as ZFC. The work of [HJLM26] uses the unprovability of proof complexity lower bounds from feasible cryptography (concretely, APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$" + LWE + SXDH) to construct non-adaptive SNARGs for NP. Here, unprovability from bounded arithmetic can be viewed as uniform proof complexity lower bounds, due to the connection of bounded theories and propositional proofs (see, e.g., [Coo75] and [Kra19, Section 9]).

Extending [Ila25a, HJLM26], our work discovers a novel application of proof complexity lower bounds to addressing the problem of superfluous padding – a barrier to succinctness in cryptography that is orthogonal to those addressed by prior work.

2 Technical Overview

We now give a more detailed account of our techniques, expanding on the high-level overview in Section 1.

2.1 The Master Theorem and Bounded Arithmetic

To explain the intuition behind the master theorem, we start with the easiest setting where $\mathcal{T}$ is the true theory of arithmetic, i.e., every true sentence is in $\mathcal{T}$. Note that in this case, Assumption 1 essentially becomes Assumption 2.

Let $\mathcal{U} \in \mathsf{NP}$ be the language that specifies the valid instances of a security game. Suppose that there is a cryptographic game such that:

- It is *secure* with the restriction that (x, w) is compressible, i.e., $\boldsymbol{K}^t(x\|w) \leq \ell(|x| + |w|)$.
- Assume for contradiction that the cryptographic game is *insecure* without such a restriction.

We will define the following polynomial time algorithm $V(y, \pi)$ that breaks Assumption 2:

- It parses y as (x, w). If w is not an NP witness for $x \in \mathcal{U}$, $V(y, \pi)$ rejects immediately.
- It then parses π as the description of an adversary $\mathcal{A}$ in the security game. It checks whether the adversary wins in the security game with a non-negligible advantage, if so it accepts, and otherwise it rejects.

Note that the simulation of the security games requires randomness. With the assumption that $\mathsf{prBPP} = \mathsf{prP}$, we can implement $V(y, \pi)$ as a deterministic polynomial time algorithm.

To see that the algorithm V breaks Assumption 2, we prove that it is sound and complete:

- (*Completeness*). By the assumption that the security game is insecure, for infinitely many instances (x, w), there are adversaries that win in the security game with a non-negligible advantage. Subsequently, the algorithm V accepts infinitely many strings y.
- (*Soundness*). We need to show that if $V(y, \pi) = 1$, then y is incompressible. Recall that $y = (x, w)$ and π is an adversary in the security game. It can be observed that this is exactly security under compression: if y is compressible, then there is no adversary that wins with a non-negligible probability.

We also note that a similar proof follows for other parameter regimes: if V takes subexponential (or slightly superpolynomial) time, we can derive a similar consequences using subexponentially-secure PCGs or slightly superpolynomially secure PCGs.

This gives a proof for a weaker master theorem that utilizes Assumption 2. To prove Theorem 1.1, we need to further show that if the security under compression is provable in $\mathcal{T}$, then V is $\mathcal{T}$-provably sound. This would require $\mathcal{T}$ to support basic reasoning tools for probabilistic polynomial-time algorithms, as well as $\mathsf{prBPP} = \mathsf{prP}$ as an axiom for the correctness of the derandomization. We verified that both conditions are satisfied if $\mathcal{T}$ is a sound extension of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$".

2.2 Building Primitives assuming Compression

Next, we explain how to leverage the power of the master theorem to build succinct cryptographic primitives. As summarized in the previous section, we aim to build succinct primitives which

- are secure with the restriction that the "challenge" (x, w) is compressible, and
- the security under compression is provable in $\mathcal{T}$, so that an adversary breaking the scheme implies a violation of Assumption 1 against $\mathcal{T}$.

We now describe how we can view each of our applications through the above lens. For brevity, we will focus on showing security under compression, and defer the details of the formalization in $\mathcal{T}$ to Section 10, with a brief discussion in Section 2.2.6.

2.2.1 Rate-1 iO for circuits and EF-iO for TMs

In this section, we discuss how we derive Theorem 1.2 and Theorem 1.3 in more detail, beginning with rate-1 iO for circuits.

Ingredients. Our construction relies on PRGs, as well as the notion of *description-succinct* iO for TMs introduced by [JJMP25] (JJMP). We consider programs with split description consisting of a secret part M and a public part pub, and write $M[\mathsf{pub}]$ to denote the full program. We say that an iO scheme achieves description succinctness if $|\mathsf{iO}(1^\lambda, M[\mathsf{pub}])|$ grows with $|M|$ but not $|\mathsf{pub}|$. The evaluator still requires pub to evaluate the obfuscation, and security holds for all pairs of machines $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$ that are functionally equivalent. JJMP show that this notion can be achieved with $|\mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n)| = \mathsf{poly}(|M|, n, \lambda)$, assuming subexponential iO for circuits and subexponential LWE, where n bounds the supported input length.

Construction. The construction is a simplification of the high-rate scheme of JJMP, based on the "double encryption" trick of [NY90, GGH+13, GGG+14], and is secure when $\boldsymbol{K}^t(C_0\|C_1) \leq \ell(|C_0|+|C_1|) = o(|C_b|)$. The key feature is that ct_1 — a random string of length $\ell(2|C|)$ — serves as a placeholder that will hold a compressed description of $C_0\|C_1$ in the hybrid argument, without contributing to the rate.

Obfuscation algorithm: $\mathcal{O}_\ell(1^\lambda, C, 1^n)$:

- Let s_0 be a PRG seed, and compute $\mathsf{ct}_0 = C \oplus G(s_0)$. Let ct_1 be a random string of length $\ell(2|C|)$.
- Construct the program $P[\mathsf{ct}_0, \mathsf{ct}_1](x)$[a] which does the following:
  - Use s_0 to decrypt ct_0 to obtain C. Output $C(x)$.
- Let $\Gamma \leftarrow \mathsf{iO}(1^\lambda, P[\mathsf{ct}_0, \mathsf{ct}_1], 1^n)$, where iO is a description-succinct iO scheme.
- Output $(\mathsf{ct}_0, \mathsf{ct}_1, \Gamma)$.

Evaluation algorithm: Given $(\mathsf{ct}_0, \mathsf{ct}_1, \Gamma)$ and an input $x \in \{0,1\}^n$, output $\Gamma[\mathsf{ct}_0, \mathsf{ct}_1](x)$.

[a] We additionally need to pad the runtime of P in the obfuscated program to match the runtime of the hybrids (a function of t, but we gloss over this detail here.)

By description-succinctness, the obfuscated program has size $|C| + \ell(2|C|) + \mathsf{poly}(\lambda, n) = |C|(1 + o(1)) + \mathsf{poly}(\lambda, n)$.

Security assuming compression. Suppose that $\boldsymbol{K}^t(C_0\|C_1) \leq \ell(2|C|)$, and let S be a machine witnessing this fact. We invoke the following hybrid argument:

1. Start from $\mathcal{O}_\ell(1^\lambda, C_0, 1^n)$.
2. Replace ct_1 with $S \oplus r_1$ for a uniformly random string r_1. Since S has length at most $\ell(2|C|)$ and ct_1 was already uniformly random, the distribution is identical.
3. Switch to sampling $r_1 \leftarrow G(s_1)$ for a fresh PRG seed s_1. This follows from PRG security.
4. Replace Γ to an obfuscation of the following program $Q[\mathsf{ct}_0, \mathsf{ct}_1]$:
   - Use s_1 to decrypt ct_1 to obtain S.
   - Use S to derive $C_0\|C_1$.
   - Output $C_1(x)$.

   Since $P[\mathsf{ct}_0, \mathsf{ct}_1]$ and $Q[\mathsf{ct}_0, \mathsf{ct}_1]$ are functionally equivalent, this step follows from description-succinct iO security.
5. Switch ct_0 from $C_0 \oplus G(s_0)$ to a fresh random string, by PRG security.
6. Reverse the above steps to reach $\mathcal{O}_\ell(1^\lambda, C_1, 1^n)$.

This completes the proof assuming t-time ℓ-compressibility of $C_0\|C_1$.

Applying our master theorem. The remaining step is to remove the compressibility assumption. The iO security game is naturally a game over the language

$$\mathcal{U} = \{(n, C_0, C_1) \mid \forall x \in \{0,1\}^n,\ C_0(x) = C_1(x)\},$$

which is not efficiently falsifiable — verifying membership requires time 2^n. To handle this, we fix a polynomial $q(n) = cn^c$ and restrict to $\mathcal{U}_q \subseteq \mathcal{U}$, the instances with $|C_b| = q(n)$; see Section 5.1 for the general case. Since $\mathcal{U}_q \in \mathsf{DTIME}[T]$ for $T(m) = 2^{m^{1/c+o(1)}}$, we need subexponential PCG hardness. Assuming 2^{m^δ}-security of the PCG with $\ell = o(m)$, choosing c large enough so that $1/c < \delta$ lets us apply Theorem 1.1 and boost security to all of $\mathcal{U}_q$.

2.2.2 Achieving rate-1 EF-iO

We now show how to achieve rate-1 EF-iO via nearly the same argument (here, EF denotes the Extended Frege propositional proof system, which augments standard Frege with extension axioms; see Section 3.1.2 for details). The construction is essentially identical to the rate-1 circuit case - the main difference being that we instantiate iO with a description-succinct version of EF-iO.

Main ingredient: Fully succinct EF-iO. We say that two machines $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$ are (ρ, τ)-EF-equivalent[20] if there exists a machine π_{EF} of size ρ which, on input n, outputs an EF proof that $M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)$ for all $x \in \{0,1\}^n$, in time $\tau(n)$. We drop τ for brevity and say the machines are ρ-EF-equivalent. We say that an EF-iO scheme is ρ-secure if indistinguishability holds for all ρ-equivalent machine pairs, and *fully succinct* if $|\mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n)| = \mathsf{poly}(\lambda, |M|, \rho)$, independent of both $|\mathsf{pub}|$ and n. As shown by JJMP, assuming subexponential iO for circuits and LWE, there exists a fully succinct ρ-secure EF-iO scheme. Crucially, the real scheme does not need π_{EF} — it only appears in the security proof. Our goal now is to construct a fully succinct EF-iO scheme where the obfuscation size is in fact rate-1 in M, and independent of ρ.

Modified construction. Consider the same construction as for rate-1 iO for circuits, with the following modifications:

- the input C is of the form $M[\mathsf{pub}]$,
- iO is a fully succinct $O(\ell)$-EF-secure scheme,
- the program $P[\mathsf{ct}_0, \mathsf{ct}_1]$ additionally takes pub as public input,
- the evaluator runs $\Gamma[\mathsf{ct}_0, \mathsf{ct}_1, \mathsf{pub}](x)$.

By the efficiency guarantee of iO, we have $|\mathcal{O}_\ell(1^\lambda, M[\mathsf{pub}], 1^n)| = \mathsf{poly}(\lambda, |M|, \ell)$.

Security assuming compression. The analysis proceeds as in the circuit case, with one key difference.

Consider two ρ-EF-equivalent machines $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$ with equivalence machine π_{EF}. Suppose that $\boldsymbol{K}^t(M_0, M_1, \mathsf{pub}, \pi_{\mathsf{EF}}) \le \ell$, witnessed by machine S. The key observation is that compressibility implies EF-equivalence: one can use S to expand and recover $(M_0, M_1, \mathsf{pub}, \pi_{\mathsf{EF}})$, and then run $\pi_{\mathsf{EF}}(n)$

[20] In Definition 3.16, we give a more careful characterization of ρ; this simplified definition suffices for the overview.

to produce an $O(\ell)$-sized equivalence proof on inputs of length n. So $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$ are in fact $O(\ell)$-EF-equivalent.

With this, we run the same hybrids as in the circuit case, modifying Step 4 as follows. Let $Q[\mathsf{ct}_0, \mathsf{ct}_1, \mathsf{pub}]$ be the program that:

- uses s_1 to decrypt ct_1 and obtain S,
- expands S to recover $(M_0, M_1, \mathsf{pub}, \pi_{\mathsf{EF}})$,
- outputs $M_1[\mathsf{pub}](x)$.

Since P outputs $M_0[\mathsf{pub}](x)$ and Q outputs $M_1[\mathsf{pub}](x)$, and the two machines are $O(\ell)$-EF-equivalent, the switch follows from $O(\ell)$-EF-security of iO.

Using our master theorem. The remaining step is again to remove the compressibility assumption. We view ρ-security of EF-iO as a game over the language

$$\mathcal{U}_\rho = \left\{ (1^n, M_0, M_1, \mathsf{pub}) : \begin{array}{c} \exists\, \pi_{\mathsf{EF}} \in \{0,1\}^{\rho(m)} \text{ demonstrating} \\ \rho(k)\text{-equivalence of } M_0[\mathsf{pub}] \text{ and } M_1[\mathsf{pub}], \\ \text{where } k = |M_0| + |M_1| + |\mathsf{pub}|. \end{array} \right\}$$

Unlike the circuit case, this game is *efficiently falsifiable*: given π_{EF}, membership in $\mathcal{U}_\rho$ can be verified in $\mathsf{poly}(n)$ time by running π_{EF} on all input lengths. This means we can work in the polynomial hardness regime of the master theorem and choose $\ell(m) \ll n$ — in contrast to the circuit case which required subexponential PCG hardness. Applying Theorem 1.1 boosts security to all instances in $\mathcal{U}_\rho$, and choosing $\ell(m) = m^{o(1)}$ removes the dependence on ρ for any polynomially-bounded ρ.

Remark 4. Note that we crucially rely on the EF-iO security game being *efficiently falsifiable*: given a witness π_{EF}, membership in $\mathcal{U}_\rho$ can be verified in polynomial time. This allows us to work in the polynomial hardness regime of the master theorem and choose $\ell(m) \ll n$, in contrast to the circuit case which required subexponential PCG hardness.

Weak universality rate-1 EF-iO. The rate-1 EF-iO for TMs satisfies the following bonus form of universality: if there *exists* any rate-1 iO scheme for TMs $\mathsf{iO}(1^\lambda, M, 1^n)$ with a PV proof of correctness, then the EF-iO scheme is secure for *all* pairs of TMs, not just EF-equivalent ones. Letting $\mathcal{O}$ be our rate-1 EF-iO scheme, the argument is the following chain:

$$\begin{aligned} \mathcal{O}(1^\lambda, M_0[\mathsf{pub}], 1^n) &\approx_c \mathcal{O}(1^\lambda, \mathsf{iO}(1^\lambda, M_0[\mathsf{pub}], 1^n), 1^n) \\ &\approx_c \mathcal{O}(1^\lambda, \mathsf{iO}(1^\lambda, M_1[\mathsf{pub}], 1^n), 1^n) \\ &\approx_c \mathcal{O}(1^\lambda, M_1[\mathsf{pub}], 1^n) \end{aligned}$$

where the first and third steps use the PV correctness of iO to invoke EF-iO security of $\mathcal{O}$, and the second uses the security of iO itself. This follows the approach of [JJMP25], who show a similar universality result in the fully succinct setting. Since each hybrid only obfuscates a program of size $|M_b|(1 + o(1)) + \mathsf{poly}(\lambda)$, the real scheme needs only $\mathsf{poly}(\lambda)$ padding, and rate-1 is preserved.

2.2.3 Succinct and Extractable Witness Encryption

In this section, we give an overview of our succinct WE in Theorem 1.4 and the succinct extractable WE in Theorem 1.5. For these results, we will not explicitly use our master theorem, but instead derive these as corollaries of rate-1 EF-iO. In particular, we do not need to formalize these proofs in $\mathcal{T}$.

Consider a language $\mathcal{L} \in \mathsf{NP}$ where every $x \notin \mathcal{L}$ has a size ρ proof of non-membership π_x if $R(x, \cdot)$ and $\mathsf{Null}(\cdot)$ are ρ-EF-equivalent (here, Null is the circuit that always outputs 0). We now argue that rate-1 EF-iO implies succinct WE for such languages. Indeed, consider the following witness encryption construction, which takes as input the security parameter 1^λ, witness length in unary 1^n, instance x and bit b.

$\mathsf{WE.Enc}(1^\lambda, 1^n, x, b)$:

- Construct the program $P_b[x](w)$ which on input w, outputs b if $R(x, w) = 1$, and 0 otherwise.
- Output $\Gamma \leftarrow \mathcal{O}(1^\lambda, P[x], 1^n)$,

$\mathsf{WE.Dec}(\Gamma, x, w)$: Output $\Gamma[x](w)$.

Succinctness follows directly from the fact that $\mathcal{O}$ is rate-1. Semantic security follows from the fact that π_x can be used to construct a $\rho + O(1)$-sized proof that $P_0[x]$ and $P_1[x]$ are equivalent:

- For all w, $P_0[x](w) = \mathsf{Null}(w)$.
- For all w, $P_1[x](w) = R(x, w)$.
- π_x produces a proof that $\mathsf{Null}(w) = R(x, w)$.
- By transitivity, we have that $P_0[x](w) = P_1[x](w)$.

Therefore, one can invoke $\rho + O(1)$-EF-security of $\mathcal{O}$ to complete the proof. Since $\mathcal{O}$ does not grow with ρ, we directly achieve succinctness.

Extractability for x with proofs of unambiguity. We now consider a language $\mathcal{L}$ where $x \in \mathcal{L}$ admits a proof of the fact that "$R(x, w) = 1$ and $R(x, w') = 1$, then $w = w'$." We say that x has a size ρ proof of unambiguity if the above proof can be efficiently generated by a machine π_x of size ρ. We claim that the above witness encryption construction is extractable on these x, i.e., any adversary distinguishing $\mathsf{WE.Enc}(x, 0)$ and $\mathsf{WE.Enc}(x, 1)$ can be used to construct an extractor outputting the witness of x. To do so, we borrow ideas from [BCP14], who use a distinguisher for obfuscations of two machines M_0 and M_1 to extract an input on which the two machines differ, under the assumption that the machines disagree on exactly one input. The key observation is that the EF proof of unambiguity can be used in place of this unique-input assumption, allowing us to carry out an analogous extraction argument within the EF-iO framework.

Let $\widehat{w}$ be the witness of x such that $R(x, \widehat{w}) = 1$. For every index $i \in [n]$, define the program $P'_i[x]$ such that $P'_i[x](w) = R(x, w) \wedge (w_i = 1)$, where w_i denotes the i-th bit of w. We observe that the combination of π_x and $\widehat{w}$ can be used to construct a $\rho + n$ sized proof that $P_{\widehat{w}_i}[x]$ and $P'_i[x]$ are equivalent:

- For all w, $P_0[x](w) = \mathsf{Null}(w)$.
- For all w, $P_1[x](w) = R(x, w)$.

- π_x produces a proof that $R(x,w) = 1 \wedge R(x,w') = 1 \to w = w'$
- $\widehat{w}$ produces a proof that $R(x,\widehat{w}) = 1$
- By transitivity, for each i, we have $R(x,w) = 1 \to w_i = \widehat{w}_i$. For the case $\widehat{w}_i = 0$, this implies $R(x,w) = 1 \wedge (w_i = 1) = \mathsf{Null}(w)$. Similarly, for the case $\widehat{w}_i = 1$, $R(x,w) = 1 \wedge (w_i = 1) = R(x,w)$.
- Therefore, we have $P'_i[x](w) = P_{\widehat{w}_i}[x](w)$

Invoking the $(\rho + n)$-EF-security of $\mathcal{O}$, we can argue that $\Gamma_b \leftarrow \mathsf{WE.Enc}(x, b)$ is indistinguishable to $\Gamma'_i \leftarrow \mathcal{O}(P'_i)$ when $b = \widehat{w}_i$. This immediately provides an extractor for the i-th bit of the witness. An efficient adversary distinguishing Γ_0 and Γ_1 must not distinguish Γ'_i and $\Gamma_{\widehat{w}_i}$ by security of $\mathcal{O}$, and therefore must distinguish Γ'_i and $\Gamma_{\neg\widehat{w}_i}$. Hence, an extractor can easily determine $\widehat{w}_i$ for each i with black box access to the adversary. By running through each index, we conclude with an extractor outputting $\widehat{w}$.

Note that the description size of P'_i is $|R| + |i| = |R| + \log n$. The succinctness thus follows from the fact that $\mathcal{O}$ is rate-1 in $|P'_i|$ and does not grow with the proof size $(\rho + n)$.

2.2.4 SNARGs with Short CRS for Subclass of $\mathsf{NP} \cap \mathsf{coNP}$

We now give a brief overview of the proof strategy for Theorem 1.6 and how our master theorem applies.

Encrypt-hash-and-BARG construction. We first recap the encrypt-hash-and-BARG construction of [JKLV24] (JKLV), which relies on three main tools:

- an unleveled fully homomorphic encryption scheme (FHE),
- a somewhere extractable hash family with local openings (SEH) (Definition 3.8), constructible from FHE,
- a somewhere extractable batch-argument scheme (BARG) (Definition 3.11).

Informally, the CRS of the scheme comprises the following components:

- a FHE ciphertext ct encrypting the *description* of some dummy circuit E, and an FHE evaluation key ek,
- a BARG CRS crs,
- SEH hash keys $\mathsf{hk}_1, \ldots, \mathsf{hk}_{\mathsf{loc}}$ for some $\mathsf{loc} = \mathsf{poly}(\lambda)$. For simplicity, we write $\mathsf{hk} = (\mathsf{hk}_1, \ldots, \mathsf{hk}_{\mathsf{loc}})$, and define hashing and local openings with respect to hk as hashing and opening with respect to each hk_i in parallel.

To prove $x \in \mathcal{L}$ for an NP relation R, instance x, and witness w, the prover does the following:

- Compute $R(x,w)$. Let τ denote the wire values of this computation.
- Evaluate the dummy circuit E *homomorphically* on τ to obtain wire values τ'.
- Hash the wire values τ and τ' to obtain hash value h.

- Compute a batch proof π that for every gate g in the computation of C and E, the prover has local openings from h to wire values w_1, w_2, w_3 such that the gate computation is satisfied.
- Output $(h, \pi, \gamma_{\mathsf{out}})$, where γ_{out} is an opening from h to the output wire of C.

The verifier checks that the BARG proof accepts, the openings are valid, and the output wire value is 1.

Recap of general soundness analysis for EF-friendly languages. Consider a language $\mathcal{L}$ given by relation R where every $x \notin \mathcal{L}$ has a length L EF proof of non-membership. From a bird's eye view, one can break down the soundness analysis of [JKLV24] as follows: fix some $x \notin \mathcal{L}$ and a cheating prover who creates accepting proofs with probability ε.

1. For any cheating prover $\mathcal{P}$, he continues to produce accepting proofs with probability $\approx \varepsilon$ even when the SNARG CRS is generated with an FHE ciphertext ct containing the description of any fixed circuit E_x. This follows straightforwardly from the semantic security of the FHE scheme. We say that the prover produces accepting proofs *relative to* E_x.
2. Given a prover $\mathcal{P}$ who creates accepting proofs with respect to E_x, one can construct an object known as a local assignment generator [PR17][21] for E_x. This analysis is rather involved, and relies on BARG and SEH security.
3. For any x for which there exists a proof π_x that $R(x, w) = 0$ for all w, there exists an efficiently computable circuit E_x of size $\mathsf{poly}(|\pi_x|)$ for which there is *no* local assignment generator.

Since E_x in Step (3) can have an $O(L)$-length description in general, the real construction must encrypt a dummy string of length $O(L)$ as the FHE ciphertext.

Note that the above soundness analysis accommodates *any* description of E_x, and hence allows one to embed any (possibly small) program that outputs E_x. We now use this to prove security assuming compression.

Soundness analysis assuming compression. Fix some function ℓ. Consider $x \notin \mathcal{L}$ for which the proof of non-membership π_x satisfies $\boldsymbol{K}^t(\pi_x) \leq \ell(|x|)$. In this case, observe that the unsatisfiable circuit E_x constructed in Step (3) can be generated by a machine of size $\ell + O(1)$: one simply runs the machine for π_x and then constructs E_x from it. Therefore, the security proof can be carried out with an FHE ciphertext encrypting a dummy string of length only $\ell + O(1)$, rather than $O(L)$. In other words, when $\boldsymbol{K}^t(x, \pi_x) \leq \ell(|x|)$, security holds for a construction whose CRS grows with ℓ rather than L.

As shown in [HJLM26], the above soundness proof can indeed be formalized in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

Formalizing SNARG soundness as a falsifiable game. Consider an L-EF-friendly language $\mathcal{L}$. The soundness game is conducted over the language $\mathcal{U}$ of instances $x \notin \mathcal{L}$ which have an associated coNP witness of length $L(|x|)$ given by an EF proof of non-membership π_x. Clearly, $\mathcal{U} \in \mathsf{NP}$, and membership can be verified in polynomial time given π_x. This places us in the same efficiently falsifiable regime as EF-iO, and in contrast to the circuit iO case, allows us to invoke the polynomial hardness regime of the master theorem.

[21] We refer the reader to Definition 7.4 for details on local assignment generators, but we will not delve into these details here.

Upgrading security via the master theorem. One can therefore invoke Theorem 1.1 with polynomial security of the underlying PCG to establish soundness with compression parameter $\ell(m) = m^{o(1)}$, where $m = |x| + L(|x|)$. This gives full security of the scheme with a CRS of size $\mathsf{poly}(\lambda, L(|x|)^{o(1)}) = \mathsf{poly}(\lambda)$.

2.2.5 EF-Multi-Input Functional Encryption

In this section, we give an overview of our construction of succinct EF-MIFE for Theorem 1.7.

We begin by recalling the notion of MIFE. In a MIFE scheme supporting functions of arity n, the setup algorithm prepares a master secret key msk and n encryption keys $\mathsf{ek}_1, \ldots, \mathsf{ek}_n$ for each input index. The master secret key msk is used to generate decryption keys for n-arity functions, and the encryption keys ek_i are used to encrypt messages for the i-th input. Given ciphertexts of x_i under ek_i for each $i \in [n]$, the decryption key for function f computes $f(x_1, \ldots, x_n)$. The security of MIFE guarantees that an adversary cannot distinguish between ciphertext sets $\{\mathsf{ct}^0_{i,q} = \mathsf{Enc}(\mathsf{ek}_i, x^0_{i,q})\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}$ and $\{\mathsf{ct}^1_{i,q} = \mathsf{Enc}(\mathsf{ek}_i, x^1_{i,q})\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}$, given corrupted encryption keys $\{\mathsf{ek}_i\}_{i\in\mathcal{I}}$ together with decryption keys $\{\mathsf{sk}_{f_q}\}$, provided that the two sets are not trivially distinguishable. Formally, we require that for every queried function f_q and every $(q_1, \ldots, q_n)$,

$$f_q(x^0_{1,q_1}, \ldots, x^0_{n,q_n}) = f_q(x^1_{1,q_1}, \ldots, x^1_{n,q_n}),$$

where for corrupted indices $i \in \mathcal{I}$, $(x^0_{i,q_i}, x^1_{i,q_i})$ ranges over all pairs of identical messages (x, x). We refer to such a collection of input/function queries $\mathsf{ch} = \{(f_q, x^0_{i,q}, x^1_{i,q})\}$ as an *admissible* challenge. Notably, this admissibility condition is *not falsifiable* in polynomial time, as verifying it requires time Q^n.

An EF-MIFE scheme restricts the security guarantee to challenges equipped with an EF-*proof of admissibility*, namely, a proof certifying that the admissibility condition holds for the set of queried functions and inputs.

MIFE from [GGG+14]. We recall the bounded-query MIFE construction from [GGG+14] with the following overly-simplified illustration.

- **Setup.** Let n be the arity and Q be the query bound of the adversary. The scheme prepares
  - Two PKE public keys $\mathsf{pk}_0, \mathsf{pk}_1$ and a NIWI CRS crs.
  - A commitment Z_{ch} serving as a placeholder for committing the challenge queries, set to a commitment of 0 in honest execution.
  - n commitments $Z_{\mathsf{ek},i}$ committing bit 1, indicating that ek_i is initially treated as corrupted and can be given to the adversary. The corresponding openings $\sigma_{\mathsf{ek},i}$ of $Z_{\mathsf{ek},i}$ are included in ek_i.
- **Encryption.** To encrypt a message x under ek_i, the encryption algorithm generates ciphertexts $c_0 = \mathsf{Enc}(\mathsf{pk}_0, x)$ and $c_1 = \mathsf{Enc}(\mathsf{pk}_1, x)$, and then provides a NIWI proof π that [22]

  either c_0, c_1 encrypt identical messages and $Z_{\mathsf{ek},i}$ opens to 1,
  or c_0, c_1 encrypt the message pair $(x^0_{i,q}, x^1_{i,q})$ within the challenge ch committed in Z_{ch}.

[22] Technically, the NIWI language requires an additional branch which proves against commitments of *challenge ciphertexts* as opposed to *challenge messages*. This is an artifact of using NIWI instead of NIZK. As NIWI does not guarantee to hide the witness, semantic security of the PKE cannot be applied when proofs of ciphertext validity are given. We omit the details here for simplicity. We refer the readers to Section 6.2 for the full construction and security proof.

- **Decryption.** The decryption key for a function f contains the PKE keys, the NIWI crs, and the commitments from setups, together with an obfuscation $\widetilde{M_f} = \mathsf{iO}(M_{f,\mathsf{sk}_0})$, where M_{f,sk_0} is a program that hardcodes the PKE secret key sk_0, takes as input n ciphertexts of the form $(c_{i,0}, c_{i,1}, \pi_i)$, verifies the NIWI proofs, decrypts $x_i \leftarrow \mathsf{Dec}(\mathsf{sk}_0, c_{i,0})$, and outputs $f(x_1, \ldots, x_n)$ if all proofs are valid.

We now sketch the security proof of the construction. The goal of the proof is to show that the experiment in which the challenger encrypts $(x^0_{i,q}, x^0_{i,q})$ under the two PKE slots is indistinguishable from the one in which it encrypts $(x^1_{i,q}, x^1_{i,q})$.

First, program the challenge queries $\{(x^0_{i,q}, x^1_{i,q})\}$ into the commitment Z_{ch}. This allows the challenger to switch ciphertexts from encrypting identical messages $(x^0_{i,q}, x^0_{i,q})$ to encrypting the challenge queries $(x^0_{i,q}, x^1_{i,q})$, while maintaining validity of the NIWI proofs using the second branch[22]. Next, the commitments $Z_{\mathsf{ek},i}$ for non-corrupted indices are modified to commit to 0, ensuring that only challenge ciphertexts admit valid proofs. At this point, the admissibility condition guarantees that for any collection of ciphertexts with valid proofs, evaluating f on the decrypted inputs yields the same result whether decryption is performed using sk_0 or sk_1. By the security of iO, the obfuscation can therefore be switched to use sk_1. Finally, ciphertexts are switched to encrypt $(x^1_{i,q}, x^1_{i,q})$, completing the proof.

Although [GGG+14] focuses on circuits, the construction extends to Turing machines. In particular, for EF-MIFE, the obfuscation $\widetilde{M_f}$ can be instantiated using rate-1 EF-iO for Turing machines[23] from Section 2.2.1, yielding size $\mathsf{poly}(\lambda)$. Nevertheless, the decryption key size still depends on the challenge size, as the commitments Z_{ch} and Z_{ek_i} are included in the key.

Compressing the challenge. To obtain succinct decryption keys, we assume that the challenge queries $\mathsf{ch} = \{(f_q, x^0_{i,q}, x^1_{i,q})\}$ is compressible with large compression rate, i.e., $\boldsymbol{K}^t(\mathsf{ch}) \leq \ell(|\mathsf{ch}|) = \mathsf{poly}(\lambda)$, and we only commit the compressed challenge to Z_{ch}. This reduces the size of Z_{ch} from $O(Qn)$ to $\mathsf{poly}(\lambda)$. However, the NIWI *relation size* remains large, as it must express that Z_{ch} opens to a compressed string that decompresses to ch. To address this, we employ *relation-succinct* NIWI, whose proof size depends only on $\mathsf{poly}(\lambda, |x| + |w|)$ and is independent of the relation size. Such primitives can be obtained generically from NIWI and FHE (see Lemma 6.4).

A similar compression applies to the commitments $Z_{\mathsf{ek},i}$: the corrupted set can be derived from the compressed challenge, and thus the openings $\sigma_{\mathsf{ek},i}$ can be replaced by NIWI proofs certifying membership of i in this set. Again, relation succinctness ensures efficiency.

With all the commitments compressed to $\mathsf{poly}(\lambda)$ size, and all NIWI proofs and obfuscations equipped with succinctness guarantees, we obtain a succinct bounded-query EF-MIFE scheme assuming compressible challenges, which can then be upgraded using the master theorem.

We emphasize that, while ciphertext size becomes succinct, the *encryption time* still scales with the size of the challenge. The master theorem does not offer time compression, and therefore, this approach does not apply to unbounded-query MIFE. Extending our result to handle unbounded queries remains an interesting open question. See Section 2.3 for further discussion of limitations of our techniques.

2.2.6 Formalizing security proofs in bounded arithmetic

Recall that to apply the master theorem, the security (under compression) must be formalized in feasible cryptography, namely $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ + cryptographic security axioms. The formalization requires

[23] In the actual construction in Section 6.2, we will use description-succinct iO instead of rate-1 iO for a more direct security argument.

tedious but routine work. With the comprehensive toolkit developed in [HJLM26], most parts of the cryptographic arguments can be formalized in a straightforward manner. Specifically, we observe that:

- For our results on MIFE and iO, the security is formalized using the notion of indistinguishability [HJLM26, Section 4.2]. The security proof consists of a hybrid argument (formalizable using [HJLM26, Lemma 4.15]), and then the indistinguishability between consecutive hybrids. Moreover, the indistinguishability between consecutive hybrids are proved by one of the following two ways:
  (i) two hybrids are clearly identical, which can be resolved using a tool called isomorphic distributions [HJLM26, Lemma 4.14]);
  (ii) an application of a security axiom, which can be resolved using the reduction lemma in [HJLM26, Lemma 7.12]).
- For our results on SNARG, the security proof is a minor modification from the formalization of an EF-SNARG [JKLV24] in [HJLM26]. It suffices to verify that the modification does not significantly affect the proof complexity of the construction.

Throughout the formalization, the main challenge is to handle superpolynomial and even subexponential security in a feasible way. As only polynomial-time functions are allowed in PV proofs, we cannot formalize exponential-time algorithm or exponential size bounds as PV functions. This can be resolved by a careful "indirect" handling of superpolynomial functions (see Section 3.1.3) and appropriate encoding of security axioms (see Section 10).

2.3 Discussion: Non-Examples of our Master Theorem

While we have covered the technical content, we will now discuss a few examples where our master theorem does not help with overcoming the superfluous padding issue.

We first recap the common features across the above applications. Let (x, w) be a valid instance of the corresponding security game. In the constructions and analyses of the above primitives, we have that:

- the compression of $(x||w)$ is *not* required in the real scheme but only in the hybrids,
- when $(x||w)$ is compressible, one can show security of the construction while incurring a smaller padding.

With the above in mind, we now turn to settings where one or both of these conditions fail and our master theorem is not applicable.

Compressing cryptographic strings. One type of padding where our master theorem does not seem to apply is padding that arises due to *cryptographic* strings. The difficulty is that while most cryptographic strings do admit compressed representations (e.g., as the pre-image of a sampling algorithm), the presence of such a compression typically nullifies the security guarantee of the primitive in question.

As a concrete example, consider the padding that arises in iO-based constructions from the use of puncturable PRFs[24]. Via a simple modification of the GGM construction [GGM84], one obtains a PPRF with key size $\mathsf{poly}(\lambda)$ and punctured key size $\mathsf{poly}(\lambda, |\alpha^*|)$, so any iO-based construction using such a PPRF must pad the obfuscated program to size at least $\mathsf{poly}(\lambda, |\alpha^*|)$.

[24] A puncturable PRF [BW13, BGI14, SW14, KPTZ13] is a pseudorandom function whose key k can be "punctured" at a point α^* to obtain a punctured key $k\{\alpha^*\}$ that evaluates identically to the original on all inputs $\alpha \neq \alpha^*$, while revealing nothing about $F_k(\alpha^*)$.

One might hope to circumvent this by arguing security only when $k\{\alpha^*\}$ is compressible, then applying the master theorem to lift to full security. However, every punctured key $k\{\alpha^*\}$ is *already* compressible by definition: the pair (k, α^*) is a short description. But this compression immediately leaks $F_k(\alpha^*)$, directly contradicting punctured security. In other words, the master theorem requires that the compressibility of $(x\|w)$ be *compatible* with the security of the underlying primitives - a condition that could fail whenever the string being compressed is itself cryptographic.

A case of necessary padding in iO. As mentioned in Section 1.6, the work of Holmgren [Hol15] demonstrates an iO-based example where padding is necessary to prove security. The work defines two samplers Samp_0 outputting (aux_0, C_0) and Samp_1 outputting (aux_1, C_1) where the following hold:

- distributions $(\mathsf{aux}_0, \mathsf{iO}(1^\lambda, C_0))$ and $(\mathsf{aux}_1, \mathsf{iO}(1^\lambda, C_1))$ are *distinguishable*,
- but $(\mathsf{aux}_0, \mathsf{iO}(1^\lambda, C_0||0^s))$ and $(\mathsf{aux}_1, \mathsf{iO}(1^\lambda, C_1||0^s))$ are *indistinguishable*, where $s = \mathsf{poly}(\lambda)$.

Concretely, aux_b is a witness encryption of the bit b for the claim that "there exists a small circuit of size less than s with a specific truth table", and C_b is an obfuscated puncturable PRF whose truth table serves as a valid witness. In the unpadded setting, the distinguisher decrypts aux_b using C_b as a witness. In the padded setting, puncturable PRF security allows one to replace the truth table of C_b with a truly random string of size s, rendering the WE claim false with high probability. Now, one can invoke semantic security to argue indistinguishability.

The master theorem does not apply here because security fails even in the compressible setting. The reason is structural: the valid witness is the circuit C_b itself, and a compressible circuit can still compute the correct truth table and decrypt aux_b. Compressibility of (aux_b, C_b) is therefore orthogonal to the distinguishing attack, and the compressibility condition of the master theorem is violated regardless of the Kolmogorov complexity of the instance.

Remark 5. Note that our rate-1 iO construction does not contradict Holmgren's counterexample because rate-1 iO will still require one to pay for the size of the largest circuit appearing in any intermediate hybrid of the proof. Hence, even our rate-1 scheme will produce an obfuscation that grows with s when applied to Holmgren's padded circuits.

2.4 Organization of the Paper

The paper is organized into three parts.

- Part I consists of the introduction and technical overview (Section 2), followed by preliminaries (Section 3). We then formally define proof complexity generators against any PPS (i.e., the strong version, Assumption 2) and present our master theorem for upgrading security (Section 4) under this assumption.
- Part II presents our applications: rate-1 obfuscation for circuits and EF-equivalent Turing machines, along with succinct witness encryption (Section 5); succinct multi-input functional encryption (Section 6); and SNARGs with short CRS (Section 7). Throughout this part, we focus on cryptographic constructions and state our results under Assumption 2.
- Part III shows how to derive our results from the weaker PCG assumption against feasible cryptography (Assumption 1). We formalize proof complexity generators against bounded arithmetic, prove the master theorem in this setting, and carry out the formal security proofs required for our applications (Section 10).

The rationale behind separating the cryptographic constructions (in Part II) and their formalizations in bounded arithmetic (in Section 8) is to improve readability and modularity.

Additional discussion and cryptanalysis of proof complexity generators, as well as some technical details for APC_1 proofs, are deferred to the appendix.

3 Preliminaries

3.1 Bounded Arithmetic

For results built on proof complexity generators against provability, we assume that the readers have basic familiarity with bounded arithmetic, including Cook's theory PV and its first-order extension PV_1 [Coo75], and Jeřábek's theory APC_1 [Jeř07a]. Interested readers are referred to classical textbooks [Kra95, Kra19, CN10], the preliminary section of [HJLM26], as well as more recent materials [Oli25, Li25, Mat25].

3.1.1 Notation and Basic Concepts

We will use the following standard notation. Let $[n] := \{1, 2, \dots, n\}$, and $|x|$ be the length of a string x. We say $n \in \mathsf{Log}$ if n is the abbreviation of $|N|$ for another variable N. In particular, $\forall n \in \mathsf{Log}\ \phi(n)$ stands for $\forall N\ \phi(|N|)$, and $\exists n \in \mathsf{Log}\ \phi(n)$ stands for $\exists N\ \phi(|N|)$. We use $f : \mathsf{Log} \to \mathsf{Log}$ to denote a PV function where the input is unary encoded, where $f(n) \in \mathsf{Log}$ is defined only for $n \in \mathsf{Log}$; specifically, for $n = |N|$, $f(n)$ is the abbreviation of $|f(N)|$.

A *bounded quantifier* is of form $\forall x \le t$ or $\exists x \le t$, where t is a term. A sentence is said to be a $\forall\Sigma^b_i$-sentence if it is logical equivalent to a sentence with outermost unbounded universal quantifiers followed by i alternations of bounded quantifiers start with $\exists$: $\exists x_1 \le t_1\ \forall x_2 \le t_2\ \exists x_3 \le t_3 \dots$ It is said to be a $\forall\Sigma^b_\infty$-sentence if it is a $\forall\Sigma^b_i$-sentence for some $i \in \mathbb{N}$. Similarly, a sentence is said to be a $\forall\Pi^b_i$ formula if it is logically equivalent to a sentence with outermost unbounded universal quantifiers followed by i alternations of bounded quantifiers start with $\forall$: $\forall x_1 \le t_1\ \exists x_2 \le t_2\ \forall x_3 \le t_3 \dots$ Note that a $\forall\Pi^b_{i+1}$-sentence is also a $\forall\Sigma^b_i$-sentence if we write the outermost bounded $\forall$ as unbounded quantifiers.

A *sharply bounded quantifier* is a quantifier of form $\forall x \le |t|$ or $\exists x \le |t|$ for a term t. These quantifiers can be replaced PV terms that exhaustively enumerates all possible $x \le |t|$, and thus do not contribute to the quantifier complexity as $\forall\Sigma^b_i$ or $\forall\Pi^b_i$ sentences; see [Bus85, Oli25] for details.

Fix a language of first-order logic. Let Γ be a set of sentences and $\mathcal{T}$ be a theory in the same language. We use $\mathcal{T} \vdash \Gamma$ to denote that for every sentence $\phi \in \Gamma$, $\mathcal{T} \vdash \phi$.

3.1.2 Extended Frege and Propositional Translation

Extended Frege (EF) is a propositional proof system introduced by Cook and Reckhow [CR79]. Similar to Frege system (the "textbook" proof system), a proof is list of formulas (called *proof lines*), where each proof line is either an assumption, derived from an axiom scheme, or a deduction rule. In addition, Extended Frege allows the extension rule where for any formula ϕ, we can pick a fresh variable z and write a proof line $z \leftarrow \phi$, which allows efficient manipulation of circuits. We refer readers to standard textbooks [Kra95, Kra19] for more details.

Similar to the relationship between P and $\mathsf{P}_{/\mathsf{poly}}$, there is a tight connection between Cook's theory PV and EF proof system via the propositional translation.

- (*Translation of formulas*). Let $t_1(x) = t_2(x)$ be an equation in the language of PV. On a fixed input length n, we know by the Cook-Levin theorem that the equation can be encoded as a formula ϕ_n, in

the sense that $\forall x \in \{0,1\}^n\ t_1(x) = t_2(x)$ if and only if ϕ_n is a tautology. The formula ϕ_n is denoted as $[\![t_1(x) = t_2(x)]\!]_n$.

- (*Translation of proofs*). Suppose that $\mathsf{PV} \vdash e$, there is a polynomial $p(n)$ such that $[\![e]\!]_n$ admits a $p(n)$-sized EF proof. Moreover, given a PV proof π of e and an input length 1^n, there is a polynomial-time algorithm that generates the EF proof, denoted $[\![\pi]\!]_n$ [Coo75].

Furthermore, it can be proved that the correctness of the EF proof generator can be proved within PV. Specifically:

Theorem 3.1 (implicit in [Coo75]; see also [Li25])**.** *Suppose that* $\mathsf{PV} \vdash e$*, then there is an polynomial time algorithm* G *and a constant* k *such that*

$$\mathsf{PV}_1 \vdash \forall n \in \mathsf{Log}\ G(1^n) \text{ outputs an valid } \mathsf{EF} \text{ proof concluding } [\![e]\!]_n \text{ of size } kn^k.$$

Accordingly, the PV *equation encoding this* $\forall\Sigma_0^b$ *sentence is provable in* PV*.*

3.1.3 Formalizing Super-Polynomial Size Bounds

To formalize the notion of *superpolynomial security*, it is convenient to introduce a unified approach to define and manipulate superpolynomial size bounds.

Definition 3.2. A function $f(n) = n^{\Omega(1)}$, $f(n) \geq 1$ is said to be a PV-formalizable size bound if there is a PV function $f_{\leq}(\cdot,\cdot) : \mathsf{Log} \times \mathsf{Log} \to \mathsf{Log}$ such that the following conditions hold.

- (*Correctness*). $f(n) \leq m$ if and only if $f_{\leq}(n,m) = 1$ holds in the standard model.
- (*Provable Monotonicity*). PV_1 proves that for every $n, m_1, m_2 \in \mathsf{Log}$ satisfying that $m_1 \leq m_2$, $f_{\leq}(n,m_1) \leq f_{\leq}(n,m_2)$.

For $n, m \in \mathsf{Log}$, we write $f(n) \leq m$ as an abbreviation of the PV_1 formula $f_{\leq}(n,m) = 1$, and $f(n) \geq m$ as $f_{\leq}(n,m-1) = 0$. We write $m = f(n)$ as an abbreviation of $f(n) \leq m \wedge f(n) \geq m$.

It can be easily verified that simple size bounds such as $f_1(n) = cn^c$ or $f_2(n) = c2^{n^{1/c}}$, $c \in \mathbb{N}$, are PV-formalizable size bounds, as these functions are monotone and their inversions are polynomial-time computable.

Proposition 3.3. *Suppose that* $f(n)$ *is a* PV*-formalizable size bound,* $f(n) \cdot n^k$ *is also a* PV*-formalizable size bound for every integer* $k \geq 1$*.*

3.2 Time Bounded Kolmogorov Complexity

Fix a universal Turing machine $\mathsf{U_{TM}}(M, x, 1^t)$ that evaluates the Turing machine M on input x for t steps; it outputs $\perp$ if $M(x)$ does not halt within time t, and outputs $M(x)$ otherwise.

The time-bounded Kolmogorov complexity of a string $x \in \{0,1\}^n$, or $\boldsymbol{K}^t(x)$, is defined as the length of the shortest (M, z) such that $\mathsf{U_{TM}}(M, z, 1^{t(n)}) = x$. In this work, we will consider the special case where $t(n)$ is a fixed polynomial.

Note that the Kolmogorov complexity of strings depends on $\mathsf{U_{TM}}$. Nevertheless, all our results hold for any standard choice of $\mathsf{U_{TM}}$.

3.3 Cryptographic Definitions and Tools

3.3.1 Extractable commitment

An extractable commitment scheme is a tuple of algorithms $\mathsf{COM.Gen}, \mathsf{COM.Com}, \mathsf{COM.Ext}$ with the following syntax:

$\mathsf{Gen}(1^\lambda) \to (\mathsf{crs}, \mathsf{td})$**.** On input the security parameter λ, outputs a common reference string crs and a trapdoor td. In honest generation, we omit the output of td.

$\mathsf{Com}(\mathsf{crs}, b; \sigma) \to Z$**.** On input the common reference string crs, a bit b, and a commitment randomness σ, outputs a commitment Z. The randomness σ is the *opening* of the commitment.

$\mathsf{Ext}(\mathsf{td}, Z) \to b$**.** On input the trapdoor td and a commitment Z, outputs a bit b.

Definition 3.4 (Extractable commitment)**.** An extractable commitment scheme $\mathsf{COM}.(\mathsf{Gen}, \mathsf{Com}, \mathsf{Ext})$ is required to satisfy the following properties:

Extractability. For every $\lambda \in \mathbb{N}$, and every $(\mathsf{crs}, \mathsf{td})$ generated by $\mathsf{Gen}(1^\lambda)$, it holds that

$$\forall (Z, \sigma), \quad \mathsf{Com}(\mathsf{crs}, b; \sigma) = Z \implies \mathsf{Ext}(\mathsf{td}, Z) = b.$$

(Computational) Hiding. For every $\lambda \in \mathbb{N}$, every polynomial-time adversary $\mathcal{A}$, there exists a negligible function negl such that

$$|\Pr[\mathcal{A}(\mathsf{crs}, Z_0) = 1] - \Pr[\mathcal{A}(\mathsf{crs}, Z_1) = 1]| \leq \mathsf{negl}(\lambda),$$

where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Gen}(1^\lambda)$, and $Z_b \leftarrow \mathsf{Com}(\mathsf{crs}, b; \sigma)$ for random σ.

A commitment scheme for long message can be similarly defined and trivially constructed from single-bit commitments by committing to the message bit by bit.

It is well known that any perfectly correct public-key encryption scheme directly implies an extractable commitment scheme. It is shown in [Mat25] that the (leveled) GSW FHE scheme [GSW13], when instantiated with bounded noise, is a perfectly correct encryption scheme with PV proofs of correctness. This gives the following lemma.

Lemma 3.5 (Extractable commitment with PV proofs of extractability)**.** *Assuming LWE, there exists an extractable commitment scheme where the extractability admits a PV proof.*

3.3.2 Fully Homomorphic Encryption

A fully homomorphic encryption (FHE) scheme is a tuple of algorithms $\mathsf{FHE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ with the following syntax:

$\mathsf{KeyGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$. On input a security parameter 1^λ, outputs a public key pk and a secret key sk.

$\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$. On input a public key pk and a message m, outputs a ciphertext ct.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to m$. On input a secret key sk and a ciphertext ct, outputs a message m.

$\mathsf{Eval}(f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n) \to \mathsf{ct}_f$. On input a function f and ciphertexts $\mathsf{ct}_1, \ldots, \mathsf{ct}_n$, outputs a ciphertext ct_f.

Additionally, the ciphertext size is a fixed polynomial in the security parameter and the message length.

Definition 3.6 (FHE)**.** A fully homomorphic encryption scheme $\mathsf{FHE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ is required to satisfy the following properties:

Perfect Correctness. For every $\lambda \in \mathbb{N}$, every key pair $(\mathsf{pk}, \mathsf{sk})$ generated by $\mathsf{KeyGen}(1^\lambda)$, and every message m, it holds that

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m] = 1.$$

Malicious Evaluation Correctness. For every $\lambda \in \mathbb{N}$, every key pair $(\mathsf{pk}, \mathsf{sk})$ generated by $\mathsf{KeyGen}(1^\lambda)$, every function $f : \{0,1\}^n \to \{0,1\}$ and ciphertexts $\mathsf{ct}_1, \ldots, \mathsf{ct}_n$, it holds that

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)) = f(\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}_1), \ldots, \mathsf{Dec}(\mathsf{sk}, \mathsf{ct}_n))\right] = 1.$$

Semantic Security. For every $\lambda \in \mathbb{N}$, every polynomial-time adversary $\mathcal{A}$, and every polynomial message length $\ell = \ell(\lambda)$, there exists a negligible function negl such that for all messages $m_0, m_1 \in \{0,1\}^\ell$,

$$|\Pr\left[\mathcal{A}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, m_0)) = 1\right] - \Pr\left[\mathcal{A}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, m_1)) = 1\right]| \leq \mathsf{negl}(\lambda),$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.

[Mat25] shows that the (full-fledged) GSW FHE scheme [GSW13], when instantiated with bounded noise, is a perfectly correct with PV proofs of correctness. This gives the following lemma.

Lemma 3.7 (FHE with PV proofs of correctness)**.** *Assuming circular LWE, there exists a fully homomorphic encryption scheme where the perfect correctness and malicious evaluation correctness admits a PV proof.*

3.3.3 Somewhere Statistically-Binding Hash

A somewhere statistically binding (SEH) hash scheme is a tuple of algorithms $\mathsf{SEHash}.(\mathsf{Gen}, \mathsf{TGen}, \mathsf{Hash}, \mathsf{Open}, \mathsf{Ver}, \mathsf{Ext})$ with the following syntax:

$\mathsf{Gen}(1^\lambda, 1^N) \to \mathsf{hk}$. On input a security parameter 1^λ, message length 1^N, a locality parameter 1^{loc}, and outputs a hash key hk.

$\mathsf{TGen}(1^\lambda, 1^N, i \in [N]) \to (\mathsf{hk}^*, \mathsf{td})$**.** On input a security parameter 1^λ, message length 1^N, a locality parameter 1^{loc} and in index $i \in [N]$, outputs a hash key hk^* along with a trapdoor td.

$\mathsf{Hash}(\mathsf{hk}, \mathbf{x} \in \{0,1\}^N) \to \tau$**.** On input a hash key hk and a string $\mathbf{x}$, output a hash value τ.

$\mathsf{Open}(\mathsf{hk}, \mathbf{x}, i) \to \rho$. On input a hash key hk, a string $\mathbf{x} \in \{0,1\}^N$ and an index $i \in [N]$, output a "local opening" ρ.

$\mathsf{Ver}(\mathsf{hk}, \tau, i, y, \rho) \to 0/1$**.** On input a hash key hk, hash value τ, index $i \in [N]$, symbol $y \in \{0,1\}$ and opening ρ, the verification algorithm decides to accept or reject the local opening.

$\mathsf{Ext}(\mathsf{hk}^*, \mathsf{td}, \tau) \to \boldsymbol{y}$**.** On input the hash key hk^*, trapdoor td and hash value τ, the extraction algorithm outputs the bit $\mathbf{x}_i$ (the index which was chosen during the TGen algorithm).

Definition 3.8. A somewhere statistically binding (SEH) family $\mathsf{SEHash}.(\mathsf{Gen}, \mathsf{TGen}, \mathsf{Hash}, \mathsf{Open}, \mathsf{Ver}, \mathsf{Ext})$ is required to satisfy the following properties:

Succinct Key. The size of the key is bounded by $\mathsf{poly}(\lambda, |S|, \log N)$.

Succinct Hash. The size of the hash value c is bounded by $\mathsf{poly}(\lambda, |S|, \log N)$.

Succinct Local Opening. The size of the local opening $\pi_i \leftarrow \mathsf{Open}(K, m, i, r)$ is bounded by $\mathsf{poly}(\lambda, |S|, \log N)$.

Succinct Verification. The running time of the verification algorithm is bounded by $\mathsf{poly}(\lambda, |S|, \log N)$.

Key Indistinguishability. For any non-uniform PPT adversary $\mathcal{A}$ and any polynomial $N = N(\lambda)$, there exists a negligible function $\nu(\lambda)$ such that for all $i^* \in [N]$

$$\Bigg| \Pr\left[S \leftarrow \mathcal{A}(1^\lambda, 1^N), \mathsf{hk} \leftarrow \mathsf{Gen}(1^\lambda, 1^N) : \mathcal{A}(\mathsf{hk}) = 1\right] - \Pr\left[S \leftarrow \mathcal{A}(1^\lambda, 1^N), (\mathsf{hk}^*, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^N, i^*) : \mathcal{A}(\mathsf{hk}^*) = 1\right] \Bigg| \leq \nu(\lambda).$$

Opening Completeness. For any hash key hk, any message $\mathbf{x} = (x_1, \ldots, x_N) \in \{0,1\}^N$, any randomness r, and any index $i \in [N]$, we have

$$\Pr\left[\tau \leftarrow \mathsf{Hash}(\mathsf{hk}, \mathbf{x}), \pi_i \leftarrow \mathsf{Open}(\mathsf{hk}, \mathbf{x}, i) : \mathsf{Ver}(\mathsf{hk}, \tau, x_i, i, \rho_i) = 1\right] = 1.$$

Extraction Correctness. For any subset $i^* \in [N]$, any trapdoor key $(\mathsf{hk}^*, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^N, S)$, any hash τ, any bit x_{i^*}, and any proof π_{i^*}, we have

$$\Pr\left[\mathsf{Ver}(\mathsf{hk}, \tau, x_{i^*}, i^*, \rho_{i^*}) = 1 \Rightarrow \mathsf{Ext}(\tau, \mathsf{td})|_{i^*} = x_{i^*}\right] = 1.$$

Since the extracted value $\mathsf{Ext}(\tau, \mathsf{td})|_{i^*}$ is unique, the extraction correctness implies statistical binding property.

3.3.4 Non-Interactive Witness-Indistinguishable proof of knowledge

A non-interactive witness-indistinguishable proof of knowledge is a tuple of algorithms ($\mathsf{NIWI.Gen}$, $\mathsf{NIWI.Prove}$, $\mathsf{NIWI.Ver}$, $\mathsf{NIWI.Ext}$) with the following syntax.

$\mathsf{Gen}(1^\lambda, R) \to (\mathsf{crs}, \mathsf{td})$**.** On input the security parameter λ and a relation circuit R, outputs a common reference string crs and a trapdoor td. In honest generation, we omit the output of td.

$\mathsf{Prove}(\mathsf{crs}, x, w) \to \pi$**.** On input the common reference string crs, a statement x, and a witness w, outputs a proof π.

$\mathsf{Ver}(\mathsf{crs}, x, \pi) \to \{0,1\}$**.** On input the common reference string crs, a statement x, and a proof π, outputs 1 to accept the proof, or 0 to reject the proof.

$\mathsf{Ext}(\mathsf{td}, x, \pi) \to w$**.** On input the trapdoor td, a statement x, and a proof π, outputs a witness w.

Definition 3.9. A NIWIPoK supporting relation class $\mathcal{R} = \{\mathcal{R}_\lambda\}$ should satisfy the following properties:

Completeness. For every $\lambda \in \mathbb{N}$, every relation $R \in \mathcal{R}_\lambda$, every statement x and witness w such that $R(x, w) = 1$, and every crs generated by $\mathsf{Gen}(1^\lambda, R)$, it holds that

$$\Pr[\mathsf{Ver}(\mathsf{crs}, x, \mathsf{Prove}(\mathsf{crs}, x, w)) = 1] = 1,$$

Extractability. For every $\lambda \in \mathbb{N}$, every relation $R \in \mathcal{R}_\lambda$, and every $(\mathsf{crs}, \mathsf{td})$ generated by $\mathsf{Gen}(1^\lambda, R)$, it holds that

$$\forall x, \forall \pi, \quad \mathsf{Ver}(\mathsf{crs}, x, \pi) = 1 \implies R(x, \mathsf{Ext}(\mathsf{td}, x, \pi)) = 1.$$

Witness Indistinguishability. For every $\lambda \in \mathbb{N}$ and every polynomial time adversary $\mathcal{A}$, there exists a negligible function negl such that for every relation $R \in \mathcal{R}_\lambda$, statement x, and two witnesses w_0, w_1 with $R(x, w_0) = R(x, w_1) = 1$, it holds that

$$|\Pr[\mathcal{A}(\mathsf{crs}, x, \mathsf{Prove}(\mathsf{crs}, x, w_0)) = 1] - \Pr[\mathcal{A}(\mathsf{crs}, x, \mathsf{Prove}(\mathsf{crs}, x, w_1)) = 1]| \leq \mathsf{negl}(\lambda),$$

where $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, R)$.

Definition 3.10 (Relation Succinct NIWIPoK)**.** We say that a NIWIPoK scheme is relation succinct if the size of the proof π is bounded by $\mathsf{poly}(\lambda, |x| + |w|)$, i.e., the proof size grows only with the statement and witness size, but not the size of the relation circuit R.

3.3.5 Batch-NP Arguments

Syntax. A Batch Argument scheme (BARG) for NP consists of algorithms $\mathsf{BARG}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{TGen}, \mathsf{Ext})$ with the following syntax:

- $\mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L) \to \mathsf{crs}$: On input the security parameter λ, batch size m, the circuit size upperbound s, and the extraction size upperbound L, outputs a common reference string crs.
- $\mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S} \subseteq [m]) \to (\mathsf{crs}^*, \mathsf{td})$: On input the security parameter λ, batch size m, the circuit size upperbound s, the extraction size upperbound L, and a set of indices $\mathcal{S} \subseteq [m]$ with size at most L, outputs a common reference string crs^* and a trapdoor td.
- $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), (\mathbf{y}_1, \ldots, \mathbf{y}_m)) \to \pi$: On input the common reference string crs, a relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, m statements $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \{0,1\}^n$, and m witnesses $\mathbf{y}_1, \ldots, \mathbf{y}_m \in \{0,1\}^h$ such that $C(\mathbf{x}_j, \mathbf{y}_j) = 1$ for all $j \in [m]$, outputs a proof π.
- $\mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi) \to \{0,1\}$: On input the common reference string crs, a relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, m statements $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \{0,1\}^n$, and a proof π, outputs a bit indicating whether to accept or reject the proof.
- $\mathsf{Extract}(\mathsf{crs}^*, \mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi) \to \{\mathbf{y}_i^*\}_{i \in \mathcal{S}}$: On input the common reference string crs^*, the trapdoor td, a relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, m statements $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \{0,1\}^n$, and a proof π, outputs the extracted witness $\mathbf{y}_i^* = \mathbf{y}_i$ for each $i \in \mathcal{S}$ (the indices set which was chosen during the TGen algorithm).

We say that a BARG scheme supports *single point extraction* if the TGen and $\mathsf{Extract}$ algorithms only take a single index $i^* \in [m]$ as input/output instead of a set $\mathcal{S} \subseteq [m]$. In this case, we omit the 1^L input in Gen and TGen.

Definition 3.11 (Batch Argument)**.** A Batch Argument scheme $\mathsf{BARG}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{TGen}, \mathsf{Ext})$ is required to satisfy the following properties:

Succinct Proof. The size of the proof π is bounded by $\mathsf{poly}(\lambda, s, \log m, L)$.

Correctness. For all $\lambda, m, s, L \in \mathbb{N}$, all relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, and any instances $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$ and witnesses $(\mathbf{y}_1, \ldots, \mathbf{y}_m)$ such that $C(\mathbf{x}_i, \mathbf{y}_i) = 1$ for all $i \in [m]$, and all $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L)$, it holds that

$$\mathsf{Ver}\left(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \mathsf{Prove}\left(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), (\mathbf{y}_1, \ldots, \mathbf{y}_m)\right)\right) = 1.$$

CRS indistinguishability. For all polynomial sized adversaries $\mathcal{A}$, there exists a negligible function negl such that for all $\lambda \in \mathbb{N}$, $m, s, L \in \mathsf{poly}(\lambda)$, and all $\mathcal{S} \subseteq [m]$ where $|\mathcal{S}| \leq L(\lambda)$, it holds that

$$\left| \begin{array}{l} \Pr[\mathcal{A}(\mathsf{crs}) = 1 \mid \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L)] \\ -\Pr[\mathcal{A}(\mathsf{crs}) = 1 \mid (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S})] \end{array} \right| \leq \mathsf{negl}.$$

Somewhere Extractability. For all $\lambda, m, s, L \in \mathbb{N}$, all indices set $\mathcal{S} \subseteq [m]$, all relation circuit $C : \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, all instances $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$, all proof π, and all $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S})$, it holds that

$$\begin{aligned} &\forall i^* \in \mathcal{S}, \\ &\quad \mathsf{Ver}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi) = 1 \implies C(\mathbf{x}_{i^*}, \mathsf{Extract}_{i^*}(\mathsf{td}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_m), \pi)) = 1, \end{aligned}$$

where $\mathsf{Extract}_{i^*}(\cdot)$ denotes the extracted value on index i^*.

3.3.6 Witness Encryption

We recall the definition of witness encryption [GGSW13], where we follow the definition from [JJMP25] where full succinctness is considered.

Syntax. A witness encryption scheme consists of two PPT algorithms:

$\mathsf{Enc}(1^\lambda, R, x, n, b) \to \mathsf{ct}$**:** On input a security parameter λ, a machine R defining a relation, an instance x, a witness length bound n, and a message bit b, outputs a ciphertext ct.

$\mathsf{Dec}(1^\lambda, \mathsf{ct}, R, x, w) \to b$**:** On input a security parameter λ, a ciphertext ct, a machine R, an instance x and a witness w, outputs a bit b.

Definition 3.12. A witness encryption scheme $\mathsf{WE} = (\mathsf{Enc}, \mathsf{Dec})$ for a class of relations $\mathcal{R}$ should satisfy the following properties:

- **Correctness.** For every $R \in \mathcal{R}$, every $n \leq 2^\lambda$, every (x, w) such that $|w| \leq n$ and $R(x, w)$ outputs 1 in time at most 2^λ, and every $b \in \{0,1\}$,

$$\Pr[\mathsf{Dec}(1^\lambda, \mathsf{Enc}(1^\lambda, R, x, n, b), R, x, w) = b] = 1.$$

- **Efficiency.** For every $R \in \mathcal{R}$, every $n \leq 2^\lambda$, and every w such that $|w| \leq n$, the following holds.
  - Enc runs in time polynomial in λ, $|R|$, $|x|$, and n.
  - Dec runs in time polynomial in λ and the running time of $R(x, w)$.
  - **Witness succinctness.** We additionally say that WE is *witness-succinct* if the ciphertext size $|\mathsf{ct}|$ is $\mathsf{poly}(\lambda, |R|, |x|)$, independent of the witness size n.

- **Full succinctness.** We additionally say that WE is *fully succinct* if the ciphertext size $|\mathsf{ct}|$ is $\mathsf{poly}(\lambda, |R|)$, independent of both the instance size $|x|$ and the witness size n.

- **Security.** For every PPT adversary $\mathcal{A}$, and every polynomial bound s, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, all $R \in \mathcal{R}$ where $|R| \leq s(\lambda)$, all x where $|x| \leq s(\lambda)$ and $R(x, w)$ outputs 0 for all w, and all $n \leq s(\lambda)$,

$$\left|\Pr[\mathcal{A}(1^\lambda, \mathsf{Enc}(1^\lambda, R, x, n, 0)) = 1] - \Pr[\mathcal{A}(1^\lambda, \mathsf{Enc}(1^\lambda, R, x, n, 1)) = 1]\right| \leq \mathsf{negl}(\lambda).$$

Furthermore, we say that WE is an *extractable* witness encryption scheme for $\mathcal{R}$ if it satisfies the following additional property.

- **Extractability.** For every PPT adversary $\mathcal{A}$, and every polynomial bound s, and every polynomial $p(\cdot)$, there exists a PPT extractor $\mathcal{E}$, a polynomial $q(\cdot)$, and a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, all $R \in \mathcal{R}$ where $|R| \leq s(\lambda)$, all x where $|x| \leq s(\lambda)$, and all $n \leq s(\lambda)$,

$$\left|\Pr[\mathcal{A}(1^\lambda, \mathsf{Enc}(1^\lambda, R, x, n, 0)) = 1] - \Pr[\mathcal{A}(1^\lambda, \mathsf{Enc}(1^\lambda, R, x, n, 1)) = 1]\right| \geq \frac{1}{p(\lambda)}$$
$$\implies \Pr[R(x, \mathcal{E}(1^\lambda, R, x)) = 1] \geq \frac{1}{q(\lambda)} - \mathsf{negl}(\lambda).$$

Note that the witness length bound n can be omitted when the construction is succinct in n. We adopt the current notion for full generality.

3.3.7 Fully succinct obfuscation for TMs

The following section is taken nearly verbatim from [JJMP25].

We start by introducing the notion of iO for Turing machines with *split description*, following [JJMP25]. We consider programs that are represented by a public part pub and a (possibly) secret part M. Formally, we fix some universal machine $\mathcal{U}_{\mathsf{split}}$ that takes as input both parts of the program (M, pub), an input x to the program and simulates (M, pub) on x. We use $M[\mathsf{pub}](\cdot)$ to denote the machine $\mathcal{U}_{\mathsf{split}}(M, \mathsf{pub}, \cdot)$.

Remark 6 (Turing machines with explicit clocks)**.** For simplicity, we assume that in this work, programs given to iO are clocked; namely, the description of a machine M explicitly contains an efficiently computable polynomial p that describes its runtime.

In particular, we assume that there is a polynomial time algorithm $\mathsf{EQRT}(M_1, M_2)$ that outputs 1 if and only if M_1 and M_2 have identical clocks (subsequently, M_1 and M_2 have the same running time on every input). Moreover, we assume that the correctness of EQRT is provable in PV, i.e., PV proves that for clocked machines M_1, M_2, if $\mathsf{EQRT}(M_1, M_2) = 1$, then for every input x, and every $t \in \mathsf{Log}$, $M_1(x)$ halts within t steps if and only if $M_2(x)$ halts within t steps.

Definition 3.13 (iO for TMs for Programs with Split Description)**.** An indistinguishability obfuscation scheme iO for $\mathcal{U}_{\mathsf{split}}$ satisfies the following properties:

- **Correctness:** For every λ, $n \leq 2^\lambda$, machine $M[\mathsf{pub}]$, and input x such that $|x| \leq n$ and the running time of $M[\mathsf{pub}](x)$ is at most 2^λ, we have that

$$\Pr\left[\ \widetilde{M}[\mathsf{pub}](x) = M[\mathsf{pub}](x) \ \ :\ \ \widetilde{M} \leftarrow \mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n)\ \right] = 1\ .$$

- **Efficiency:** In the correctness experiment above:
  - The running time of iO is $\mathsf{poly}(\lambda, |M|, |\mathsf{pub}|)$.
  - There is a polynomial q such that on each input length x, the running time of $\widetilde{M}[\mathsf{pub}](x)$ is $O(q(t))$, where t is the running time of $M[\mathsf{pub}](x)$.
  - The size of the obfuscation $|\widetilde{M}| = \mathsf{poly}(\lambda, |M|, |\mathsf{pub}|, n)$.
    - *Input-succinctness:* We say that the scheme is *input-succinct* if additionally, $|\widetilde{M}| = \mathsf{poly}(\lambda, |M|, |\mathsf{pub}|)$.
    - *Description-succinctness:* We say that the scheme is *description-succinct* if $|\widetilde{M}| = \mathsf{poly}(\lambda, |M|, n)$.
    - *Full succinctness:* We say the scheme is *fully succinct* if it is both input-succinct and description-succinct, i.e. $|\widetilde{M}| = \mathsf{poly}(\lambda, |M|)$.
- **Security for split programs:** For every poly-size $\mathcal{A}$ and polynomial p, there exists a negligible function μ such that for every λ, M_0, M_1 and bounds $n \leq p(\lambda)$ such that
  - $|M_0| = |M_1| \leq p(\lambda)$, and $|\mathsf{pub}| \leq p(\lambda)$.
  - $\mathsf{EQRT}(M_0[\mathsf{pub}], M_1[\mathsf{pub}]) = 1$.
  - For every input x of length at most n, it holds that $M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)$.

  we have that

$$\Pr\left[\ \mathcal{A}(\widetilde{M}_b, \mathsf{pub}) = b \ \ :\ \ \begin{array}{c} b \leftarrow \{0,1\} \\ \widetilde{M}_b \leftarrow \mathsf{iO}(1^\lambda, M_b[\mathsf{pub}], 1^n) \end{array}\ \right] \leq \frac{1}{2} + \mu(\lambda)\ .$$

We will omit the input length n from the input to iO when the scheme is input or fully-succinct.

iO with PV proof of correctness. We now define what it means for an iO scheme to have a PV proof of correctness. For this, we first introduce the PV symbol U_{TM} which on input $M[\mathsf{pub}]$, x and 1^t, simulates $M[\mathsf{pub}](x)$ for t steps, and outputs $M[\mathsf{pub}](x)$ if the computation has halted, and $\perp$ otherwise. As discussed in [Li25, Section 2.1.1], this is in fact a valid function symbol.

Definition 3.14 (Indistinguishability obfuscation with PV proof of N-correctness)**.** We say that an indistinguishability obfuscation scheme iO has a PV proof of completeness of its correctness can be formalized in theory PV as function symbols:

$$\left(\begin{array}{c} \widetilde{M} = \mathsf{iO}(1^\lambda, M[\mathsf{pub}], n; r), \\ n \leq N(\lambda), \\ |x| \leq n \\ \mathsf{U}_{\mathsf{TM}}(M[\mathsf{pub}], x, 1^t) \neq \perp \end{array}\right) \rightarrow \left(\mathsf{U}_{\mathsf{TM}}(M[\mathsf{pub}], x, 1^t) = \mathsf{U}_{\mathsf{TM}}(\widetilde{M}[\mathsf{pub}], x, 1^{q(\lambda,t)})\right)$$

where q is some polynomial.

Fully succinct EF-iO for programs with split description. We now define a restriction of iO security to EF-equivalent programs, extending the definition in Section 3.3.7 to the case of programs with split description.

Definition 3.15 (EF-equivalence for programs with split description)**.** Let $\rho \in \mathbb{N}$ and τ be a polynomial. We say that programs $M_0[\mathsf{pub}](\cdot)$ and $M_1[\mathsf{pub}](\cdot)$ are (ρ, τ)-EF-equivalent if there exists a machine $\mathcal{M}_\Pi$ of size ρ which takes as input a length bound 1^n, along with M_0, M_1, pub and outputs an EF proof that $[\![M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)]\!]_n$ in $\tau(n)$ time.

In this work, we will introduce the following relaxed definition, which only asserts the correctness of the uniform EF proof generated by $\mathcal{M}_\Pi$ for inputs of bounded length. This makes it compatible with our master theorem (and the theory APC_1) that utilizes the proof complexity generator assumptions.

Definition 3.16 (EF-equivalence for programs with split description on bounded input length)**.** Let $\rho, \tau, m \in \mathbb{N}$. We say that programs $M_0[\mathsf{pub}](\cdot)$ and $M_1[\mathsf{pub}](\cdot)$ are (ρ, τ)-EF-equivalent for inputs of length at most m if there exists a machine $\mathcal{M}_\Pi$ of size ρ which takes as input a length bound 1^n where $n \leq m$, along with M_0, M_1, pub and outputs an EF proof that $[\![M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)]\!]_n$ in $\tau(n)$ time.

Definition 3.17 ((ρ, τ)-EF secure iO for programs with split description)**.** Let $\rho(\lambda)$ and $\tau(\lambda, m)$ be polynomials. We say that an iO scheme satisfies (ρ, τ)-EF-security if for every λ, and polynomial-sized adversary $\mathcal{A}$ and polynomial p, there exists a negligible function μ such that for every $M_0[\mathsf{pub}]$, $M_1[\mathsf{pub}]$, and n satisfying:

- $|M_0[\mathsf{pub}]| = |M_1[\mathsf{pub}]| \leq p(\lambda)$ and $n \leq p(\lambda)$,
- $\mathsf{EQRT}(M_0[\mathsf{pub}], M_1[\mathsf{pub}]) = 1$,
- $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$ are $(\rho(\lambda), \tau(\lambda, m))$-equivalent on inputs of length up to n (see Definition 3.16),

then

$$\Pr\left[\ \mathcal{A}(\widetilde{M}_b[\mathsf{pub}]) = b \ \ : \ \begin{array}{c} b \leftarrow \{0,1\} \\ \widetilde{M}_b[\mathsf{pub}] \leftarrow \mathsf{iO}(1^\lambda, M_b[\mathsf{pub}], 1^n) \end{array}\ \right] \leq \frac{1}{2} + \mu(\lambda) \ .$$

We say the scheme is Γ-secure if the advantage above is at most $(\Gamma(\lambda))^{-O(1)}$ for all adversaries of size $\mathsf{poly}(\Gamma)$.

Definition 3.18 (Fully succinct EF-iO)**.** An EF-iO for programs with split description is given algorithms $\mathsf{iO}_{\rho,\tau}$ for every pair of polynomials $\rho(\lambda)$ and $\tau(\lambda, n)$, $\mathsf{iO}_{\rho,\tau}$ is input-succinct iO scheme with (ρ, τ)-security, satisfying the following efficiency: In the correctness experiment described in Definition 3.13:

- The running time of $\mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$ is $\mathsf{poly}(\lambda, |M|, |\mathsf{pub}|, \rho(\lambda))$. We say the scheme is additionally *fully succinct* if the size of the obfuscation is $\mathsf{poly}(\lambda, |M|, \rho(\lambda))$.
- The exists a polynomial q such that $\mathsf{RT}(\widetilde{M}[\mathsf{pub}], x) = q(\lambda, \mathsf{RT}(M[\mathsf{pub}], x), \rho(\lambda), \tau(\lambda, |x|))$.

Theorem 3.19 ([JJMP25])**.** *Assuming*

- *subexponential hardness of Learning with Errors,*
- *subexponentially secure one-way functions, and*
- *subexponentially secure indistinguishability obfuscation for circuits,*

there exists a subexponentially secure fully succinct EF*-iO scheme. Moreover, the scheme has a* PV *proof of correctness.*

Description-succinct iO. Given subexponentially secure EF-iO, [JJMP25] show how to additionally obtain description-succinct iO for all machines. Recall that description-succinct iO is an iO scheme where the size of an obfuscation of $M[\mathsf{pub}]$ for inputs of length n is $\mathsf{poly}(\lambda, |M|, n)$.

Theorem 3.20 ([JJMP25]). *Assuming subexponentially secure fully-succinct* EF*-iO, there exists a description-succinct iO scheme for programs with split description.*

3.3.8 Succinct Non-Interactive Arguments

The following is taken nearly verbatim from [HJLM26].

Syntax. A succinct non-interactive argument (SNARG) system for an NP relation $\mathcal{M}$ consists of polynomial-time algorithms $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ with the following syntax:

- The randomized setup algorithm Gen takes as input a security parameter $\lambda \in \mathbb{N}$ and an input length n, both in unary, and outputs a common reference string crs.
- The prover algorithm $\mathcal{P}$ takes as input the common reference string crs, the relation $\mathcal{R}$, an input $x \in \{0,1\}^n$ and its associated witness $w \in \{0,1\}^m$, and outputs a proof π.
- The verifier algorithm $\mathcal{V}$ takes as input the common reference string crs, the relation $\mathcal{R}$, an input $x \in \{0,1\}^n$ and a proof π. It outputs a bit indicating if it accepts or rejects.

Definition 3.21 (SNARG). A triple of algorithms $(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$ is a SNARG system for a Turing machine $\mathcal{M}$ if the following hold:

Completeness. For every $\lambda \in \mathbb{N}, n, m \leq 2^\lambda$, polynomial time relation machine $\mathcal{R}$, and every $x \in \{0,1\}^n$ and $w \in \{0,1\}^m$ such that $\mathcal{M}(x,w) = 1$,

$$\Pr\left[\mathcal{V}(\mathsf{crs}, \mathcal{R}, x, \pi) = 1 \ : \ \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^m, \mathcal{R}), \\ \pi \leftarrow \mathcal{P}(\mathsf{crs}, \mathcal{R}, x, w) \end{array} \right] = 1.$$

We say that the scheme is N-complete for a function $N(\lambda) \leq 2^\lambda$ if the above only holds for $n \leq N(\lambda)$.

Efficiency. In the completeness experiment above, the length of crs is $\mathsf{poly}(\lambda, n)$. We say the CRS is *fully succinct* if $|\mathsf{crs}| = \mathsf{poly}(\lambda)$. The length of a proof π is $\mathsf{poly}(\lambda)$. The runtime of $\mathcal{V}$ is $\mathsf{poly}(|\mathsf{crs}|, |\pi|, n, m)$[25].

Non-adaptive soundness. For every $x \in \{0,1\}^n$ where $x \notin \mathcal{L}_\mathcal{M}$ and every poly-size adversary Adv, there is a negligible function μ such that

$$\Pr\left[\mathcal{V}(\mathsf{crs}, \mathcal{R}, x, \pi^*) = 1 \ : \ \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^m, \mathcal{R}), \\ \pi^* \leftarrow \mathsf{Adv}(\mathsf{crs}) \end{array} \right] \leq \mu(\lambda).$$

We say the scheme is Γ-secure if the advantage above is at most $(\Gamma(\lambda))^{-O(1)}$ for all adversaries of size $\mathsf{poly}(\Gamma)$.

[25] One can generically improve the runtime of the verifier to $\mathsf{poly}(|\pi|, \lambda, \log n, \log m) \cdot n$ by applying a RAM delegation protocol or SNARG for P. See for example, [WW24, Remark 2.7]. Therefore, we will not focus on verifier efficiency in this work.

Remark 7. Recall that given a SNARG, one can generically improve the runtime of the verifier to $\mathsf{poly}(|\pi|, \lambda, \log n, \log m) \cdot n$ by applying a RAM delegation protocol or SNARG for P. In more detail, one can compute a hash of the crs, $\mathsf{crs}_{\mathcal{V}}$, and the new verifier $\mathcal{V}'(\mathsf{crs}_{\mathcal{V}}, x, \pi)$ can then use a RAM delegation protocol to verify that $\mathcal{V}(\mathsf{crs}, \mathcal{R}, x, \pi)$ would have indeed accepted in quasilinear time (see for example [WW24, Remark 2.7] for more details). Note that we do not have to prove that this transformation is secure in our theory since under our assumption, the resulting SNARG for NP will be sound in the *plain* model. Therefore, for simplicity, we will focus on constructing a SNARG with this weaker verifier efficiency.

3.3.9 Multi-Input Functional Encryption

In this section, we recall the definition of multi-input functional encryption (MIFE) adapted from [GGG+14], along with the multi-input functional encryption with EF proof of admissibility (EFMIFE) adapted from [MDS25]. For the purpose of this work, we will focus on the case of bounded-secure MIFE and EFMIFE, where the security only holds for a bounded number of ciphertexts and key queries.

A *bounded-query multi-input functional encryption* scheme for function family $\mathcal{F}$ consists of the following algorithms:

$\mathsf{Setup}(1^\lambda, 1^n, 1^L, 1^Q) \to (\mathsf{msk}, \mathsf{ek}_1, \ldots, \mathsf{ek}_n)$. On input the security parameter λ, the arity $n \le 2^\lambda$, the message length L, and the maximum number of queries Q, outputs a master secret key msk and n encryption keys $\mathsf{ek}_1, \ldots, \mathsf{ek}_n$.

$\mathsf{KeyGen}(\mathsf{msk}, f) \to \mathsf{sk}_f$. On input the master secret key msk and a t-ary function $f : (\{0,1\}^L)^n \to \{0,1\}^m \in \mathcal{F}$ of arity n, outputs a functional decryption key sk_f.

$\mathsf{Enc}(\mathsf{ek}_i, x_i) \to \mathsf{ct}_i$. On input an encryption key ek_i and a message $x_i \in \{0,1\}^L$, outputs a ciphertext ct_i.

$\mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n) \to y$. On input a functional decryption key sk_f and ciphertexts $\mathsf{ct}_1, \ldots, \mathsf{ct}_n$, outputs a message $y \in \{0,1\}^m$.

Definition 3.22 (Bounded-query MIFE)**.** A bounded-query multi-input functional encryption scheme for function family $\mathcal{F}$

$$\mathsf{MIFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$$

is required to satisfy the following properties:

Correctness. For every security parameter $\lambda \in \mathbb{N}$, every $n, L, Q \in \mathbb{N}$, there exist an negligible function negl such that for every function $f : (\{0,1\}^L)^n \to \{0,1\}^m \in \mathcal{F}$ of arity n, and every messages $x_1, \ldots, x_n \in \{0,1\}^L$,

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}_f, \mathsf{Enc}(\mathsf{ek}_1, x_1), \ldots, \mathsf{Enc}(\mathsf{ek}_n, x_n)) \neq f(x_1, \ldots, x_n)\right] = \mathsf{negl}(\lambda),$$

where $(\mathsf{msk}, \mathsf{ek}_1, \ldots, \mathsf{ek}_n) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, 1^L, 1^q)$, and $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$.

Very Selective IND-security. [26] We define a challenge with parameter (n, L, Q) as a set $\mathsf{ch} = \{\{f_q \in \mathcal{F}\}_{q\in[Q]}, \mathcal{I} \subseteq [n], \{\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)} \in \{0,1\}^L\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}\}$, consist of Q function queries $\{f_q\}_{q\in[Q]}$, a corrupted index set $\mathcal{I} \subseteq [n]$, and Q pairs of challenge messages $\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)}$ for each uncorrupted index

[26] Comparison to selective security [GGG+14]: both selective and very-selective security requires the encryption queries $\{\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)}\}$ to be determined before Setup. Very-selective security additionally requires that the key queries f_q to also be determined before Setup.

$i \in [n] \setminus \mathcal{I}$. We say that the challenge is *admissible* if

$$\forall k \in [Q], \forall \{q_i \in [Q]\}_{i \in [n]\setminus\mathcal{I}}, \forall \{\hat{\mathbf{x}}_i \in \{0,1\}^L\}_{i\in\mathcal{I}}, \quad f_k(\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_n^{(0)}) = f_k(\mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_n^{(1)}),$$

where $\mathbf{x}_i^{(b)} = \hat{\mathbf{x}}_i$ if $i \in \mathcal{I}$, and $\mathbf{x}_i^{(b)} = \mathbf{x}_{i,q_i}^{(b)}$ otherwise.

For any adversary $\mathcal{A}$, we define the following IND-security game $\mathsf{IND}_{\mathcal{A}}()$ as follows:

Experiment $\mathsf{IND}_{\mathcal{A}}(1^\lambda)$:

1. The adversary $\mathcal{A}(1^\lambda)$ provides parameters $(1^n, 1^L, 1^Q)$ and a challenge $\mathsf{ch} = \{\{f_q\}_{q\in[Q]}, \mathcal{I}, \{\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)}\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}\}$. If ch is not admissible, the experiment halts and outputs 0.
2. The challenger samples a bit $b \leftarrow \{0,1\}$ and encryption keys

   $$(\mathsf{msk}, \mathsf{ek}_1, \ldots, \mathsf{ek}_n) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, 1^L, 1^Q).$$

   It then computes decryption keys

   $$\mathsf{sk}_{f_q} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f_q) \text{ for all } q \in [Q]$$

   and ciphertexts

   $$\mathsf{ct}_{i,q} \leftarrow \mathsf{Enc}(\mathsf{ek}_i, \mathbf{x}_{i,q}^{(b)}) \text{ for all } i \in [n] \setminus \mathcal{I}, q \in [Q].$$

3. The adversary $\mathcal{A}$ is given $(\{\mathsf{ek}_i\}_{i\in\mathcal{I}}, \{\mathsf{sk}_{f_q}\}_{q\in[Q]}, \{\mathsf{ct}_{i,q}\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]})$, and outputs a guess b'.
4. The experiment outputs 1 if $b' = b$, and 0 otherwise.

We say that the bounded-query MIFE scheme MIFE is *very selectively* IND*-secure* if for every polynomial-time adversary $\mathcal{A}$, every polynomial sized parameters $n = n(\lambda), L = L(\lambda), Q = Q(\lambda)$, there exists a negligible function negl such that

$$\Pr\left[\mathsf{IND}_{\mathcal{A}}(1^\lambda) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Definition 3.23 (Bounded-query EF-MIFE)**.** A bounded-query multi-input functional encryption scheme with EF proof of admissibility EFMIFE is defined as in Definition 3.22, with the following modification.

Parameters for proof of admissibility. The Setup algorithm takes an additional parameter 1^p denoting the EF-proof size bound.

p-provable admissibility. We say a challenge ch is *p-provably admissible* if there exists an EF-proof π of size at most p for the admissibility sentence of ch:

$$\forall k \in [Q], \forall \{q_i \in [Q]\}_{i \in [n]\setminus\mathcal{I}}, \forall \{\hat{\mathbf{x}}_i \in \{0,1\}^L\}_{i\in\mathcal{I}}, \quad f_k(\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_n^{(0)}) = f_k(\mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_n^{(1)}),$$

where $\mathbf{x}_i^{(b)} = \hat{\mathbf{x}}_i$ if $i \in \mathcal{I}$, and $\mathbf{x}_i^{(b)} = \mathbf{x}_{i,q_i}^{(b)}$ otherwise.

The indistinguishability experiment in Definition 3.22 is modified to take the extra parameter p such that the $\mathcal{A}(1^\lambda)$ additionally outputs the parameter 1^p, and the experiment outputs 0 if the challenge ch is not p-provably admissible.

As a remark, an EF-MIFE scheme for function class $\mathcal{F}$ can be seen as a MIFE scheme for function class $\mathcal{F}_{\mathsf{EF},p}$ consisting of all function f satisfying that every admissible challenge with function query set $\{f\}$ has an EF-proof of size at most p.

4 Upgrading Security via Proof Complexity Generators

In this section, we will define proof complexity generators, and present our main master theorem on how to upgrade security of a construction using proof complexity generators.

4.1 Proof Complexity Generator against NP

Definition 4.1 (Proof Complexity Generator, plain)**.** Let $\ell(m) = o(m)$ and $t(m)$ be functions. A polynomial-time uniform family $\{C_m : \{0,1\}^{\ell(m)} \to \{0,1\}^m\}_{m\in\mathbb{N}}$ of circuits is said to be a t-secure proof complexity generator with seed length ℓ if for any deterministic polynomial-time algorithm $\mathcal{A}(1^m, y, \tau)$ and any $t'(m) := t(m) \cdot \mathsf{poly}(m)$, the following conditions cannot hold simultaneously:

- (*Completeness*): For infinitely many $m \in \mathbb{N}$, there exists a string $y \in \{0,1\}^m \setminus \mathsf{Range}(C_m)$ and a string $\tau \in \{0,1\}^{t'(m)}$ such that $\mathcal{A}(1^m, y, \tau) = 1$.
- (*Soundness*): For every sufficiently large $m \in \mathbb{N}$, every string $y \in \mathsf{Range}(C_m)$ and every $\tau \in \{0,1\}^{t'}$, $\mathcal{A}(1^m, y, \tau) = 0$.

A PCG is said to be *polynomially secure* if $t(m) = 1$, is said to be *slightly superpolynomially secure* if $t(m) = m^{\omega(1)}$, and is said to be *subexponentially secure* if $t(m) = 2^{m^\delta}$ for some constant $\delta > 0$.

In contrast to most of the cryptographic assumptions, the assumption is trivially *false* against a non-uniform adversary as it can simply hard-code a sequence of strings $y_1 \in \{0,1\}, \ldots, y_m \in \{0,1\}^m, \ldots$ Therefore, in the security reduction, we need to make sure that the adversary $\mathcal{A}$ is a uniform algorithm.

We consider two typical parameter regimes that we will use. The first setting is to consider polynomially secure PCGs with sub-polynomial seed length.

Assumption 3 (Polynomial PCG with Large Stretch)**.** There is a polynomially secure PCG with seed length $\ell(m) := m^{o(1)}$.

Alternatively, we can also consider slightly superpolynomial and subexponentially secure PCGs with sub-linear seed length.

Assumption 4 (Slightly Superpolynomial PCG with Large Stretch)**.** There is a s-secure PCG with seed length $\ell(m) := m^{o(1)}$ for some function $s(m) = m^{\omega(1)}$.

Assumption 5 (Subexponential PCG with Nontrivial Stretch)**.** There is a subexponentially secure PCG with seed length $\ell(m) = o(m)$.

4.1.1 An Equivalent Assumption: Proof Complexity Generators against any PPS

We propose an equivalent formulation of the same assumption, based on the notion of proof complexity generators against Cook-Reckhow propositional proof systems (see, e.g., [Kra24] for detailed exposition). We start with several definitions.

Definition 4.2 (Cook-Reckhow PPS [CR79])**.** Let $\mathsf{TAUT} \subseteq \{0,1\}^*$ be the set of propositional tautologies. A (Cook-Reckhow) propositional proof system (PPS) is a polynomial-time computable surjective function $V : \{0,1\}^* \to \mathsf{TAUT}$. For any tautology $\phi \in \mathsf{TAUT}$, a string π is said to be a V-proof of ϕ if $V(\pi) = \phi$.

Definition 4.3 (τ-formula)**.** Let $C_m : \{0,1\}^\ell \to \{0,1\}^m$ be a polynomial-time computable mapping. For any string $y \in \{0,1\}^m$, $\tau_y(C_m)$ is defined as the propositional formula with ℓ variables $x = (x_1, \ldots, x_\ell)$ formalizing that $C_m(x) \neq y$; in other words, $\tau_y(C_m)$ is a tautology if and only if $y \notin \mathsf{Range}(C_m)$.

Definition 4.4 (Proof Complexity Generator against a PPS; see, e.g., [Kra24])**.** Let V be a PPS and $C_m : \{0,1\}^\ell \to \{0,1\}^m$ be a polynomial-time computable mapping. Then C_m is said to be a t-secure proof complexity generator against V, if and only if for every constant $k \in \mathbb{N}$, length m, and $y \in \{0,1\}^m$, $\tau_y(C_m)$ requires a V-proof of size at least $t(m) \cdot \Omega(m^k)$.

Proposition 4.5. *Assumption 3 holds if and only if there exists a proof complexity generator against every propositional proof system.*

Proof Sketch. ($\Rightarrow$): If Assumption 3 holds, we prove that C_m is indeed a proof complexity generator against any PPS. Towards a contradiction, assume that a PPS V breaks C_m as a candidate PCG. We define the following algorithm $\mathcal{A}$ that violates Assumption 3: $\mathcal{A}(1^m, y, \pi) = 1$ if and only if π is a V-proof of $\tau_y(C_m)$. The completeness follows from the fact that V breaks C_m as PCG, while the soundness follows from that V is a Cook-Reckhow PPS.

($\Leftarrow$): Assume that there exists a proof complexity generator C_m against any PPS, we prove that it indeed satisfies Assumption 3. Suppose not, there is an algorithm $\mathcal{A}(1^m, y, \pi)$ that is both complete and sound. Consider the following PPS V that is given a proof z as input:

- If z can be parsed as $(0, y, \pi)$ and $\mathcal{A}(1^m, y, \pi) = 1$, it outputs $\tau_y(C_m)$.
- If z can be parsed as $(1, \pi)$, where π is an Extended Frege proof concluding ϕ, it outputs ϕ.
- Otherwise, it outputs $\top$.

Note that the second bullet ensures that every tautology admits a proof, while the first bullet as well as the soundness of $\mathcal{A}$ ensures that the output of V is always a tautology. Subsequently, V is a valid PPS. However, one can verify that the completeness of $\mathcal{A}$ implies C_m is not a PCG against V, leading to a contradiction. □

Remark 8. Note that Ilango [Ila25b] constructed a non-uniform proof complexity generator against any PPS; as mentioned in Section 1.3, it is indeed a (uniform) proof complexity generator against any PPS in CRS model. This result is insufficient to prove Assumption 3 as we require a uniform propositional proof system with no CRS.

4.1.2 Another Equivalent Assumption: Time-bounded Chaitin Incompleteness

Let $t = t(m)$ be a time-constructible function. Let $\mathsf{U}_{\mathsf{TM}}(M, x, 1^t)$ be a fixed universal Turing machine that parses M as a Turing machine, simulates $M(x)$ for t steps, and outputs the string on the tape. Recall that the *time-bounded Kolmogorov complexity* $\boldsymbol{K}^t(y)$ (with respect to U_{TM}) is defined as the length of the shortest description of a pair (M, x) such that $\mathsf{U}_{\mathsf{TM}}(M, x, 1^{t(|y|)}) = y$.

Definition 4.6. Let $\ell(m) = o(m)$ be a function (called the *compression rate*), $t(m), t'(m)$ be functions. A deterministic t'-time algorithm $\mathcal{A}(1^m, y, \tau)$ is said to witness t-time ℓ-incompressible strings if the following two conditions hold simultaneously:

- (*Completeness*): For infinitely many $m \in \mathbb{N}$, there exists a string $y \in \{0,1\}^m$ and a string $\tau \in \{0,1\}^{t'(m)}$ such that $\boldsymbol{K}^t(y) > \ell(m)$ and $\mathcal{A}(1^m, y, \tau) = 1$.
- (*Soundness*): For every sufficiently large $m \in \mathbb{N}$ and every $y \in \{0,1\}^m$ such that $\boldsymbol{K}^t(y) \leq \ell(m)$ and $\tau \in \{0,1\}^{t'(m)}$, $\mathcal{A}(1^m, y, \tau) = 0$.

Assumption 6 (Time-bounded Chaitin Incompleteness)**.** There is a polynomial $t = t(m)$ such that there is no $\mathsf{poly}(m)$ time algorithm that witnesses t-time $m^{o(1)}$-incompressible strings.

One can observe that Assumption 6 is equivalent to say that for a polynomial $t = t(m)$, the circuit $C_m : \{0,1\}^{\ell(m)} \to \{0,1\}^m$ defined by $(M, x) \mapsto \mathsf{U}_{\mathsf{TM}}(M, x, 1^{t(m)})$ is a secure proof complexity generator against MA. It follows from a simple modification of [RSW22, Theorem 6.7] that Assumption 3 and Assumption 6 are equivalent for $\ell(m) = m^{o(1)}$. Namely, $(M, x) \mapsto \mathsf{U}_{\mathsf{TM}}(M, x, 1^{t(m)})$ is the "hardest" proof complexity generator.

Theorem 4.7 (Implicit in [RSW22, Theorem 6.7])**.** *Let $k(m) \leq o(m)$ and $t'(m)$ be functions. Then the following two statements are equivalent:*

(1). *There exists a t'-secure PCG with seed length $O(k(m))$.*

(2). *There is a polynomial $t(m)$ such that there is no $t'(m) \cdot \mathsf{poly}(m)$ time algorithm that witnesses t-time $O(k(m))$-incompressible strings.*

Proof. (1) $\Rightarrow$ (2). Let $\{C_m\}_{m\in\mathbb{N}}$ be a proof complexity generator with seed length $\ell(m) = O(k(m))$. As C_m is a uniform family of circuits, there is a polynomial-time algorithm $P(1^m)$ that outputs C_m. Let $c \in \mathbb{N}$ be a constant such that $P(1^m)$ runs in time $O(m^c)$ and outputs C_m of size at most $O(m^c)$.

Towards a contradiction, assume that (2) is false for $t(m) = m^{c+1}$ and compression rate $\ell'(m) = \ell(m) + c$, where $c \in \mathbb{N}$ is a constant to be determined later. Fix any input length m that is sufficiently large. Let $C_m : \{0,1\}^{\ell(m)} \to \{0,1\}^m$ be the circuit generated by $P(1^m)$, and $M(x)$ be the Turing machine that generates the circuit C_m and outputs $C_m(x)$. Notice that for every $y = C_m(x) \in \mathsf{Range}(C_m)$, $M(x)$ outputs y in time $t(m) = m^{c+1}$ (when m is sufficiently large). Therefore, $\boldsymbol{K}^t(y) \leq |(M, x)| \leq \ell(m) + c$ if we set $c \in \mathbb{N}$ to be sufficiently large.

Let $\mathcal{A}(1^m, y, \tau)$ be a deterministic $t'(m)\cdot\mathsf{poly}(m)$ time algorithm that witnesses t-time ℓ'-incompressible strings. We show that on each such input length m that is sufficiently large, $\mathcal{A}$ is also a sound and complete adversary for the proof complexity generator P. To see this, notice that:

- (*Soundness*). For every $y \in \mathsf{Range}(C_m)$, we have $\boldsymbol{K}^t(y) \leq \ell(m) + c$, and thus the algorithm $\mathcal{A}(1^m, y, \tau)$ rejects any witness τ of length $t'(m) \cdot \mathsf{poly}(m)$.
- (*Completeness*). There is $y \in \{0,1\}^m$ and τ of length $t'(m)\cdot\mathsf{poly}(m)$ such that $\boldsymbol{K}^t(y) > \ell(m)+c$ and $\mathcal{A}(1^m, y, \tau)$ accepts. By the discussion above, $y \notin \mathsf{Range}(C_m)$ and thus the completeness property holds when we view $\mathcal{A}$ as an adversary to the PCG $\{C_m\}$.

This leads to a contradiction to (1).

(2) $\Rightarrow$ (1). Suppose that (2) holds for the compression rate $\ell(m) = O(k(m))$ and time bound $t = t(m)$. Consider the proof complexity generator $C_m : \{0,1\}^{\ell(m)} \to \{0,1\}^m$: C_m parses the input as a pair (M, x) and outputs $\mathsf{U}_{\mathsf{TM}}(M, x, 1^{t(m)})$. Note that $P(1^m)$ runs in $(t(m))^{O(1)} = m^{O(1)}$ time. By the definition, we can see that $y \in \mathsf{Range}(C_m)$ *if and only if* $\boldsymbol{K}^t(y) \leq \ell(m)$, which immediately implies that it is a proof complexity generator. □

4.2 Falsifiable Security Games

We consider the following general model for *falsifiable security games* of cryptographic primitives against non-uniform adversaries.

Let $T(n) \geq n$ be a time bound. Let $\mathcal{C}(1^T, x, w, A; r)$ be a probabilistic polynomial-time algorithm called the *challenger*, and $\mathcal{U} \in \mathsf{NTIME}[T]$ be a language. Let $n, s \in \mathbb{N}$ and $T = T(n)$. The security game with input length n and adversary circuit size s, denoted by $\boldsymbol{G}_{n,s}$, is defined as follows.

- The adversary picks a string $x \in \mathcal{U}$ of length n, a witness w for $x \in \mathcal{U}$ of length at most T, and a circuit A of size at most s.
- The challenger samples a random seed r for the algorithm $\mathcal{C}$.
- The adversary wins if and only if $\mathcal{C}(1^T, x, w, A; r) = 1$.

This game is said to be T-time falsifiable.

We introduce an additional parameter $\gamma \in (0, 1)$ to accommodate different types of security games; we say that security game is secure if the winning probability of the adversary is at most $\gamma + \mathsf{negl}(\lambda)$ for a negligible function $\mathsf{negl}(\cdot)$. More generally, it is said to be ε-secure if the winning probability is at most $\gamma + \varepsilon$. Formally:

Definition 4.8 (Security of Games)**.** Let $\gamma \in (0, 1)$ be a constant, $\varepsilon = \varepsilon(\lambda) \in (0, 1)$, $s = s(\lambda)$, $n = n(\lambda)$ be functions. The game $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is said to be (s, ε)-*secure* if for every polynomial $q(\lambda)$ and sufficiently large $\lambda \in \mathbb{N}$, any adversary in $\boldsymbol{G}_{n(\lambda), s(\lambda) \cdot q(\lambda)}$ has winning probability at most $\gamma + \varepsilon$.

In particular, the game is said to be *polynomially secure* if it is $(1, \mathsf{negl}(\lambda))$-secure for some negligible function $\mathsf{negl}(\lambda)$.

We note that the function $n = n(\lambda)$ specifies the dependency of the input length n and the security parameters. The parameters λ and n are usually polynomially related, while in different settings, we may set either $n = \lambda^c$ or $n = \lambda^{1/c}$ for a constant $c > 1$.

Examples. Most standard falsifiable security games can be formalized within this framework. For instance, iO may not be formalized as a polynomial-time falsifiable game as the security only holds on functionally equivalent circuits, while the set of functionally equivalent circuits is not an NP language (unless NP = coNP). However, the security game of EF-iO can be formalized as an NP-falsifiable security game.

Example 4.1 (Security of EF-iO for circuits)**.** We can formalize the security of EF-iO for circuits [JJ22] as the following security game. Let $\mathsf{EF\text{-}iO}(C; r)$ be the obfuscation scheme and $p(\cdot)$ be a polynomial, we set $\gamma := 1/2$.

- We define the language $\mathcal{U} \in \mathsf{NP}$ as follows: $(C_1, C_2) \in \mathcal{U}$ if there is an Extended Frege proof of size $p(|C_1| + |C_2|)$ of $C_1 \equiv C_2$.
- The challenger $\mathcal{C}(1^T, x, w, A; r)$ works as follows: It parses x as (C_1, C_2), randomly generates $b \in \{1, 2\}$ and the randomness r' for EF-iO, and accepts if and only if $A(\mathsf{EF\text{-}iO}(C_b; r')) = b$. Note that the random string r is used to both sample b, r' and simulate A, while the part of the randomness for sampling b, r' is hidden from A.

Remark 9 (indistinguishability game)**.** Similar to the example above, indistinguishability-based security $\mathcal{H}_0 \approx \mathcal{H}_1$ can be converted into a security game, where given the adversary A, the challenger samples $b \in \{0, 1\}$, generates $m \leftarrow \mathcal{H}_b$, and accepts if $A(m) = b$. By setting threshold $\gamma := 1/2$, the advantage of A in this game is exactly the advantage of A to distinguish $\mathcal{H}_0$ and $\mathcal{H}_1$.

In the setting of security games defined as a *search problem* (instead of a *decision problem*), one may set the parameter γ to be 0:

Example 4.2 (Security of CRHFs)**.** We can formalize the security of CRHFs as follows. Let H be the hash family, $\mathcal{U} = \{1^n \mid n \in \mathbb{N}\}$, and $\gamma := 0$. The challenger $C(1^T, x, w, A; r)$ works as follows:

- Let $x = 1^n$. It parses r as (r_1, r_2), uses r_1 to sample a hash function $h \leftarrow H_n$ from the hash family on the input length n, and uses r_2 to simulate $A(1^n, h)$. It accepts if $A(1^n, h)$ outputs a pair $x_1, x_2 \in \{0,1\}^n$ such that $x_1 \neq x_2$ and $h(x_1) = h(x_2)$.

Security under compression. We will consider a weaker security condition that formalizes security of the cryptographic primitive when the string $x \in \mathcal{U}$ and the witness w of $x \in \mathcal{U}$ are *compressible*. Let $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ be a security game, $t(n), \ell(n)$ be functions. Let $n, s \in \mathbb{N}$ and $T = T(n)$. The *t-time ℓ-compressible only* game with parameters (n, s), denoted by $\boldsymbol{G}^{t,\ell}_{n,s}$, is defined as follows.

- The adversary picks a string $x \in \mathcal{U}$ of length n, a circuit A of size s, and a witness w for $x \in \mathcal{U}$ such that $\boldsymbol{K}^t(x\|w) \leq \ell(|x| + |w|)$.
- The challenger samples a random seed r.
- The adversary wins if and only if $\mathcal{C}(1^T, x, w, A; r) = 1$.

For simplicity, we always assume that the NP witness w for $x \in \mathcal{U}$ has length exactly $p(|x|)$ for some polynomial $p(n) \geq n$.

Definition 4.9 (Security under compression)**.** Let $t(n), \ell(n)$ be functions (called the *time bound* and *compression rate*), $\gamma \in (0,1)$ be a constant, $\varepsilon = \varepsilon(\lambda) \in (0,1)$, $s = s(\lambda)$, $n = n(\lambda)$ be functions. The game $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is said to be (s, ε)*-secure assuming t-time ℓ-compression* if for every pair of polynomials $n(\lambda), q(\lambda)$ and every sufficiently large $\lambda \in \mathbb{N}$, any adversary in $\boldsymbol{G}^{t,\ell}_{n(\lambda), s(\lambda)\cdot q(\lambda)}$ has winning probability at most $\gamma + \varepsilon$.

In particular, the game is said to be *polynomially secure* if it is $(1, \mathsf{negl})$-secure for some negligible function negl.

4.3 Master Theorem for Upgrading Security

Now we are ready to prove the main result: any falsifiable game that is secure under compressibility is also secure unless Assumption 3 does not hold. Formally:

Theorem 4.10 (Master Theorem)**.** *Let $\varepsilon(\lambda) \in (0,1)$, $s(\lambda)$, $n(\lambda) \leq s(\lambda)$, $T(n) \geq n$, $\ell(n)$ be functions, $\gamma \in (0,1)$, $\mathcal{U} \in \mathsf{NTIME}[T]$ be a language with witness length $p(n) \leq T(n)$. Let $t(n)$ be a polynomial, and $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ be a T-time falsifiable security game.*

Suppose that $\mathsf{prBPP} = \mathsf{prP}$ and there is no $\mathsf{poly}(T(n(\lambda)), s(\lambda), \varepsilon^{-1}(\lambda), \lambda)$ time algorithm that witnesses t-time ℓ-incompressible strings of length $n(\lambda) + p(n(\lambda))$. Then if $\boldsymbol{G}$ is (s, ε)-secure assuming t-time ℓ-compression, it is also $(s, 2\varepsilon)$-secure.

Proof. Fix $\varepsilon(\lambda), s(\lambda), T(n), \ell(n), \gamma, \mathcal{U}, n(\lambda), t(n)$ and the security game $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$. Let $p(n) \leq T(n)$ be the witness length for $x \in \mathcal{U}$. Suppose that there is no $\mathsf{poly}(T(n(\lambda)), s(\lambda), \varepsilon^{-1}(\lambda), \lambda)$ time algorithm that witnesses t-time ℓ-incompressible strings of length $n(\lambda) + p(n(\lambda))$.

Recall that the verifier for $\mathcal{U} \in \mathsf{NTIME}[T]$ has witness size $p = p(n)$. Let $\pi = (x, w, A) \in \{0,1\}^n \times \{0,1\}^p \times \{0,1\}^s$ be a tuple. We say that it is *incompressible* if $\boldsymbol{K}^t(x\|w) > \ell(|x| + |w|)$.

Suppose that $\boldsymbol{G}$ is (s, ε)-secure assuming t-time ℓ-compression. Towards a contradiction, assume that $\boldsymbol{G}$ is not $(s, 2\varepsilon)$-secure. By definition, there is a polynomial $q = q(\lambda)$ and an infinite sequence $\Lambda = \{\lambda_i\}_{i \in \mathbb{N}}$ of security parameters such that for every $\lambda \in \Lambda$, there is an adversary that has winning probability at least $\gamma + 2\varepsilon$ in $\boldsymbol{G}_{n(\lambda), s(\lambda) \cdot q(\lambda)}$. In other words, there is a string $x^*_\lambda \in \mathcal{U}$ of length $n(\lambda)$, a witness w^*_λ for $x^*_\lambda \in \mathcal{U}$, and a circuit A^*_λ of size $s(\lambda) \cdot q(\lambda)$ such that

$$\Pr_r[\mathcal{C}(1^T, x^*_\lambda, w^*_\lambda, A^*_\lambda; r) = 1] \geq \gamma + 2\varepsilon, \tag{4.1}$$

where $T = T(n(\lambda))$. Any such tuple $x^*_\lambda, w^*_\lambda, A^*_\lambda$ is said to be a *strong insecurity witness* for the security parameter λ.

We say that $(x, w, A) \in \{0,1\}^{n(\lambda)} \times \{0,1\}^{p(n(\lambda))} \times \{0,1\}^{s(\lambda) \cdot q(\lambda)}$ is a *weak insecurity witness* for the security parameter λ if

$$\Pr_r[\mathcal{C}(1^T, x, w, A; r) = 1] > \gamma + \varepsilon. \tag{4.2}$$

As $\boldsymbol{G}$ is secure assuming t-time ℓ-compression, it follows that for every sufficiently large λ, every $x \in \mathcal{U}$ of length $n(\lambda)$, witness w for $x \in \mathcal{U}$, and circuit A of size $s(\lambda) \cdot q(\lambda)$, if (x, w, A) is a weak insecurity witness, then it is incompressible.

Construction of the Adversary. Recall that $p(n) \leq T(n)$ is the length of the witness of $\mathcal{U}$ on the input length n. Consider the following deterministic polynomial-time algorithm $\mathcal{A}(1^m, y, \tau)$:

1. Set λ such that $m = n(\lambda) + p(n(\lambda))$. If it is impossible, reject immediately.
2. Parse y as $x \| w$, where $|x| = n(\lambda)$ and $|w| = p(n(\lambda))$. If w is not a witness of $x \in \mathcal{U}$, reject immediately.
3. Parse τ as a circuit A, where $|A| = s(\lambda) \cdot q(\lambda)$.
4. The algorithm distinguishes between the following two cases:
   - If $\Pr_r[\mathcal{C}(1^T, x, w, A; r) = 1] \geq \gamma + 2\varepsilon$, the algorithm accepts.
   - If $\Pr_r[\mathcal{C}(1^T, x, w, A; r) = 1] \leq \gamma + \varepsilon$, the algorithm rejects.

   This can be achieved in $\mathsf{poly}(T, |C|, |A|, |n|, |r|, \varepsilon^{-1})$ time assuming $\mathsf{prBPP} = \mathsf{prP}$.

One can see that the algorithm accepts y if there is a strong insecurity witness of form (y, w, A), and it rejects y if there is no weak insecurity witness of form (y, w, A). It is clear that the running time of $\mathcal{A}(1^m, y, \tau)$ is

$$\mathsf{poly}\left(n(\lambda), T(n(\lambda)), s(\lambda), (\varepsilon(\lambda))^{-1}, \lambda\right)$$

where $m = n(\lambda) + p(n(\lambda))$.

We will then prove that the algorithm $\mathcal{A}(1^m, y, \tau)$ witnesses t-time ℓ-incompressible strings of length $n(\lambda) + p(n(\lambda))$.

Completeness. Fix the input length $m = n(\lambda) + p(n(\lambda))$. Recall that for every sufficient large $\lambda \in \Lambda$, there exists $x^*_\lambda \in \mathcal{U}$ of length $n(\lambda)$, a witness w^*_λ for $x^*_\lambda \in \mathcal{U}$, and a circuit A^*_λ of size $s(\lambda) \cdot q(\lambda)$ such that

$$\Pr_r[\mathcal{C}(1^T, x^*_\lambda, w^*_\lambda, A^*_\lambda; r) = 1] \geq \gamma + 2\varepsilon.$$

In other words, $(x^*_\lambda, w^*_\lambda, A^*_\lambda)$ is a strong insecurity witness. Fix any such string $x^*_\lambda \in \mathcal{U}$, circuit A^*_λ, and NP witness w^* for $x^*_\lambda \in \mathcal{U}$, and let $y = x^*_\lambda \| w^*_\lambda$. Note that as $(x^*_\lambda, w^*_\lambda, A^*_\lambda)$ is a strong insecurity witness, it must be incompressible, i.e., $\boldsymbol{K}^t(y) > \ell(|y|)$. By the fourth step of the algorithm, for $\tau := A^*_\lambda$, $\mathcal{A}(1^m, y, \tau) = 1$. This leads to the completeness of $\mathcal{A}$.

Soundness. Suppose that the input length is not of form $m = n(\lambda) + p(n(\lambda))$, the algorithm always rejects. Otherwise, fix the input length $m = n(\lambda) + p(n(\lambda))$ such that λ is sufficiently large. For any string $y = x\|w$ such that $\boldsymbol{K}^t(y) \leq \ell(|y|)$, $|x| = n(\lambda)$, $|w| = p(n(\lambda))$, there is no weak insecurity witness of form (x, w, A) as otherwise it is compressible. As a result, $\mathcal{A}(1^m, y, \tau) = 0$ for any τ. This leads to the soundness of $\mathcal{A}$. □

4.4 Typical Parameter Regimes

Corollary 4.11. *Let $\gamma \in (0,1)$ be a constant, $\mathcal{C}$ be a polynomial-time algorithm, and $\mathcal{U} \in \mathsf{NTIME}[T]$ for $T(n) = \mathsf{poly}(n)$. Assume that Assumption 3 holds and $\mathsf{prBPP} = \mathsf{prP}$. Then there is a polynomial $t(n)$ and a function $\ell(n) = n^{o(1)}$ such that the following holds.*

Suppose that for every polynomial $n(\lambda)$, $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is polynomially secure assuming t-time ℓ-compression. Then for every polynomial $n(\lambda)$, $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is polynomially secure.

Proof. Fix $\gamma, \mathcal{C}$ and $\mathcal{U} \in \mathsf{NTIME}[T]$ for some polynomial $T(n)$. Suppose that Assumption 3 holds, by Theorem 4.7, there is a polynomial $t(m)$ and $\ell(m) = m^{o(1)}$ such that there is no $\mathsf{poly}(m)$-time algorithm that witnesses t-time ℓ-incompressible strings of length m.

Suppose that for every polynomial $n(\lambda)$, $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, \gamma, n)$ is polynomially secure assuming t-time ℓ-compression. Fix any polynomial $n(\lambda)$, our goal is to prove that $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, \gamma, n)$ is polynomially secure. Towards a contradiction, assume that it is not polynomially secure, i.e., there is a non-negligible function $\varepsilon(\lambda)$ such that it is not $(s(\lambda) = 1, \varepsilon)$-secure. Note that

$$\mathsf{poly}(T(n), s(\lambda), \varepsilon^{-1}, \lambda) = \mathsf{poly}(n(\lambda) + T(n(\lambda))),$$

we know that there is no $\mathsf{poly}(T(n), s(\lambda), \varepsilon^{-1}, \lambda)$ time algorithm that witnesses t-time ℓ-incompressible strings of length $m := n(\lambda) + T(n(\lambda))$.

Therefore, by Theorem 4.10, the game $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, \gamma, n)$ is not $(1, \varepsilon/2)$-secure assuming t-time ℓ-compression. This leads to a contradiction, as ε is non-negligible. □

We need a variant of this corollary that deals with slightly superpolynomial security. The proof is almost identical and is thus omitted.

Corollary 4.12. *Let $\gamma \in (0,1)$ be a constant, $\mathcal{C}$ be a polynomial-time algorithm, and $\mathcal{U} \in \mathsf{NTIME}[T]$ for $T(n) = \mathsf{poly}(n)$. Assume that Assumption 4 holds and $\mathsf{prBPP} = \mathsf{prP}$. Then there is a polynomial $t(n)$, a function $\ell(n) = n^{o(1)}$, and a function $s(\lambda) = \lambda^{\omega(1)}$ such that the following holds.*

Suppose that for every polynomial $n(\lambda)$, $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is $(s, 1/s)$-secure assuming t-time ℓ-compression. Then for every polynomial $n(\lambda)$, $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is $(s, 2/s)$-secure.

We will also need the parameter regime where $T(n)$ is a subexponential function and $\mathcal{U} \in \mathsf{DTIME}[T]$. This requires PCG with subexponential security.

Corollary 4.13. *Assume that Assumption 5 holds and $\mathsf{prBPP} = \mathsf{prP}$. Then there exists a constant $\delta_{\mathrm{PCG}} \in (0,1)$ such that for every constant $c > 0$ and any $0 < \delta < \delta_{\mathrm{PCG}}$, there exists a polynomial $t(n)$ and a function $\ell(n) = o(n)$ such that the following holds.*

Let $\gamma \in (0,1)$ be any constants, $T(n) = 2^{n^\delta}$, $n(\lambda) = \lambda^c$, $\mathcal{C}$ be a $\mathsf{poly}(T)$-time algorithm, and $\mathcal{U} \in \mathsf{DTIME}[T]$ (i.e. the verifier runs in time $T(n)$ and there is no witness). Suppose that the game $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is polynomially secure assuming t-time ℓ-compression, it is also polynomially secure.

Proof. Let $c > 0$ be any constant and $\delta \in (0,1)$ be sufficiently small. Fix $\gamma \in (0,1)$, $\mathcal{C}$, and $\mathcal{U} \in \mathsf{DTIME}(2^{n^\delta})$. Suppose that Assumption 5 holds, by Theorem 4.7, there is a polynomial $t(m)$, a function $\ell(m) = o(m)$, and a constant $\delta' \in (0,1)$ such that there is no $2^{m^{\delta'}} \cdot \mathsf{poly}(m)$ time algorithm that witnesses t-time ℓ-incompressible strings of length m. Fix $t(m), \ell(m), \delta'$.

Suppose that $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, \gamma, n)$ is $(1,\varepsilon)$-secure assuming t-time ℓ-compression for a negligible function $\varepsilon(\lambda)$. Fix a constant $\delta < \eta < \delta'$ and set

$$\widehat{\varepsilon}(\lambda) = \max\{\varepsilon(\lambda), 2^{-n(\lambda)^\eta}\}.$$

Since $n(\lambda) = \lambda^c$, the function $\widehat{\varepsilon}$ is negligible. Moreover, $\widehat{\varepsilon} \geq \varepsilon$, so $\boldsymbol{G}$ is also $(1,\widehat{\varepsilon})$-secure assuming t-time ℓ-compression. We will prove that $\boldsymbol{G}$ is $(1, 2\widehat{\varepsilon})$-secure.

Note that $\mathcal{U} \in \mathsf{DTIME}[2^{n^\delta}]$ and there is no witness. Set $s(\lambda) := n(\lambda)$, $T(n) := 2^{n^\delta}$, and $p(n) := 0$. This choice satisfies $n(\lambda) \leq s(\lambda)$ as required by Theorem 4.10. Since $n(\lambda)$ is polynomially bounded, the assumed $(1,\widehat{\varepsilon})$-security also gives $(s,\widehat{\varepsilon})$-security under compression. We have

$$\mathsf{poly}(T(n), s(\lambda), (\widehat{\varepsilon}(\lambda))^{-1}, \lambda) \leq 2^{O(n^\eta)} \cdot \mathsf{poly}(n) \leq 2^{n^{\delta'}} \cdot \mathsf{poly}(n)$$

for all sufficiently large n, using $(\widehat{\varepsilon}(\lambda))^{-1} \leq 2^{n^\eta}$, $n = n(\lambda) = \lambda^c$, and $\delta < \eta < \delta'$. Therefore, there is no $\mathsf{poly}(T(n), s(\lambda), (\widehat{\varepsilon}(\lambda))^{-1}, \lambda)$-time algorithm that witnesses t-time ℓ-incompressible strings of length $m := n(\lambda) + p(n(\lambda)) = n(\lambda)$. By Theorem 4.10, the game $\boldsymbol{G}$ is $(s, 2\widehat{\varepsilon})$-secure, and hence also $(1, 2\widehat{\varepsilon})$-secure. Since $2\widehat{\varepsilon}$ is negligible, this proves polynomial security. □

Part II
Applications

5 Rate-1 Obfuscation and Applications

In this section, we present the following main results on rate-1 obfuscation.

Theorem 5.1. *Assuming*

- *subexponentially-secure PCG with $\ell(n) = o(n)$ stretch (Assumption 5),*
- *polynomially secure description-succinct iO,*
- *polynomially secure pseudorandom generators,*
- $\mathsf{prBPP} = \mathsf{prP}$*,*

there exists an indistinguishability obfuscation scheme $\mathcal{O}$ for circuits $C : \{0,1\}^n \to \{0,1\}$ where the obfuscated circuit size $|\mathcal{O}(1^\lambda, C, 1^n)| = |C|(1 + o(1)) + \mathsf{poly}(n, \lambda)$.

Theorem 5.2. *Let $\rho(\lambda, \cdot)$ be a function and $\tau(\lambda, \cdot)$ be a polynomial. Assuming*

- *superpolynomially secure PCG with $\ell(n) = n^{o(1)}$ stretch (Assumption 4),*
- *superpolynomially secure fully succinct* EF*-iO,*
- *superpolynomially secure pseudorandom generators,*

there exists a polynomially secure fully succinct (ρ, τ)-EF-iO scheme $\mathsf{ef\text{-}iO}_{\rho,\tau}$ where the obfuscated program is rate-1 in the secret part of the program, i.e., $|\mathsf{ef\text{-}iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)| = |M| + \mathsf{poly}(\lambda, \log n)$.

To achieve both results, we instantiate the same construction with different parameters and different underlying obfuscation schemes. We present the construction template here and defer exact instantiations and the analyses of Theorem 5.2 and Theorem 5.1 to Section 5.2 and Section 5.1 respectively.

Ingredients and construction. We present an obfuscation scheme parameterized by functions t, ℓ, κ and p. We will use the following ingredients in the construction.

- Let iO be an obfuscation scheme. Looking forward, we choose iO to be a fully succinct EF-iO scheme to prove Theorem 5.2, and a description-succinct scheme to prove Theorem 5.1.
- Let G be a pseudorandom generator.

We now describe the construction in Figure 1. Looking forward, the parameters t, ℓ, κ, p represent the following:

- t and ℓ will be specified later based on the PCG parameters,
- κ is a function that takes as input the length of the instance in the falsifiable game, and outputs the sum of the instance and witness lengths.
- p will be a polynomial which will describe the padded runtime of the obfuscated program in the construction.

$\mathcal{O}_{t,\ell,\kappa,p}(1^\lambda, M[\mathsf{pub}], 1^m)$:

1. Let $\kappa := \kappa(|(1^m, M, M, \mathsf{pub})|)$. Let $\ell := \ell(\kappa)$ and $t := t(\kappa)$. Let τ be the runtime polynomial associated with M (given in the description of M).
2. Sample $s_0 \leftarrow \{0,1\}^\lambda$ (a PRG seed) and $r_0 \leftarrow G(s_0)$ of length $|M|$.
3. Let $\mathsf{ct}_0 = r_0 \oplus M$.
4. Let $r_1 \leftarrow \{0,1\}^{\ell(\kappa)}$ be a truly random string.
5. Let $\mathsf{ct}' = (\mathsf{ct}_0 || r_1)$ (where the two strings are concatenated without any delimiters).
6. Let $P_{|M|,\ell,s_0}[\mathsf{ct}', \mathsf{pub}]$ be the program which does the following on an input x:
   - Parse ct' as (ct_0, r_1) using $|M|$ and ℓ.
   - Compute $M \leftarrow \mathsf{ct}_0 \oplus G(s_0)$.
   - Output $M[\mathsf{pub}](x)$.

   Pad P to the explicit clock $\tau'(k) = p(\lambda, \tau, t, k)$, where k is the input length and p is specified for each instantiation.
7. Let $\Gamma \leftarrow \mathsf{iO}(1^\lambda, P_{|M|,\ell,s_0}[\mathsf{ct}', \mathsf{pub}], 1^m)$ for the machine.
8. Output (ct', Γ).

Figure 1: An obfuscation scheme $\mathcal{O}$ parameterized by functions κ, ℓ and polynomials t, p.

5.1 Rate-1 iO for Circuits

We will first prove a more general theorem where we consider all machines with split description.

Fix functions $\ell(n) = o(n)$ and a polynomial $t(n) \geq n$. Consider the scheme $\mathcal{O}_{t,\ell,\kappa,p}$ (Figure 1) where we choose:

- the function $\kappa(n) = n$,
- a sufficiently large fixed polynomial $p(\lambda, \tau, t, k)$ in λ, t, k, and $\tau(k)$, chosen to cover the PRG, parsing, and simulation costs in the proof below,
- and iO is a description-succinct iO scheme.

Let $\mathcal{O} := \mathcal{O}_{t,\ell,\kappa,p}$. Let $q(m)$ and $\tau(m)$ be polynomials, and suppose both polynomials are superlinear in m. We now define the security game $\boldsymbol{G}_{q,\tau} = (\mathcal{C}, \mathcal{U}_{q,\tau}, T, \gamma, n)$ as follows:

- Define the language $\mathcal{U}_{q,\tau} \in \mathsf{DTIME}[T]$ to be set of tuples $(1^m, M_0, M_1, \mathsf{pub})$ where the following hold:
  - $m \in \mathbb{N}$, $M_0[\mathsf{pub}], M_1[\mathsf{pub}]$ are TMs with split description,
  - $|M_b[\mathsf{pub}]| \geq q(m)$ for $b \in \{0,1\}$, and $|M_0| = |M_1|$,

- $\mathsf{EQRT}(M_0[\mathsf{pub}], M_1[\mathsf{pub}]) = 1$, and their common explicit clock is at most $\tau(n)$ on every input length $k \le m$, where $n = |(1^m, M_0, M_1, \mathsf{pub})|$,
- for all $x \in \{0,1\}^{\le m}$, $M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)$.

For an instance $(1^m, M_0, M_1, \mathsf{pub})$ of length n, we first check the size and clock conditions above, then test equivalence on every input of length at most m. This takes $2^{O(m)}\mathsf{poly}(n, \tau(n))$ time. Since the size restriction gives $n \ge q(m)$, for any fixed $1/\deg(q) < \delta < 1$, this is at most 2^{n^δ} for all sufficiently large n. We therefore set $T(n) = 2^{n^\delta}$. In particular, when $q(m) = dm^d$, we need $1/d < \delta$. For the application of Corollary 4.13, first fix $0 < \delta < \delta_{\mathrm{PCG}}$, then choose d sufficiently large that $1/d < \delta$.

- The challenger $\mathcal{C}$ takes as input $(1^T, (1^m, M_0, M_1, \mathsf{pub}), \perp, \mathcal{A}; (b, r, r_{\mathcal{A}}))$ where $T = T(n)$,
  - Check the size and clock conditions above, and that for all $x \in \{0,1\}^{\le m}$, $M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)$. Reject if any check fails.
  - Let $\Gamma \leftarrow \mathcal{O}(1^\lambda, M_b[\mathsf{pub}], 1^m; r)$.
  - Compute $\beta \leftarrow \mathcal{A}(\Gamma; r_A)$.
  - If $b = \beta$, accept. Else, reject.
- Set $\gamma = 1/2$.

We say that an obfuscation scheme $\mathcal{O}$ is polynomially secure for families of (q, τ) programs if $\boldsymbol{G}_{q,\tau}$ is polynomially secure when instantiated with $\mathcal{O}$.

Theorem 5.3. *Let q and τ be superlinear polynomials. Assuming the polynomial-security of description-succinct iO and PRGs, the game $\boldsymbol{G}^{t,\ell}_{q,\tau}$ when instantiated with $\mathcal{O}$ satisfies polynomial-security under t-time ℓ-compression.*

Proof. Fix polynomial bounds on the instance length n and adversary size. All reductions below have size polynomial in λ, uniformly over instances satisfying these bounds, since t and τ are fixed polynomials. The underlying iO and PRG security therefore give a negligible bound that depends only on these polynomial bounds, rather than on the particular instance or its shared clock.

Suppose that $\boldsymbol{K}^t(1^m, M_0, M_1, \mathsf{pub}) \le \ell(\kappa)$, and suppose the machine S witnesses this fact. Let $U_S[\mathsf{pub}](x)$ denote the following machine:

- Use S to compute $(1^m, M_0, M_1, \mathsf{pub})$.
- Output $M_1[\mathsf{pub}](x)$.

On every input x with $|x| \le m$, $U_S[\mathsf{pub}](x)$ decompresses the instance in at most $t(\kappa)$ time and evaluates $M_1[\mathsf{pub}](x)$ in at most $\tau(n)$ time. Its running time is therefore $O(t(\kappa) + \tau(n))$.

We now argue via a sequence of hybrids. It suffices to prove that $\mathcal{H}_{1,0}$ and $\mathcal{H}_{1,1}$ are computationally indistinguishable.

$\mathcal{H}_{1,b}$: Sample $(\mathsf{ct}', \Gamma) \leftarrow \mathcal{O}(1^\lambda, M_b[\mathsf{pub}], 1^m)$ and write $\mathsf{ct}' = \mathsf{ct}_0 || r_1$.

$\mathcal{H}_{2,b}$: In this hybrid, replace r_1 with $\mathsf{ct}_1 = r_1 \oplus U_S$. Sample the rest of the program as before. Indistinguishability follows from the fact that both hybrids have identical distributions.

$\mathcal{H}_{3,b}$: In this hybrid, sample $s_1 \leftarrow \{0,1\}^\lambda$, and let $r_1 \leftarrow G(s_1)$. Sample the rest of the program as before. Indistinguishability follows from the security of the PRG.

$\mathcal{H}_{4,b}$: In this hybrid, switch $P_{|M_b|,\ell,s_0}[\mathsf{ct}', \mathsf{pub}](x)$ to the following program $Q_{|M_b|,\ell,s_1}[\mathsf{ct}', \mathsf{pub}](x)$:

- Parse $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$ using $|M|$ and ℓ.
- Compute $U_S \leftarrow \mathsf{ct}_1 \oplus G(s_1)$.
- Output $U_S[\mathsf{pub}](x)$.

On every input x with $|x| \le m$, the runtime of the above machine is at most $\mathsf{poly}(\lambda, n) + O(t(\kappa) + \tau(n))$, where the polynomial term accounts for parsing, computing $G(s_1)$, and recovering U_S. Since $\kappa = n$ and t, τ are fixed polynomials, all hybrid programs have polynomial size and construction time in λ under the fixed polynomial bound on n. Let τ_M be the common explicit clock of $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$; thus $\tau_M(k) \le \tau(n)$ for every $k \le m$. We give Q exactly the padded clock $\tau'(k) = p(\lambda, \tau_M, t(\kappa), k)$ used for P in Figure 1. Both programs have preprocessing costs polynomial in λ, n and simulate a machine with clock $\tau_M(k)$; Q additionally decompresses the instance in time $t(\kappa)$. Since $n = \kappa$ and $t(\kappa) \ge \kappa$, our fixed choice of p dominates both runtimes.

The clock depends only on λ, the common description lengths, pub, m, and τ_M, so it is independent of the PRG seeds and the challenge bit. We also use equal-length encodings of the fixed templates P and Q, padding by a constant amount if needed.

Sample $\Gamma \leftarrow \mathsf{iO}(1^\lambda, Q_{|M_b|,\ell,s_1}[\mathsf{ct}', \mathsf{pub}], 1^m)$, and output $(\mathsf{ct}' = (\mathsf{ct}_0 || \mathsf{ct}_1), \Gamma)$.

The two programs therefore have equal description lengths and identical explicit clocks, and agree on all inputs of length at most m. Indistinguishability follows from description-succinct iO security.

$\mathcal{H}_{5,b}$: Sample $\mathsf{ct}_0 = r_0 \oplus M_b$, where $r_0 \leftarrow \{0,1\}^{|M_b|}$ (rather than being sampled from the PRG seed s_0). Sample the rest of the hybrid as in the previous hybrid (note that s_0 is no longer used in the obfuscated program). Indistinguishability follows from PRG security.

At this point, it is easy to see that $\mathcal{H}_{5,0}$ and $\mathcal{H}_{5,1}$ have identical distributions. Therefore, via a hybrid argument, we have that $\mathcal{H}_{1,0}$ and $\mathcal{H}_{1,1}$ are indeed indistinguishable. This completes the proof of indistinguishability. □

We next lift the compressed-security guarantee uniformly over all instance lengths between λ and a fixed polynomial in λ.

Corollary 5.4. *Let τ be a polynomial. Assuming*

- *subexponentially-secure PCG with $\ell(n) = o(n)$ stretch (Assumption 5),*
- *polynomially secure description-succinct iO,*
- *polynomially secure pseudorandom generators,*
- $\mathsf{prBPP} = \mathsf{prP}$,

there exist a polynomial $t(n) \ge n$, a function $\ell(n) = o(n)$, and a constant $d_0 \in \mathbb{N}$ such that the following holds. For every integer $d \ge d_0$, set $q(m) = dm^d$ and instantiate $\mathcal{O}$ with these parameters. For every integer $c \ge 1$ and polynomial q_1, there is a negligible function μ such that, for every λ, every valid instance of $\boldsymbol{G}_{q,\tau}$ of length n satisfying $\lambda \le n \le \lambda^c$, and every adversary of size at most $q_1(\lambda)$, the winning probability is at most $1/2 + \mu(\lambda)$.

Proof. By Assumption 5 and theorem 4.7, there exist a polynomial $t(n) \geq n$, a function $\ell(n) = o(n)$, and a constant $\delta_{\mathrm{PCG}} \in (0,1)$ such that no $2^{n^{\delta_{\mathrm{PCG}}}}\mathsf{poly}(n)$-time algorithm witnesses t-time ℓ-incompressible strings. Fix these parameters. Choose a constant $0 < \delta < \delta_{\mathrm{PCG}}$ and an integer d_0 such that $1/d_0 < \delta$. For every $d \geq d_0$, the size restriction $n \geq q(m) = dm^d$ gives

$$2^{O(m)}\mathsf{poly}(n, \tau(n)) \leq 2^{n^{\delta}}$$

for all sufficiently large n, as shown above. Thus membership in $\mathcal{U}_{q,\tau}$ can be checked within $T(n) = 2^{n^{\delta}}$ time. We use the empty membership witness and $\kappa(n) = n$.

Fix $d \geq d_0$, an integer $c \geq 1$, and a polynomial q_1. By Theorem 5.3, there is a negligible bound on the winning advantage that holds uniformly over all t-time ℓ-compressible instances with $\lambda \leq n \leq \lambda^c$ and adversaries of size at most $q_1(\lambda)$. Suppose, towards a contradiction, that the claimed uniform security fails without the compression restriction. Then there is a fixed integer $k \geq 1$ and infinitely many λ for which some valid instance x_λ of length $n_\lambda \in [\lambda, \lambda^c]$ and adversary A_λ of size at most $q_1(\lambda)$ have winning probability at least $1/2 + \lambda^{-k}$.

We adapt the incompressibility verifier in the proof of Theorem 4.10. On input $(1^n, x)$, its certificate contains $(1^\lambda, A)$. It checks $|x| = n$, $\lambda \leq n \leq \lambda^c$, $|A| \leq q_1(\lambda)$, and $x \in \mathcal{U}_{q,\tau}$, rejecting if any check fails. It then distinguishes winning probability at least $1/2 + \lambda^{-k}$ from winning probability at most $1/2 + \lambda^{-k}/2$ in the game at security parameter λ, accepting in the first case and rejecting in the second. As in Theorem 4.10, this test is deterministic under $\mathsf{prBPP} = \mathsf{prP}$ and runs in time

$$\mathsf{poly}(T(n), n, q_1(\lambda), \lambda^k, \lambda) = 2^{O(n^{\delta})}\mathsf{poly}(n),$$

since $\lambda \leq n$. This is within the $2^{n^{\delta_{\mathrm{PCG}}}}\mathsf{poly}(n)$ hardness bound.

To prove soundness, fix a sufficiently large n and a string x of length n with $\boldsymbol{K}^t(x) \leq \ell(n)$. Every certificate passing the length checks has $\lambda \geq n^{1/c}$. Thus λ is sufficiently large for the uniform compressed-security bound to be at most $\lambda^{-k}/2$, and the verifier rejects every such certificate. For completeness, the verifier accepts $(1^{n_\lambda}, x_\lambda)$ with certificate $(1^\lambda, A_\lambda)$ for every sufficiently large attack parameter above. Soundness implies that these strings are incompressible. Since $n_\lambda \geq \lambda$, they occur at infinitely many distinct lengths. This contradicts the hardness of witnessing t-time ℓ-incompressible strings and proves the claimed uniform security. □

We are now ready to prove Theorem 5.1.

Proof of Theorem 5.1. We use a fixed circuit-evaluation wrapper that represents an m-input circuit C of size $S = |C|$ by a TM M_C of size $S + O(\log(S + m + 2))$. The machine evaluates C on inputs of length m and outputs 0 on all other input lengths. We give it a canonical constant clock $\tau_C(|C|, m) = \mathsf{poly}(|C| + m)$, chosen to be at least $|C| + m + 1$ and identical for all circuits with the same size and input length.

Fix a superlinear polynomial $\tau(N)$ dominating $2\tau_C(S, m)$ whenever $S, m \leq N$. Let iO be a description-succinct iO scheme, and let $\mathcal{O}$ be the scheme shown secure in Corollary 5.4 for this fixed τ. Choose $d \geq d_0$ as in that corollary and set $q(m) = dm^d$.

Consider the following scheme:

$\widehat{\mathcal{O}}(1^\lambda, C, 1^m)$:

- If $|C| \leq q(\lambda)$ or $|C| \leq q(m)$, output $\mathsf{iO}(1^\lambda, M_C, 1^m)$.
- Else,

- Let j be the smallest nonnegative integer such that $2^j \max\{1, q(m)\} \geq \tau_C(|C|, m)$. Give M_C the constant clock $2^j \max\{1, q(m)\}$.
- Output $\mathcal{O}(1^\lambda, M_C, 1^m)$.

In the first branch, $S \leq \max\{q(\lambda), q(m)\}$, so the output has size $\mathsf{poly}(q(\lambda) + q(m)) = \mathsf{poly}(m, \lambda)$. In the second branch, $S > q(m) \geq m$ and $\kappa = \Theta(S)$. The circuit wrapper and its clock use $S + O(\log(S + 2))$ bits, while the private description of the inner program has size polynomial in λ and $\log(S + 2)$. By description-succinctness, the output therefore has size

$$S + \ell(\kappa) + \mathsf{poly}(m, \lambda, \log(S + 2)).$$

We have $\ell(\kappa) = o(S)$. Separating mixed terms using $uv \leq u^2 + v^2$ bounds the polynomial term by $\mathsf{poly}(m, \lambda) + \mathsf{poly}(\log(S + 2))$, where the last term is also $o(S)$. Thus the output size is at most $|C|(1 + o(1)) + \mathsf{poly}(m, \lambda)$, which is indeed rate-1. The correctness of the above scheme follows from the correctness of both iO and $\mathcal{O}$. The rounding of the circuit-evaluation clock increases it by less than a factor of two; the subsequent padding and underlying obfuscation have polynomial slowdown.

We now prove security. Fix polynomials q_1, q_2. We show that there is a negligible function μ such that, for every λ, every adversary $\mathcal{A}$ of size at most $q_1(\lambda)$, and every pair of equivalent circuits $C_0, C_1 : \{0,1\}^m \to \{0,1\}$ with $S = |C_0| = |C_1| \leq q_2(\lambda)$ and $m \leq q_2(\lambda)$,

$$\Pr\left[\mathcal{A}(\Gamma) = b \ : \ \begin{array}{c} b \leftarrow \{0,1\} \\ \Gamma \leftarrow \widehat{\mathcal{O}}(1^\lambda, C_b, 1^m) \end{array} \right] \leq \frac{1}{2} + \mu(\lambda).$$

The branch condition depends only on S, m, λ, so both challenge circuits take the same branch.

If $S \leq q(\lambda)$ or $S \leq q(m)$, the challenge is an underlying iO obfuscation of the circuit-evaluation wrapper. The wrappers have equal description lengths, identical canonical clocks, and agree on all inputs of length at most m. Their sizes and input bound are polynomial in λ under the fixed bound q_2. Underlying iO security therefore gives a negligible winning-advantage bound μ_{iO}, uniformly over all challenges in this branch.

Otherwise, $S > q(\lambda)$ and $S > q(m)$. Let M_0, M_1 be the clocked wrappers produced in this branch, take the public part pub to be empty, and set

$$N = |(1^m, M_0, M_1, \mathsf{pub})|.$$

The wrappers have equal description lengths, each at least $S > q(m)$, so the construction computes $\kappa = N$. They agree on all inputs of length at most m, since C_0, C_1 are equivalent and both wrappers output 0 on shorter inputs. Their common clock is determined by S, m and is at most $2\tau_C(S, m) \leq \tau(N)$ on every such input. Thus $(1^m, M_0, M_1, \mathsf{pub}) \in \mathcal{U}_{q,\tau}$ for the single fixed polynomial τ chosen above.

Moreover, $S > q(\lambda) \geq \lambda$, and the fixed wrapper and clock encodings have size polynomial in $S + m$. Hence there is an integer $c \geq 1$, depending only on q_2 and these encodings, such that for all sufficiently large λ,

$$\lambda \leq S \leq N \leq \mathsf{poly}(q_2(\lambda)) \leq \lambda^c.$$

Applying Corollary 5.4 once, with this c and the adversary-size bound q_1, gives a negligible winning-advantage bound $\mu_{\mathcal{O}}$ uniformly over all challenges in the second branch.

Taking $\mu = \mu_{\mathsf{iO}} + \mu_{\mathcal{O}}$ and enlarging it on finitely many small security parameters if necessary proves the displayed security bound. Both bounds are uniform over their respective branches, so the same μ applies even when the branch varies with the challenge instance or security parameter. □

5.2 Rate-1 iO for EF-Equivalent Programs

Fix a function $\ell(n) = n^{o(1)}$ and a polynomial t. Consider the following new family of EF-iO scheme: for polynomials $\rho(\lambda, \cdot)$ and $\tau(\lambda, \cdot)$, we define $\mathsf{ef\text{-}iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$ to be $\mathcal{O}_{t,\ell,\kappa,p}$ (Figure 1) where:

- $\kappa(n) = n + \rho + O(1)$ and p is a polynomial which will be specified in Lemma 5.6,
- and iO is an $(O(1), p)$-EF-secure scheme.

Consider the following security game $\boldsymbol{G}_{\rho,\tau} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ which formalizes the (ρ, τ)-EF security.

- Define the language $\mathcal{U}_{\rho,\tau}$ of tuples $(1^m, M_0, M_1, \mathsf{pub})$ where the following hold:
  - $m \in \mathbb{N}$, $M_0[\mathsf{pub}], M_1[\mathsf{pub}]$ are TMs with split description,
  - for all $x \in \{0,1\}^{\leq m}$, $\mathsf{EQRT}(M_0[\mathsf{pub}]) = \mathsf{EQRT}(M_1[\mathsf{pub}])$, where the clock is given by τ,
  - there exists a machine $\mathcal{M}_\Pi$ which on input $(1^k, M_0, M_1, \mathsf{pub})$, in $\tau(k)$ time outputs $[\![M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)]\!]_k$ for all $k \leq m$.

  In words, this is the language of tuples $(1^m, M_0, M_1, \mathsf{pub})$ where are (ρ, τ)-equivalent for inputs of length at most m for some $\rho \in \mathbb{N}$.

  The above language can be decided in time $\mathsf{poly}(m, \tau(m)) = \mathsf{poly}(n)$.
- The challenger $\mathcal{C}$ takes as input $(1^T, (1^m, M_0, M_1, \mathsf{pub}), \mathcal{M}_\Pi, A; (b, r, r_A))$ where $T = \mathsf{poly}(n)$,
  - Use $\mathcal{M}_\Pi$ to check if $(1^m, M_0, M_1, \mathsf{pub}) \in \mathcal{U}_{\rho,\tau}$.
  - Let $\Gamma \leftarrow \mathcal{O}_{\rho,\tau}(1^\lambda, M_b, m; r)$.
  - Compute $\beta \leftarrow \mathcal{A}(\Gamma; r_A)$.
  - If $b = \beta$, accept. Else, reject.
- Set $\gamma = 1/2$.

We say that $\mathcal{O}_{\rho,\tau}$ is polynomially (ρ, τ)-EF-secure if $\boldsymbol{G}_{\rho,\tau}$ is polynomially secure when instantiated with $\mathcal{O}_{\rho,\tau}$.

Theorem 5.5. *Assuming the superpolynomial security of* EF*-iO security and PRGs, the game* $\boldsymbol{G}^{t,\ell}_{\rho,\tau}$ *when instantied via* $\mathsf{ef\text{-}iO}_{\rho,\tau}$ *satisfies superpolynomial security under* t*-time* ℓ*-compression.*

Proof. Consider machines $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$ which are (ρ, τ)-EF-equivalent via machine $\mathcal{M}_\Pi$ for inputs of length at most m. WLOG, we assume that $M_b[\mathsf{pub}](x)$ has runtime $\tau(|x|)$.

Suppose that $\boldsymbol{K}^t(m, M_0, M_1, \mathsf{pub}, \mathcal{M}_\Pi) \leq \ell(\kappa)$, and suppose the machine S witnesses this fact. Let $U_S[\mathsf{pub}](x)$ denote the following machine:

- Use S to compute $(m, M_0, M_1, \mathsf{pub}, \mathcal{M}_\Pi)$.
- Output $M_1[\mathsf{pub}](x)$.

Clearly, $U_S[\mathsf{pub}](x)$ runs in time $\tau(|x|) + t(\kappa) + O(1)$.

We now outline the hybrids.

$\mathcal{H}_{1,b}$: Sample $(\mathsf{ct}' = (\mathsf{ct}_0, r_1), \Gamma \leftarrow \mathsf{iO}(1^\tau, M_b[\mathsf{pub}], 1^m))$.

$\mathcal{H}_{2,b}$: In this hybrid, replace r_1 with $\mathsf{ct}_1 = r_1 \oplus U_S$. Sample the rest of the program as before. Indistinguishability follows from the fact that both hybrids have identical distributions.

$\mathcal{H}_{3,b}$: In this hybrid, sample $s_1 \leftarrow \{0,1\}^\lambda$, and let $r_1 \leftarrow G(s_1)$. Sample the rest of the program as before. Indistinguishability follows from the security of the PRG.

$\mathcal{H}_{4,b}$: In this hybrid, switch $P_{|M_b|,\ell,s_0}[\mathsf{ct}', \mathsf{pub}](x)$ to the following program $Q_{|M_b|,\ell,s_1}[\mathsf{ct}', \mathsf{pub}](x)$:

- Parse $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$ using $|M|$ and ℓ.
- Compute $U_S \leftarrow \mathsf{ct}_1 \oplus G(s_1)$.
- Output $U_S[\mathsf{pub}](x)$.

Sample $\Gamma \leftarrow \mathsf{iO}(1^\lambda, Q_{|M_b|,\ell,s_1}[\mathsf{ct}', \mathsf{pub}], 1^m)$, and output $(\mathsf{ct}' = (\mathsf{ct}_0||\mathsf{ct}_1), \Gamma)$.

Lemma 5.6. *The machines $P_{|M_b|,\ell,s_0}[\mathsf{ct}', \mathsf{pub}](x)$ and $Q_{|M_b|,\ell,s_1}[\mathsf{ct}', \mathsf{pub}](x)$ are $(O(1), \mathsf{poly}(\lambda, \tau, t))$-$\mathsf{EF}$-equivalent.*

Therefore, we can invoke the $(O(1), \tau')$-EF security of iO to argue indistinguishability. We present the proof of Lemma 5.6 after we state all of the hybrids.

$\mathcal{H}_{5,b}$: Sample $\mathsf{ct}_0 = r_0 \oplus M_b$, where $r_0 \leftarrow \{0,1\}^{|M_b|}$ (rather than being sampled from the PRG seed s_0). Sample the rest of the hybrid as in the previous hybrid (note that s_0 is no longer used in the obfuscated program). Indistinguishability follows from PRG security.

At this point, it is easy to see that $\mathcal{H}_{5,0}$ and $\mathcal{H}_{5,1}$ have identical distributions. Therefore, via a hybrid argument, we have that $\mathcal{H}_{0,0}$ and $\mathcal{H}_{0,1}$ are indeed indistinguishable. This completes the proof of indistinguishability. It remains to show Lemma 5.6.

Proof of Lemma 5.6. We will use the shorthand $P := P_{|M_b|,\ell,s_0}$ and $Q := Q_{|M_b|,\ell,s_1}$. If $b = 1$, then we have the following PV proof of equivalence of $P_{|M_1|,\ell,s_0}[\mathsf{ct}', \mathsf{pub}](x)$ and $Q_{|M_1|,\ell,s_1}[\mathsf{ct}', \mathsf{pub}]$:

- By definition, for all $x \in \{0,1\}^{\leq k}$, $P_{|M_1|,\ell,s_0}[\mathsf{ct}', \mathsf{pub}](x) = M_1[\mathsf{pub}](x)$.
- By definition, for all $x \in \{0,1\}^{\leq k}$, $Q_{|M_1|,\ell,s_1}[\mathsf{ct}', \mathsf{pub}](x) = U_S[\mathsf{pub}](x)$.
- By definition, for all $x \in \{0,1\}^{\leq k}$, $U_S[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)$.
- By transitivity, we therefore have that the two programs are equivalent.

The corresponding EF proof can be generated by a machine of size $O(1)$ given $1^k, P, Q, \mathsf{pub}$ as input in time $\mathsf{poly}(|P| + |Q| + |\mathsf{pub}|, t, k)$.

Suppose $b = 0$. Then, we construct the following machine $\mathcal{M}'_\Pi$ which shows EF-equivalence as follows: on input $1^k, P, Q, (\mathsf{ct}', \mathsf{pub})$

- By definition, prove that all $x \in \{0,1\}^k$, $P_{|M_1|,\ell,s_0}[\mathsf{ct}', \mathsf{pub}](x) = M_1[\mathsf{pub}](x)$.
- By definition, prove that for all $x \in \{0,1\}^k$, $Q_{|M_1|,\ell,s_1}[\mathsf{ct}', \mathsf{pub}](x) = U_S[\mathsf{pub}](x)$.
- By definition, prove that for all $x \in \{0,1\}^k$, $U_S[\mathsf{pub}](x) = M_0[\mathsf{pub}](x)$.
- Parse S via Q and ct' (i.e. use s_1 in the description of Q to decrypt ct_1).

- Compute $(m, M_0, M_1, \mathsf{pub}, \mathcal{M}_\Pi)$ from S.
- Compute $\mathcal{M}_\Pi(1^k, M_0, M_1, \mathsf{pub})$ and output $[\![M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)]\!]_k$.

The corresponding EF proof can be generated in time $\mathsf{poly}(|P| + |Q| + |\mathsf{pub}|, t, k)$.

Therefore, for a large enough polynomial p, we prove our claim. □

This completes the proof. □

By now invoking our master theorem following Corollaries 4.11 and 4.12, we obtain Theorem 5.2 as an immediate corollary.

We note show that the above scheme is in fact secure for *all* TMs if there exists an rate-1 iO scheme with a PV proof of its correctness property.

Theorem 5.7 (Universal rate-1 iO for TMs with a PV proof of correctness)**.** *Assuming*

- *superpolynomially secure PCG with $\ell(n) = n^{o(1)}$ stretch (Assumption 4),*
- *superpolynomially secure fully succinct* EF*-iO,*
- *superpolynomially secure pseudorandom generators,*
- existence *of a superpolynomially secure rate-1 iO scheme for TMs with split description whose correctness is provable in Cook's theory PV (Definition 3.14),*

there exists a superpolynomially secure fully succinct iO scheme for TMs where $|\mathsf{ef\text{-}iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)| = |M| + \mathsf{poly}(\lambda, \log n)$, i.e. the obfuscated program is rate-1 in the secret part of the program.

Proof. Let $\mathcal{O}$ denote a rate-1 EF-iO scheme from Theorem 5.2. Suppose iO is some rate-1 iO scheme with PV provable correctness satisfying the following properties:

- $\mathsf{iO}(1^\lambda, M[\mathsf{pub}], n)$ outputs obfuscations of size $f(n, |M|, \lambda) = |M|(1 + o(1)) + p(\lambda)$ with runtime slowdown given by a polynomial q.
- The PV proof of correctness can be translated to a uniform EF proof of correctness with runtime p_π.

Consider the following new obfuscation scheme:

$\widehat{\mathcal{O}}(1^\lambda, M[\mathsf{pub}], 1^n)$:

- Pad M to size $f(n, |M|, \lambda)$ and pad the runtime to be $q \circ \tau$, where τ was the original runtime of $M[\mathsf{pub}]$.
- Output $\mathcal{O}(1^\lambda, M[\mathsf{pub}], 1^n)$, where $\mathcal{O}$ is a rate-1 EF-iO which is $(O(\lambda), p_\pi)$-secure.

Since $f(n, |M|, \lambda) = |M|(1 + o(1)) + p(\lambda)$ and $\mathcal{O}$ is rate-1, $\widehat{O}$ outputs an obfuscation of size $|M|(1 + o(1)) + \mathsf{poly}(\lambda)$, i.e. is still rate-1.

It is now sufficient to show security of scheme for arbitrary functionally equivalent programs $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$. We will use the rate-1 iO scheme iO in the following security proof.

$\mathcal{H}_{1,b}$: In this hybrid, output $\widehat{O}(1^\lambda, M_b[\mathsf{pub}], 1^n)$.

$\mathcal{H}_{2,b}$: In this hybrid, first sample $\Gamma_b \leftarrow \mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n; r)$. Then, output $O(1^\lambda, \Gamma_b[\mathsf{pub}], 1^n)$. Note here that we are using $\mathcal{O}$ rather than $\widehat{O}$.

Note that given r, M_b and Γ_b are $(O(\lambda), p_\pi)$-EF equivalent via the proof of correctness of the iO scheme. Therefore, indistinguishability follows from the EF-iO security of $\mathcal{O}$.

Now, note that $\mathcal{H}_{2,0}$ and $\mathcal{H}_{2,1}$ are indistinguishable by relying on iO security to say that Γ_0 and Γ_1 are indistinguishabe distributions. This completes the proof. □

Remark 10. Since the obfuscation algorithms in Theorem 5.2 and Theorem 5.7 have runtime independent of n (only grows polynomially with $\log n$), one can now remove 1^n from the syntax of the obfuscation and give the algorithm n as input instead. Moreover, assuming superpolynomial security, one can remove n entirely and support arbitrary unbounded (polynomial) input lengths via the standard powers-of-two trick as follows:

- Let s be the superpolynomial function in Assumption 4, and let s' be a superpolynomial function satisfying $s'(\lambda) \leq s(\lambda)$, and let $k = \log_2 s'(\lambda)$.
- Consider a new obfuscation scheme which samples $\Gamma_i \leftarrow \mathcal{O}(1^\lambda, M[\mathsf{pub}], \lambda^i)$, for $i \in [k]$ and outputs $\{\Gamma_i\}_{i\in[k]}$. To evaluate the obfuscation on some x of size $\mathsf{poly}(\lambda)$, find $j = \lceil \log_2 |x| \rceil$, and evaluate Γ_j. It is clear that the scheme now supports arbitrary $\mathsf{poly}(\lambda)$-sized inputs. Security of the new scheme follows from the security of $\mathcal{O}$.

5.3 Applications to Witness Encryption

In this section, we will consider witness encryption for the following classes of relations.

Definition 5.8 (Relation with (ρ, τ)-EF proof of non-membership)**.** We say that an efficient machine R defines a relation with a (ρ, τ)-EF proof of non-membership if for every instance x such that $R(x, w)$ outputs 0 for all w, the machines with split descriptions $R[x]$ and $\mathsf{Null}[x]$ are $(\rho(|x|), \tau(|x|))$-EF equivalent (Definition 3.15), where $R[x](w) = R(x, w)$ and $\mathsf{Null}[x](w) = 0$ for all w.

Definition 5.9 (Total relation with (ρ, τ)-EF proof of unambiguity)**.** We say that an efficient machine R defines a relation with a (ρ, τ)-EF proof of unambiguity if for every instance x, there exists a machine $\mathcal{M}_\Pi$ of size at most $\rho(|x|)$ that takes as input a length bound 1^n and outputs an EF proof that

$$[\![R[x](w) = R[x](w') = 1 \rightarrow w = w']\!]_n$$

in $\tau(|x|, n)$ time. We say that R defines a *total* relation with m-bounded witnesses and (ρ, τ)-EF proof of unambiguity if, additionally, for every x, there exists a witness w of length at most $m(|x|)$ such that $R(x, w) = 1$.

WE construction. From an EF-iO construction, we immediately get a witness encryption scheme for relations with (ρ, τ)-EF proof of non-membership, and an extractable witness encryption scheme for total relations with m-bounded witnesses and (ρ, τ)-EF proof of unambiguity. Furthermore, we show in Theorem 5.12 that by using rate-1 EF-iO with succinctness in the size of the EF proof, the resulting witness encryption scheme is fully succinct.

Construction 1 (Succinct witness encryption)**.** Let $\mathsf{iO}_{\rho,\tau}$ be a EF-iO for (split-description) machines, where (ρ, τ) are the parameters of the EF proofs. Let ρ_0, τ_0 be appropriate functions to be determined later in Theorems 5.10 and 5.11. We construct a witness encryption scheme $\mathsf{WE} = (\mathsf{Enc}, \mathsf{Dec})$ as follows.

- $\mathsf{Enc}(1^\lambda, R, x, n, b)$: On input a security parameter λ, a machine R, an instance x, a witness length bound n, and a message bit b, define the split-description machine $M_{R,n,b}[x](w)$ which outputs b if $|w| \leq n$ and $R(x, w)$ outputs 1, and outputs 0 otherwise. Output

$$\mathsf{ct} = \widetilde{M_{R,n,b}}[x] \leftarrow \mathsf{iO}_{\rho_0(|x|),\tau_0(|x|,n)}(1^\lambda, M_{R,n,b}[x], 1^n).$$

- $\mathsf{Dec}(1^\lambda, \mathsf{ct}, R, x, w)$: On input a security parameter λ, a ciphertext $\mathsf{ct} = \widetilde{M_{R,n,b}}[x]$, a machine R, an instance x and a witness w, output $b \leftarrow \widetilde{M_{R,n,b}}[x](w)$.

The correctness and efficiency follow immediately from the properties of EF-iO. We now argue the security and extractability of the construction when ρ_0 and τ_0 are appropriately set. We argue the (full) succinctness of the construction when instantiated with appropriate EF-iO constructions in Theorem 5.12.

Theorem 5.10. *Let $\rho_0 = \rho(|x|) + O(1)$, $\tau_0 = O(\tau(|x|)) + \mathsf{poly}(\lambda)$, then Construction 1 is a witness encryption for all relations with (ρ, τ)-EF proof of non-membership.*

Proof. To he security of the witness encryption essentially states that for false statement x, $\widetilde{M_{R,n,0}}[x] \leftarrow \mathsf{Enc}(1^\lambda, R, x, n, 0)$ and $\widetilde{M_{R,n,1}}[x] \leftarrow \mathsf{Enc}(1^\lambda, R, x, n, 1)$ are computationally indistinguishable. Toward this, it suffices to argue that the machines $M_{R,n,0}[x]$ and $M_{R,n,1}[x]$ are $(\rho_0(|x|), \tau_0(|x|, n))$-EF equivalent. First, since $M_{R,n,0}[x]$ always outputs 0 by definition, we have that $M_{R,n,0}[x]$ and $\mathsf{Null}[x]$ are $(O(1), \mathsf{poly}(\lambda))$-EF equivalent. Second, since $M_{R,n,1}[x]$ outputs 1 if and only if $R(x, w)$ outputs 1 for some w of length at most n, and that there is a (ρ, τ)-EF proof of $R(x, w) = 0$ for all w, we have that $M_{R,n,1}[x]$ and $\mathsf{Null}[x]$ are $(\rho(|x|) + O(1), O(\tau(|x|)) + \mathsf{poly}(\lambda))$-EF equivalent. This implies that $M_{R,n,0}[x]$ and $M_{R,n,1}[x]$ are $(\rho_0(|x|), \tau_0(|x|, n))$-EF equivalent, and thus the security of the construction follows from the security of the iO. □

Theorem 5.11. *Let $\rho_0 = \rho(|x|) + O(m(|x|))$, $\tau_0 = \mathsf{poly}(\lambda, \tau(|x|, m(|x|)), \mathsf{RT}(R, m(|x|) + |x|))$, where $\mathsf{RT}(R, L)$ denotes the runtime of machine R on inputs of length at most L, then Construction 1 is an extractable witness encryption for all total relations with m-bounded witnesses (ρ, τ)-EF proof of unambiguity.*

Proof. Fix the relation R in the class and a statement x, since R is total, there exists a witness w_x of length $m_x \leq m(|x|)$ such that $R(x, w_x) = 1$. We first argue that, if $n < m_x$, the WE ciphertexts $\widetilde{M_{R,n,0}}[x]$ and $\widetilde{M_{R,n,1}}[x]$ are indistinguishable, therefore no adversary $\mathcal{A}$ satisfying the precondition of the extractability can exist. Indeed, in this case, both $M_{R,n,0}[x]$ and $M_{R,n,1}[x]$ always output 0. By definition, $M_{R,n,0}[x]$ and $\mathsf{Null}[x]$ are $(O(1), \mathsf{poly}(\lambda))$-EF equivalent. Moreover, there is a $\rho(|x|)$ sized machine outputting an EF proof that $[\![R[x](w) = R[x](w') = 1 \to w = w']\!]_{m_x}$, with variables w and w' of length m_x. Combined with the trivial machine hardcoding (x, w_x) and outputting an EF proof that $[\![R[x](w_x) = 1]\!]$ in time $\mathsf{poly}(\mathsf{RT}(R, |x| + |w_x|))$, there exist a $\rho(|x|) + O(m(|x|))$ sized uniform EF proof that $[\![|w| < m_x \to R[x](w) = 0]\!]$, which implies that $M_{R,n,1}[x]$ and $\mathsf{Null}[x]$ are $(\rho(|x|) + O(m(|x|)), O(\tau(|x|, m(|x|))) + \mathsf{poly}(\mathsf{RT}(R, |x| + m(|x|))))$-EF equivalent. Thus, by the security of the iO, we have that $\widetilde{M_{R,n,0}}[x]$ and $\widetilde{M_{R,n,1}}[x]$ are computationally indistinguishable.

Next, for the case $n \geq m_x$, we describe the extractor $\mathcal{E}$, which makes black box access to the adversary $\mathcal{A}$ with (known) advantage $p(\lambda)$. The extractor $\mathcal{E}$ on input 1^λ, R, and x works as follows.

- **Estimate advantage.** The extractor first estimates

$$\varepsilon_0 := \Pr[\mathcal{A}(1^\lambda, \mathsf{Enc}(1^\lambda, R, x, n, 0)) = 1]$$
$$\varepsilon_1 := \Pr[\mathcal{A}(1^\lambda, \mathsf{Enc}(1^\lambda, R, x, n, 1)) = 1]$$

  by running $\mathcal{A}$ with the fresh ciphertexts for $O(\lambda) \cdot p(\lambda)$ times. With overwhelming probability, the estimation is accurate up to an additive error of $1/(8p(\lambda))$.

- **Extract witness length.** For each $m_0 \leq m(|x|)$, define the machine M'_{R,m_0} which on input w outputs 1 if $|w| = m_0$ and $R(x, w)$ outputs 1, and outputs 0 otherwise. Estimate

$$\varepsilon_{m_0} := \Pr[\mathcal{A}(1^\lambda, \mathsf{iO}(1^\lambda, M'_{R,m_0}[x], 1^{m_0}))]$$

  If the estimation is closer to ε_1 than ε_0, set $m_x = m_0$ and proceed to the next step.

- **Extract witness per bit.** For each $i \in [m_x]$, define the machine $M''_{R,i}$ which on input w outputs 1 if $|w| = m_x$, $R(x, w)$ outputs 1, and the i-th bit of w is 1, and outputs 0 otherwise. Estimate

$$\varepsilon_{m_x,i} := \Pr[\mathcal{A}(1^\lambda, \mathsf{iO}(1^\lambda, M''_{R,i}[x], 1^{m_x}))].$$

  If the estimation is closer to ε_1 than ε_0, set the i-th bit of the output witness to be 1, and otherwise set it to be 0.

- Finally, output the extracted witness w.

We claim that the extractor $\mathcal{E}$ succeeds in extracting a valid witness with probability at least $1 - \mathsf{negl}(\lambda)$. To see this, we need to argue that in each of the extraction steps, the extractor makes the correct guess with overwhelming probability. For the witness length extraction step, if $m_0 \neq m_x$, then again by combining the EF proof of unambiguity and the EF proof of $R(x, w_x) = 1$, we have that $M'_{R,m_0}[x]$ and $M_{R,n,0}[x]$ are $(\rho_0(|x|), \tau_0(|x|, n))$-EF equivalent, and thus the iO security guarantees that with overwhelming probability, the estimation ε_{m_0} is close to ε_0. On the other hand, if $m_0 = m_x$, then $M'_{R,m_0}[x]$ and $M_{R,n,1}[x]$ are $(\rho_0(|x|), \tau_0(|x|, n))$-EF equivalent following the identical argument, and thus the estimation ε_{m_0} is close to ε_1 with overwhelming probability. For the witness bit extraction step, if the i-th bit of w_x is 0, then by combining the EF proof of unambiguity and the EF proof of $R(x, w_x) = 1$, we have that $M''_{R,i}[x]$ is a null function, and that $M''_{R,i}[x]$ and $M_{R,n,0}[x]$ are $(\rho_0(|x|), \tau_0(|x|, n))$-EF equivalent, therefore the estimation $\varepsilon_{m_x,i}$ is close to ε_0 with overwhelming probability. On the other hand, if the i-th bit of w_x is 1, then $M''_{R,i}[x]$ and $M_{R,n,1}[x]$ are $(\rho_0(|x|), \tau_0(|x|, n))$-EF equivalent, and thus the estimation $\varepsilon_{m_x,i}$ is close to ε_1 with overwhelming probability. Therefore, by a union bound over all the extraction steps, we have that the extractor $\mathcal{E}$ succeeds in extracting a valid witness with probability at least $1 - \mathsf{negl}(\lambda)$. □

Remark 11 (On the padding size of M)**.** In the proof of Theorem 5.11, we rely on iO security for intermediate machines $M'_{R,m_0}[x]$ and $M''_{R,i}[x]$. We stress that the size of the private descriptions of these machines are succinct: they additionally hardcode indices m_0 and i which has description size at most λ. Importantly, this means that the size padding for the construction is λ, independent of the size of $|x|$. This is crucial for the full-succinctness of the WE construction.

Remark 12 (Statement-wise security)**.** Currently, the security of the WE construction is defined for relations where *every* (false) statements has an appropriate EF proof. On the other hand, it is easy to observe that the proofs of Theorems 5.10 and 5.11 only relies on the existence of an appropriate EF proof for the specific statement x used in the encryption. Therefore, one can strengthen the results to support all NP relations, and achieves instance-wise security of the from:

- Instance-wise security, where the security guarantee only holds for false statements x that has an appropriate EF proof of non-membership.
- Instance-wise extractability, where the extractability guarantee only holds for true statements x that has an appropriate EF proof of unambiguity and a m-bounded witness.

Finally, by instantiating Construction 1 with the rate-1 EF-iO construction from Theorem 5.2, we immediately get a succinct witness encryption scheme for the corresponding classes of relations.

Theorem 5.12 (Succinct witness encryption). *Let $\rho(\lambda, \cdot)$ be a function and $\tau(\lambda, \cdot)$, $m(\cdot)$ be polynomials. Assuming*

- *superpolynomially secure PCG with $\ell(n) = n^{o(1)}$ stretch (Assumption 4),*
- *superpolynomially secure fully succinct EF-iO,*
- *superpolynomially secure pseudorandom generators,*

there exists

- *a succinct witness encryption scheme for all relations with (ρ, τ)-EF proof of non-membership, and*
- *a succinct extractable witness encryption scheme for all total relations with m-bounded witnesses and (ρ, τ)-EF proof of unambiguity,*

both with ciphertexts of size $\mathsf{poly}(\lambda)$.

6 Succinct Multi-Input Functional Encryption

In this section, we present the following result on succinct multi-input functional encryption schemes.

Theorem 6.1. *Assuming*

- *polynomially secure PCG with $\ell(n) = n^{o(1)}$ compression rate (Assumption 3),*
- *polynomial hardness of circular LWE*
- *polynomial hardness of fully-succinct EF-iO,*

there exist a very-selectively secure succinct bounded-query EF-MIFE scheme with $\mathsf{poly}(\lambda, L)$-size encryption keys and ciphertexts, and $|f| + \mathsf{poly}(\lambda, L)$-size decryption keys, where $|f|$ is the size of the function description, and L is the per-index input length.

To achieve the result, we build a succinct EF-MIFE scheme assuming t-time ℓ-compression for suitable parameters, and then apply Theorem 4.10 with large compression rate (Assumption 3) to upgrade the scheme to full security.

6.1 Relation-succinct NIWIPoK with PV proofs.

Before describing the construction, we first start with one crucial building block for our EF-MIFE construction, which is a relation-succinct NIWIPoK (Definition 3.10) for general relations with PV proofs of completeness and extractability. Such NIWIPoK is directly implied by existing results, but we need to carefully examine the existing constructions to confirm that the completeness and extractability properties can be proved in PV.

Our starting point is the NIZK[27] from LWE construction from [PS19], which shows that assuming LWE, there exists a statistically sound NIZK for Hamiltonian cycle. The core component in their construction is a statistically correlation intractable hash (CIH) function from LWE. It was shown in [DHJQ26] that the correlation intractability property of the construction admits a PV proof. Toward a NIZK, [PS19] follows the same template of [CCH+19], which combines CIH with a (lossy) PKE. The prover with a graph G and a cycle $C \subseteq G$ computes the proof via the following three steps:

1. Compute λ ciphertexts $\mathsf{ct}_1, \ldots, \mathsf{ct}_\lambda$, each bit-by-bit encrypts the graph $H_i = \pi_i(C)$ under some public key pk in crs, where π_i is a random permutation of the vertex set for each $i \in [\lambda]$.
2. Hash the ciphertexts $\{\mathsf{ct}_i\}_i$ with the CIH hash to obtain a λ-bit challenge $e_1, \ldots, e_\lambda$.
3. If $e_i = 0$, provide the full encryption randomness for ct_i; if $e_i = 1$, reveal π_i and the partial randomness for ct_i that encrypts the non-edges of $\pi_i(G)$. During verification, the verifier checks if ct_i is a valid encryption of a cycle when $e_i = 0$, and checks if all revealed non-edges are indeed non-edges when $e_i = 1$.

The completeness of the scheme follows straight from definition, and is therefore provable in PV. The soundness of the scheme relies on the correlation intractability of the hash function. In particular, the hash is set to be correlation intractable for the function which decrypts each ct_i, checks if ct_i is a cycle, specifies $e_i = 1$ if it is not, and $e_i = 0$ otherwise. Such correlation intractability properties guarantee that every accepting proof must have some index i where ct_i is not a cycle and the full encryption randomness is revealed, or where ct_i is a cycle and π_i is revealed. The correctness of the encryption scheme guarantees that the former case cannot give a valid proof. This gives a natural extractor that extracts the cycle by decrypting the ct_i for H_i where π_i is revealed, and recovers the cycle $\pi_i^{-1}(H_i)$. When instantiated with the CIH with PV proof of correlation intractability from [PS19] and any public-key encryption scheme with PV proof of correctness (Lemma 3.7), the above construction gives a NIWIPoK for Hamiltonian cycle with PV proofs of completeness and extractability.

Lemma 6.2. *Assuming LWE, there exists a perfectly sound NIWIPoK for Hamiltonian cycle, where the completeness and extractability admit a PV proof.*

To obtain a NIWIPoK for general relation, we simply follow the Karp reduction from Circuit SAT to Hamiltonian Cycle, where the prover transforms the relation $R(x, \cdot)$ and its witness w into a graph with a Hamiltonian cycle. Such reduction can be formalized and proven in PV. As shown in Section B.1, there exists PV functions $f, \mathcal{R}_s, \mathcal{R}_c$ transforming a circuit C into a graph $G = f(C)$, and $\mathcal{R}_s$ and $\mathcal{R}_c$ allows transforming between satisfying assignments of C and Hamiltonian cycle in G. The PV proof of completeness of the transformed system follows from the completeness of $f, \mathcal{R}_c$, and the PV proof of extractability follows from the soundness of $f, \mathcal{R}_s$.

Lemma 6.3. *Assuming LWE, there exists a perfectly sound NIWIPoK for all bounded-sized relations, where the completeness and extractability admit a PV proof.*

Finally, we boost the NIWI to a relation succinct NIWI (Definition 3.10) using FHE (for *unbounded-depth circuits*). The NIWI proof from Lemma 6.3 for relation R transforms a statement-witness pair (x, w) into a graph that has size polynomial in the size of R, resulting in a proof size that grows with $|R|$. To reduce the proof size, we can construct a NIWI proof with the following components:

[27] While it seems like an overkill to build NIZK from NIWI, using only the weaker WI guarantee largely simplifies the formulation of the security proof in bounded arithmetic (see Section 10). In particular, WI can be easily formulated as indistinguishability between simple distributions, while ZK requires a simulation-based definition.

- An FHE public key pk, and ciphertext ct_w encrypting the witness w under pk.
- A homomorphic ciphertext $\mathsf{ct}_R = \mathsf{FHE.Eval}(R(x,\cdot), \mathsf{ct}_w)$, which encrypts $R(x,w)$.
- A NIWI proof $\pi_{\mathsf{pk},\mathsf{ct}_w,\mathsf{ct}_R}$ generated with witness $(\mathsf{sk}, w, r_w, r_R)$ for the relation R' checking that
  - sk is a valid secret key corresponding to pk;
  - ct_w is a valid encryption of w under pk and the randomness r_w;
  - ct_R is a valid encryption of 1 under pk and the randomness r_R.

The verifier checks the homomorphic evaluation ct_R by recomputing the evaluation, and verifies the NIWI proof π. The extractor simply decrypts ct_w. This construction is valid if the prover can generate a succinct randomness r_R for the homomorphic evaluation, which can be achieved by the circular GSW scheme. With such instantiation, the NIWI is extractable assuming the evaluation correctness of circular GSW, which can be proved in PV (lemma 3.7). Moreover, since the size of the ciphertexts is independent of the circuit size *and depth*, the relation R' has size dependent only on the witness size.

Lemma 6.4. *Assuming circular LWE, there exists a perfectly sound, relation succinct NIWIPoK for all bounded-sized relations, where the completeness and extractability admit a PV proof.*

6.2 Construction of succinct EF-MIFE

In this section, we construct a bounded-query EF-MIFE scheme with size succinctness in the arity n assuming t-time ℓ-compression, where the public parameters, secret keys, and ciphertext sizes are $\mathsf{poly}(\lambda, \ell)$. As argued earlier, such a construction together with the master theorem (Theorem 4.10) implies EF-MIFE without the compressibility condition (Theorem 6.1).

Construction 2 (EF-MIFE assuming t-time ℓ-compression)**.** We first define the following building blocks.

- Let $t = t(|\mathsf{ch}| + |\pi|)$, $\ell = \ell(|\mathsf{ch}| + |\pi|)$ be the compressibility parameters dependent on the Setup parameters (n, L, Q, p). Let $\mathsf{U}_{\mathsf{TM}}(\cdot, 1^t)$ be the *circuit description of* the universal Turing machine that defines $\boldsymbol{K}^t$. $\mathsf{U}_{\mathsf{TM}}(\mathsf{seed}, 1^t)$ parses $\mathsf{seed} \in \{0,1\}^\ell$ as a description of a Turing machine and mimics a time t evaluation over the machine to obtain (ch, π). For simplicity, we assume that $\mathsf{U}_{\mathsf{TM}}(0^\ell, 1^t)$ always outputs an empty challenge.[28]
- Let $\mathsf{NIWI} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Ext})$ be a NIWIPoK scheme which is PV-provably statistically sound and relation-succinct (Lemma 6.4).
- Let $\mathsf{COM} = (\mathsf{Gen}, \mathsf{Com}, \mathsf{Ext})$ be an commitment scheme which is PV-provably extractable (Lemma 3.5).
- Let $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme, which is PV-provably correct (Lemma 3.7), with ciphertexts of length $L \cdot \ell_{\mathsf{ct}}(\lambda)$ for messages of length L.
- Let iO be a (ρ, τ)-EF-iO for programs with split description as defined in Definition 3.17, where $\rho = \ell \cdot \mathsf{poly}(\lambda)$ and $\tau = \mathsf{poly}(\lambda, t)$.

We now describe the construction of the EF-MIFE scheme.

[28] Such a requirement can also be achieved by extending the universal machine with an additional controlling bit. We adopt the simpler version here for ease of presentation.

- $\mathsf{Setup}(1^\lambda, 1^n, 1^L, 1^Q, 1^p)$: The algorithm first computes two NIWI crs $\mathsf{crs}_{\mathsf{ek}} \leftarrow \mathsf{NIWI.Gen}(1^\lambda, R_{\mathsf{ek}})$, $\mathsf{crs}_{\mathsf{ct}} \leftarrow \mathsf{NIWI.Gen}(1^\lambda, R_{\mathsf{ct}})$, where the key relation R_{ek} is defined in Figure 2, and the ciphertext relation R_{ct} is defined in Figure 3. It also samples two PKE key pairs $(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$ and $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$. The algorithm then samples a commitment setup $\mathsf{crs}_{\mathsf{com}} \leftarrow \mathsf{COM.Gen}(1^\lambda)$ for the extractable commitment scheme, and computes the following commitments:
  - A key-mode commitment $Z_{\mathsf{ek}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 1; \sigma_{\mathsf{ek}})$ with opening σ_{ek} committing a control bit $b \in \{0,1\}$ controlling the generation of ek_i. The bit is initially set to 1, denoting the "real" mode where ek_i for every arity index can be generated.
  - A challenge commitment $Z_{\mathsf{ch}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 0^\ell; \sigma_{\mathsf{ch}})$ with opening σ_{ch} committing a seed $\mathsf{seed} \in \{0,1\}^\ell$ that compresses the challenge ch. It is initially set to a dummy commitment of 0^ℓ, where $\mathsf{U}_{\mathsf{TM}}(0^\ell, 1^t)$ is an empty challenge.
  - A trapdoor commitment $Z_{\mathsf{td}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, (0, 0^{L \cdot \ell_{\mathsf{ct}}(\lambda)}, 0^{L \cdot \ell_{\mathsf{ct}}(\lambda)}); \sigma_{\mathsf{td}})$ with opening σ_{td} committing a trapdoor ciphertext tuple (i, c_0, c_1) for some arity index $i \in [n]$, a ciphertext c_0 under pk_0 and a ciphertext c_1 under pk_1, initially set to a dummy commitment of $(0, 0^{L \cdot \ell_{\mathsf{ct}}(\lambda)}, 0^{L \cdot \ell_{\mathsf{ct}}(\lambda)})$.

  Then, for each arity index $i \in [n]$, using the real witness $(\sigma_{\mathsf{ek}}, \perp)$ of R_{ek}, it computes NIWI proof $\pi_{\mathsf{ek},i} \leftarrow \mathsf{Prove}(\mathsf{crs}_{\mathsf{ek}}, (i, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}), (\sigma_{\mathsf{ek}}, \perp))$. It outputs the encryption keys

$$\mathsf{ek}_i = (\mathsf{param} = (n, L, Q, p), i, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}}, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \pi_{\mathsf{ek},i})$$

  for each $i \in [n]$, and the master secret key

$$\mathsf{msk} = (\mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \mathsf{pk}_0, \mathsf{sk}_0, \mathsf{pk}_1, \mathsf{sk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}})$$

Relation R_{ek}:

- **Statement:** $y_{\mathsf{ek}} = (i, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}})$, where $i \in [n]$ is an arity index, $\mathsf{crs}_{\mathsf{com}}$ is a crs for the extractable commitment scheme, Z_{ek} is a commitment of a bit $b \in \{0,1\}$, and Z_{ch} is a commitment of length ℓ string.
- **Witness:** $w_{\mathsf{ek}} = (\sigma_{\mathsf{ek}}, \sigma_{\mathsf{ch}})$ consists of openings $\sigma_{\mathsf{ek}}, \sigma_{\mathsf{ch}}$ for commitments Z_{ek} and Z_{ch} respectively.
- **Relation:** The relation $R_{\mathsf{ek}}(y_{\mathsf{ek}}, w_{\mathsf{ek}})$ holds if one of the following holds:
  1. $\mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 1; \sigma_{\mathsf{ek}}) = Z_{\mathsf{ek}}$, i.e., the commitment Z_{ek} opens to $b = 1$.
  2. $\mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, \mathsf{seed}; \sigma_{\mathsf{ch}}) = Z_{\mathsf{ch}}$ for some seed $\mathsf{seed} \in \{0,1\}^\ell$ such that $\mathsf{U}_{\mathsf{TM}}(\mathsf{seed}, 1^t) = \{\{f_q\}_{q \in [Q]}, \mathcal{I}, \{\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)}\}_{i \in [n] \setminus \mathcal{I}, q \in [Q]}, \pi_{\mathsf{EF}}\}$, and $i \in \mathcal{I}$.

  We say that a witness $w_{\mathsf{ek}} = (\sigma_{\mathsf{ek}}, \perp)$ is a *real witness* when it satisfies the first branch of the relation, and a witness $w_{\mathsf{ek}} = (\perp, \sigma_{\mathsf{ch}})$ is a *constrained witness* when it satisfies the second branch of the relation.

Figure 2: Key Relation R_{ek} certifying the validity of encryption keys.

Relation R_{ct}:

- **Statement:** $y = (i, c_0, c_1, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}})$, where $i \in [n]$ is an arity index, c_0, c_1 are ciphertexts of PKE under public keys $\mathsf{pk}_0, \mathsf{pk}_1$ respectively, $\mathsf{crs}_{\mathsf{ek}}$ is a NIWI crs for relation R_{ek} as defined above, $\mathsf{crs}_{\mathsf{com}}$ is a crs for the extractable commitment scheme, Z_{ek} is a commitment of a bit, Z_{ch} is a commitments of length ℓ strings, and Z_{td} is a commitment of a length $\lambda + 2L\ell_{\mathsf{ct}}(\lambda)$ string, where $\ell_{\mathsf{ct}}(\lambda)$ is the length of the PKE ciphertext of a bit.
- **Witness:** $w = (\mathbf{x}_0, r_0, \mathbf{x}_1, r_1, \pi_{\mathsf{ek}}, \sigma_{\mathsf{ch}}, \sigma_{\mathsf{td}})$ consists of plaintexts $\mathbf{x}_b \in \{0,1\}^L$, encryption randomness r_b for ciphertexts c_b, a NIWI proof π_{ek} for relation R_{ek}, and openings σ_{ch}, σ_{td} for commitments Z_{ch} and Z_{td} respectively.
- **Relation:** The relation $R_{\mathsf{ct}}(y, w)$ holds if one of the following holds:
  1. The ciphertexts c_0, c_1 satisfies $\mathsf{PKE.Enc}(\mathsf{pk}_b, \mathbf{x}_b; r_b) = c_b$ for $b \in \{0,1\}$, $\mathbf{x}_0 = \mathbf{x}_1$, and the NIWI verification $\mathsf{Ver}(\mathsf{crs}_{\mathsf{ek}}, (i, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}), \pi_{\mathsf{ek}}) = 1$.
  2. The ciphertexts c_0, c_1 satisfies $\mathsf{PKE.Enc}(\mathsf{pk}_b, \mathbf{x}_b; r_b) = c_b$ for $b \in \{0,1\}$, and the commitment Z_{ch} opens to some seed seed where $\mathsf{U}_{\mathsf{TM}}(\mathsf{seed}, 1^t) = \{\{f_q\}_{q\in[Q]}, \mathcal{I}, \{\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)}\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}, \pi_{\mathsf{EF}}\}$ such that $i \notin \mathcal{I}$ and there exist $q \in [Q]$ such that $\mathbf{x}_0 = \mathbf{x}_{i,q}^{(0)}$ and $\mathbf{x}_1 = \mathbf{x}_{i,q}^{(1)}$.
  3. The commitment Z_{td} opens to the tuple (i, c_0, c_1).

  We call a witness $w = (\mathbf{x}_0, r_0, \mathbf{x}_1, r_1, \pi_{\mathsf{ek}}, \bot, \bot)$ a *real witness* when it satisfies the first branch of the relation, a witness $w = (\mathbf{x}_0, R_0, \mathbf{x}_1, R_1, \bot, \sigma_{\mathsf{ch}}, \bot)$ a *constrained witness* when it satisfies the second branch of the relation, and a witness $w = (\bot, \bot, \bot, \bot, \bot, \bot, \sigma_{\mathsf{td}})$ a *trapdoor witness* when it satisfies the last branch of the relation.

Figure 3: Ciphertext Relation R_{ct} certifying the validity of ciphertexts.

- $\mathsf{Enc}(\mathsf{ek}_i, \mathbf{x}_i)$: On input the i-th encryption key of the form

$$\mathsf{ek}_i = (\mathsf{param} = (n, L, Q, p), i, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}}, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \pi_{\mathsf{ek},i})$$

  and a message $\mathbf{x}_i \in \{0,1\}^L$, the algorithm samples randomness r_0, r_1 for PKE encryption, computes ciphertexts $c_0 \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_0, \mathbf{x}_i; r_0)$ and $c_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_1, \mathbf{x}_i; r_1)$, and then compute the NIWI proof of relation R_{ct} using the real witness as follows:

$$\pi \leftarrow \mathsf{Prove}\left(\mathsf{crs}_{\mathsf{ct}}, (i, c_0, c_1, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}}), (\mathbf{x}_i, r_0, \mathbf{x}_i, r_1, \pi_{\mathsf{ek},i}, \bot, \bot)\right).$$

  It outputs the ciphertext $\mathsf{ct}_i = (c_0, c_1, \pi)$.

- $\mathsf{KeyGen}(\mathsf{msk}, f)$: On input the master secret key msk and a function $f : (\{0,1\}^L)^n \to \{0,1\}^m$, the algorithm constructs a Turing machine M which hardcodes the secret key sk_0 and takes a split public description $\mathsf{pub} = (f, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}})$, where $M[\mathsf{pub}]$ takes as input n ciphertexts $\mathsf{ct}_i = (c_{i,0}, c_{i,1}, \pi_i)$ for $i \in [n]$, and does the following:
  - Checks that for each $i \in [n]$, the NIWI verification for the ciphertext relation R_{ct} passes, i.e.,

$$\mathsf{Ver}\left(\mathsf{crs}_{\mathsf{ct}}, (i, c_{i,0}, c_{i,1}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}}), \pi_i\right) = 1.$$

    If any check fails, outputs $\bot$.

- Decrypts $\mathbf{x}_{i,0} = \mathsf{PKE.Dec}(\mathsf{sk}_0, c_{i,0})$ for each $i \in [n]$.
- Computes $y = f(\mathbf{x}_{1,0}, \ldots, \mathbf{x}_{n,0})$ and outputs y.

The algorithm computes the obfuscation of $\widetilde{M_f}[\mathsf{pub}] = \mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^{n\cdot|\mathsf{ct}|})$ and output $\widetilde{M_f}[\mathsf{pub}]$.

- $\mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$: On input the functional decryption key $\mathsf{sk}_f = \widetilde{M_f}[\mathsf{pub}]$ and ciphertexts $\mathsf{ct}_i = (c_{i,0}, c_{i,1}, \pi_i)$ for $i \in [n]$, output $\widetilde{M_f}[\mathsf{pub}](\mathsf{ct}_1, \ldots, \mathsf{ct}_n)$.

Theorem 6.5. *Construction 2 is correct.*

Proof. The correctness follows immediately from the correctness of NIWI, PKE, and iO. In particular, every honest ciphertext $\mathsf{ct}_i = (c_{i,0}, c_{i,1}, \pi_i)$ encrypting $\mathbf{x}_i$, π_i is a valid proof for R_{ct}, and $\mathbf{x}_i = \mathbf{x}_{i,0} = \mathsf{PKE.Dec}(\mathsf{sk}_0, c_{i,0})$. Therefore the evaluation of the obfuscated program $\widetilde{M_f}[\mathsf{pub}]$ on the ciphertexts will output $f(\mathbf{x}_1, \ldots, \mathbf{x}_n)$. □

Theorem 6.6. *Construction 2 has public parameters, encryption keys, and ciphertext sizes of* $\mathsf{poly}(\lambda, \ell, L)$*, and decryption key size of* $|f| + \mathsf{poly}(\lambda, \ell, L)$.

Proof. Note that while the sizes of the relation circuits R_{ek} and R_{ct} are as large as $t = \mathsf{poly}(n, L, Q, p)$, the witness sizes of both relations are small. In particular, the arity parameter n does not show up in the witness size. First the witness for R_{ek} is of size $|\sigma_{\mathsf{ek}}| + |\sigma_{\mathsf{ch}}| = \ell \cdot \mathsf{poly}(\lambda)$, thus by the relation-succinctness of NIWI, the size of $\mathsf{crs}_{\mathsf{ek}}$ and π_{ek} are $\mathsf{poly}(\lambda, \ell)$. Therefore, the witness for R_{ct} is of size $2L + |\pi_{\mathsf{ek}}| + |\sigma_{\mathsf{ch}}| + |\sigma_{\mathsf{td}}| = \mathsf{poly}(\lambda, \ell, L)$, and by the relation-succinctness of NIWI, the size of $\mathsf{crs}_{\mathsf{ct}}$ and π_{ct} are $\mathsf{poly}(\lambda, \ell, L)$. This immediately bounds the size of public parameters, encryption keys, and ciphertexts to be $\mathsf{poly}(\lambda, \ell, L)$. Finally, for decryption key, the size of the public description pub is $|f| + \mathsf{poly}(\lambda, \ell, L)$, and the machine M hardcoding sk_0 has description size of $\mathsf{poly}(\lambda)$, thus by the fully-succinctness of iO, the size of the obfuscation $\widetilde{M_f}$ is $\mathsf{poly}(\lambda)$, and the total size of the decryption key is $|f| + \mathsf{poly}(\lambda, \ell, L)$. □

Theorem 6.7. *Assuming* NIWI *is PV-provably extractable and computationally WI,* PKE *is semantically secure and admits a* PV *proof of correctness,* Com *is PV-provably extractable and computationally hiding, and* iO *satisfies* (ρ, τ)*-*EF*-security, the* EFMIFE *scheme in Construction 2 is very-selective IND secure assuming* t*-time* ℓ*-compression.*

Proof. To argue the security assuming t-time ℓ-compression, it suffice to argue that, for every polynomial parameters (n, L, Q, p) and every admissible challenge $\mathsf{ch} = \{\{f_q\}_{q\in[Q]}, \mathcal{I}, \{\mathbf{x}^{(0)}_{i,q}, \mathbf{x}^{(1)}_{i,q}\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}\}$ with an EF-proof of admissibility π_{EF}, which further satisfies the t-time ℓ-compressibility condition $\boldsymbol{K}^t(\mathsf{ch}\|\pi_{\mathsf{EF}}) \leq \ell(|\mathsf{ch}| + |\pi_{\mathsf{EF}}|)$, the adversary's view in the case where the challenge bit $b = 0$ is indistinguishable from the case where $b = 1$.

We argue the indistinguishability via the following sequence of hybrids.

$\mathcal{H}_0$: This is the real view of the adversary when the challenge bit is 0. We lay out all components of the experiment explicitly with respect to Construction 2. In particular, this hybrid samples its output following how the challenger samples its message to the adversary, described as follows.

- **Encryption keys.** The challenger samples two NIWI crs $\mathsf{crs}_{\mathsf{ek}} \leftarrow \mathsf{NIWI.Gen}(1^\lambda, R_{\mathsf{ek}})$, $\mathsf{crs}_{\mathsf{ct}} \leftarrow \mathsf{NIWI.Gen}(1^\lambda, R_{\mathsf{ct}})$, commitment setup $\mathsf{crs}_{\mathsf{com}} \leftarrow \mathsf{COM.Gen}(1^\lambda)$, and two PKE key pairs $(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$ and $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$. It also compute commitments $Z_{\mathsf{ek}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 1; \sigma_{\mathsf{ek}})$, $Z_{\mathsf{ch}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 0^\ell; \sigma_{\mathsf{ch}})$ and $Z_{\mathsf{td}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, (0, 0^{L\cdot\ell_{\mathsf{ct}}(\lambda)}, 0^{L\cdot\ell_{\mathsf{ct}}(\lambda)}); \sigma_{\mathsf{td}})$.

Then, for each arity index $i \in [n]$, it computes NIWI proof $\pi_{\mathsf{ek},i} \leftarrow \mathsf{Prove}(\mathsf{crs}_{\mathsf{ek}}, (i, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}), (\sigma_{\mathsf{ek}}, \bot))$ using the real witness of R_{ek}. For each $i \in [n]$, it sets

$$\mathsf{ek}_i = (\mathsf{param} = (n, L, Q, p), i, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}}, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \pi_{\mathsf{ek},i}) .$$

- **Decryption keys.** For each function query f_q, the challenger computes $\mathsf{sk}_{f_q} = \widetilde{M_{f_q}}[\mathsf{pub}] = \mathsf{iO}(M_{f_q}[\mathsf{pub}])$ where $M_{f_q}[\mathsf{pub}]$ is the Turing machine defined in KeyGen of Construction 2 for function f_q.
- **Ciphertexts.** For each uncorrupted index $i \in [n] \setminus \mathcal{I}$ and each index $q \in [Q]$, the challenger computes ciphertexts encrypting $\mathbf{x}^{(0)}_{i,q}$ by

$$\mathsf{ct}_{i,q} = \left(c_{i,q,0} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_0, \mathbf{x}^{(0)}_{i,q}; r_0), c_{i,q,1} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_1, \mathbf{x}^{(0)}_{i,q}; r_1), \pi_{i,q}\right)$$

where the NIWI proof $\pi_{i,q}$ is generated using the real witness for relation R_{ct} as

$$\pi_{i,q} \leftarrow \mathsf{Prove}\left(\mathsf{crs}_{\mathsf{ct}}, \begin{pmatrix} i, c_{i,q,0}, c_{i,q,1}, \mathsf{pk}_0, \mathsf{pk}_1, \\ \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}} \end{pmatrix}, (\mathbf{x}^{(0)}_{i,q}, r_0, \mathbf{x}^{(0)}_{i,q}, r_1, \pi_{\mathsf{ek},i}, \bot, \bot)\right) .$$

The output of this hybrid is the following.

$$\{\mathsf{ek}_i\}_{i\in\mathcal{I}}, \{\mathsf{sk}_{f_q}\}_{q\in[Q]}, \{\mathsf{ct}_{i,q}\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}$$

$\mathcal{H}_1$: This is the same as $\mathcal{H}_0$ except that the commitment Z_{ch} is changed to committing seed seed.

Observe that among all the NIWI proofs in the experiment, the R_{ek}-proofs $\pi_{\mathsf{ek},i}$ are generated using the real witness $(\sigma_{\mathsf{ek}}, \bot)$, and the R_{ct}-proofs $\pi_{i,q}$ are generated using the real witness $(\mathbf{x}^{(0)}_{i,q}, r_0, \mathbf{x}^{(0)}_{i,q}, r_1, \pi_{\mathsf{ek},i}, \bot, \bot)$. In both cases, the opening σ_{ch} of the commitment Z_{ch} is not used. Therefore, $Z_{\mathsf{ch}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 0^\ell)$ and $Z_{\mathsf{ch}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, \mathsf{seed})$ are computationally indistinguishable by the hiding property of Com. This gives the following lemma.

Lemma 6.8. *Assuming* Com *is computationally hiding, for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_0(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_1(1^\lambda, \mathcal{A}) = 1]\right| \le \mathsf{negl}(\lambda).$$

$\mathcal{H}_2$: We now consider a series of hybrids, which runs through all the challenge indices $i \in [n]\setminus\mathcal{I}$ and $q \in [Q]$. All these hybrids differs with $\mathcal{H}_1$ only on how the challenge ciphertexts $\mathsf{ct}_{i,q} = (c_{i,q,0}, c_{i,q,1}, \pi_{i,q})$ are computed.

We order all these pairs (i, q) in some canonical order, and consider the following intermediate hybrids.

$\mathcal{H}_{2,(i,q),0}$: For all pairs $(i', q') \ge (i, q)$, $c_{i',q',0}, c_{i',q',1}$ are computed as encryption of $(\mathbf{x}^{(0)}_{i',q'}, \mathbf{x}^{(0)}_{i',q'})$ as in hybrid 1, and $\pi_{i',q'}$ are generated using the real witness. For all pairs $(i', q') < (i, q)$, $c_{i',q',0}, c_{i',q',1}$ are computed as encryption of $(\mathbf{x}^{(0)}_{i',q'}, \mathbf{x}^{(1)}_{i',q'})$, and $\pi_{i',q'}$ are generated using the constrained witness

$$\pi_{i',q'} \leftarrow \mathsf{Prove}\left(\begin{matrix}\mathsf{crs}_{\mathsf{ct}}, (i', c_{i',q',0}, c_{i',q',1}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}}), \\ (\mathbf{x}^{(0)}_{i',q'}, r_0, \mathbf{x}^{(1)}_{i',q'}, r_1, \bot, \sigma_{\mathsf{ch}}, \bot)\end{matrix}\right) .$$

By definition, $\mathcal{H}_1$ is exactly $\mathcal{H}_{2,(i',q'),0}$ where (i', q') is the first pair in the canonical order.

$\mathcal{H}_{2,(i,q),1}$: Same as hybrid $\mathcal{H}_{2,(i,q),0}$, except that the ciphertext pair on index (i,q)

$$\left(c_{i,q,0} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_0, \mathbf{x}_{i,q}^{(0)}), c_{i,q,1} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_1, \mathbf{x}_{i,q}^{(0)})\right)$$

is generated along with the encryption key elements, right after the setups. The trapdoor commitment Z_{td} is changed to commit to $(i, c_{i,q,0}, c_{i,q,1})$.

Note that in $\mathcal{H}_{2,(i,q),0}$ and $\mathcal{H}_{2,(i,q),1}$, all NIWI proofs of R_{ct} are generated either using the real witness (for the case $(i',q') \geq (i,q)$) or the constrained witness (for the case $(i',q') < (i,q)$). Neither witness uses the opening σ_{td} of the commitment Z_{td}. Therefore, $Z_{\mathsf{td}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, (0, 0^{L\cdot\ell_{\mathsf{ct}}(\lambda)}, 0^{L\cdot\ell_{\mathsf{ct}}(\lambda)}))$ and $Z_{\mathsf{td}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, (i, c_{i,q,0}, c_{i,q,1}))$ are computationally indistinguishable by the hiding property of Com. This gives the following lemma.

Lemma 6.9. *Assuming* Com *is computationally hiding, and for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that for every pair* (i,q)*,*

$$\left|\Pr[\mathcal{H}_{2,(i,q),0}(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_{2,(i,q),1}(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_{2,(i,q),2}$: Same as hybrid $\mathcal{H}_{2,(i,q),1}$, except that the proof $\pi_{i,q}$ is generated using the trapdoor witness, i.e.,

$$\pi_{i,q} \leftarrow \mathsf{Prove}\left(\begin{matrix}\mathsf{crs}_{\mathsf{ct}}, (i, c_{i,q,0}, c_{i,q,1}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}}),\\ (\bot, \bot, \bot, \bot, \bot, \bot, \sigma_{\mathsf{td}})\end{matrix}\right).$$

Note that the encryption randomness r_0, r_1 are not used in the trapdoor witness.

The indistinguishability between $\mathcal{H}_{2,(i,q),1}$ and $\mathcal{H}_{2,(i,q),2}$ follows immediately from the witness indistinguishability of the NIWI scheme.

Lemma 6.10. *Assuming* NIWI *is computationally WI, for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that for every pair* (i,q)*,*

$$\left|\Pr[\mathcal{H}_{2,(i,q),1}(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_{2,(i,q),2}(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_{2,(i,q),3}$: Same as hybrid $\mathcal{H}_{2,(i,q),2}$, except that the ciphertexts $c_{i,q,0}, c_{i,q,1}$ are changed to encryptions of $(\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)})$ (instead of $(\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(0)})$ in $\mathcal{H}_{2,(i,q),2}$)

The indistinguishability between $\mathcal{H}_{2,(i,q),2}$ and $\mathcal{H}_{2,(i,q),3}$ follows from the semantic security of PKE, as the encryption randomness r_0, r_1 are not used in the adversary's view.

Lemma 6.11. *Assuming* PKE *is semantically secure, for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that for every pair* (i,q)*,*

$$\left|\Pr[\mathcal{H}_{2,(i,q),2}(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_{2,(i,q),3}(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_{2,(i,q),4}$: Same as hybrid $\mathcal{H}_{2,(i,q),3}$, except that the proof $\pi_{i,q}$ is generated using the constrained witness, i.e.,

$$\pi_{i,q} \leftarrow \mathsf{Prove}\left(\begin{matrix}\mathsf{crs}_{\mathsf{ct}}, (i, c_{i,q,0}, c_{i,q,1}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}}),\\ (\mathbf{x}_{i,q}^{(0)}, r_0, \mathbf{x}_{i,q}^{(1)}, r_1, \bot, \sigma_{\mathsf{ch}}, \bot)\end{matrix}\right).$$

To see that this constrained witness is indeed a witness for the statement, we note that the commitment Z_{ch} opens to a seed seed such that $\mathsf{U}_{\mathsf{TM}}(\mathsf{seed}, 1^t) = (\mathsf{ch}, \pi_{\mathsf{EF}})$, and that from $\mathcal{H}_{2,(i,q),3}$, $(c_{i,q,0} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_0, \mathbf{x}_{i,q}^{(0)}), c_{i,q,1} \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_1, \mathbf{x}_{i,q}^{(1)}))$ encrypts a pair of messages $(\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)})$ that are part of the challenge ch. With this, the indistinguishability between $\mathcal{H}_{2,(i,q),3}$ and $\mathcal{H}_{2,(i,q),4}$ again follows from the witness indistinguishability of the NIWI scheme.

Lemma 6.12. *Assuming* NIWI *is computationally WI, for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that for every pair* (i, q)*,*

$$\left|\Pr[\mathcal{H}_{2,(i,q),3}(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_{2,(i,q),4}(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_{2,(i,q),5}$: Same as hybrid $\mathcal{H}_{2,(i,q),4}$, except that the commitment Z_{td} is changed back to commit to dummy ciphertext $(0, 0^{L \cdot \ell_{\mathsf{ct}}(\lambda)}, 0^{L \cdot \ell_{\mathsf{ct}}(\lambda)})$. This is exactly hybrid $\mathcal{H}_{2,(i',q'),0}$ where (i', q') is the next pair after (i, q) in the canonical order.

The indistinguishability between $\mathcal{H}_{2,(i,q),4}$ and $\mathcal{H}_{2,(i,q),5}$ follows again from the hiding property of the commitment scheme, as the trapdoor witness is no longer used, and thus the opening σ_{td} of the commitment Z_{td} is not used in any NIWI proof.

Lemma 6.13. *Assuming* Com *is computationally hiding, for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that for every pair* (i, q)*,*

$$\left|\Pr[\mathcal{H}_{2,(i,q),4}(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_{2,(i,q),5}(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_3$: This is hybrid $\mathcal{H}_{2,(i',q'),5}$ where (i', q') is the last pair in the canonical order. In this hybrid, all challenge ciphertexts $\mathsf{ct}_{i,q} = (c_{i,q,0}, c_{i,q,1}, \pi_{i,q})$ are computed as encryption of $(\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)})$, and the NIWI proof $\pi_{i,q}$ is generated using the constrained witness. In particular, the hybrid no longer needs the real-witness component $\pi_{\mathsf{ek},i}$ for $i \in [n] \setminus \mathcal{I}$ to generate the NIWI proof $\pi_{i,q}$.

$\mathcal{H}_4$: Same as $\mathcal{H}_3$, except that we no longer generate the R_{ek} NIWI proof $\pi_{\mathsf{ek},i}$ for $i \notin \mathcal{I}$. As argued in $\mathcal{H}_3$, all ciphertexts within the view do not require these proofs. Furthermore, for the corrupted index $i \in \mathcal{I}$, where ek_i has to be provided to the adversary, the hybrid generates the proofs $\pi_{\mathsf{ek},i}$ using the constrained witness instead of the real witness, i.e.,

$$\pi_{\mathsf{ek},i} \leftarrow \mathsf{Prove}(\mathsf{crs}_{\mathsf{ek}}, (i, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}), (\bot, \sigma_{\mathsf{ch}})).$$

To see that this constrained witness is indeed a witness for the statement, we note that the commitment Z_{ch} opens to a seed seed such that $\mathsf{U}_{\mathsf{TM}}(\mathsf{seed}, 1^t) = \{\{f_q\}_{q\in[Q]}, \mathcal{I}, \{\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)}\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}, \pi_{\mathsf{EF}}\}$, and that by definition $i \in \mathcal{I}$. With this, the indistinguishability between $\mathcal{H}_3$ and $\mathcal{H}_4$ again follows from the witness indistinguishability of the NIWI scheme.

Lemma 6.14. *Assuming* NIWI *is computationally WI, for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_3(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_4(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_5$: Same as $\mathcal{H}_4$, except that the key-mode commitment Z_{ek} commits to 0 instead of 1. Note that in this hybrid, for every uncorrupted index $i \in [n] \setminus \mathcal{I}$, there exists no valid encryption key ek_i, i.e., the statement $(i, Z_{\mathsf{ek}}, Z_{\mathsf{ch}})$ is not in the language of R_{ek}.

The indistinguishability between $\mathcal{H}_4$ and $\mathcal{H}_5$ follows from the hiding property of the commitment scheme, as the real witness containing σ_{ek} is no longer used for generating R_{ek} NIWI proofs.

Lemma 6.15. *Assuming* Com *is computationally hiding, for every efficient adversary $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_4(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_5(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_6$: Same as $\mathcal{H}_5$, except that the decryption key $\mathsf{sk}_{f_q} = \widetilde{M}_{f_q} = \mathsf{iO}(M[\mathsf{pub}])$ is changed to $\mathsf{sk}_{f_q} = \widetilde{M}'_{f_q} = \mathsf{iO}(M'[\mathsf{pub}])$ where M' is the Turing machine with split description defined in KeyGen of Construction 2 for function f_q except that it hardcodes sk_1 instead of sk_0, and decrypts $\mathbf{x}_{i,1} = \mathsf{PKE.Dec}(\mathsf{sk}_1, c_{i,1})$ for each $i \in [n]$ instead of $\mathbf{x}_{i,0} = \mathsf{PKE.Dec}(\mathsf{sk}_0, c_{i,0})$.

We claim that the Turing machines $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$ with identical public description

$$\mathsf{pub} = (f_q, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}})$$

are (ρ, τ)-EF-equivalent. At a high level, since Z_{ek} commits to 0, there is no valid encryption key ek_i for any index $i \in [n] \setminus \mathcal{I}$. Furthermore, Z_{td} does not commit to any valid index. Therefore, any ciphertext $\mathsf{ct}_i = (c_0, c_1, \pi)$ on non-corrupted indices that passes the R_{ct} verification must have a witness in the second branch, i.e., (c_0, c_1) encrypts some messages $(\mathbf{x}_0, \mathbf{x}_1)$ within the challenge set ch compressed as seed. Combined with the EF proof of admissibility of ch, the computation $M[\mathsf{pub}]$ that computes f_q over $\mathbf{x}_0$ is equivalent to the computation $M'[\mathsf{pub}]$ that computes f_q over $\mathbf{x}_1$ for all inputs that pass the verification. By combining the PV proofs of PKE, Com, and NIWI, the given argument transforms to an uniform EF proof of the equivalence between $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$. With this, the indistinguishability between $\mathcal{H}_5$ and $\mathcal{H}_6$ follows from the EF-security of iO.

For the sake of formality, we provide a formal treatment of the proof of equivalence between $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$ in Section B.4.

Lemma 6.16. *Assuming* NIWI *admits a* PV *proof of extractability,* Com *admits a* PV *proof of extractability, and* PKE *admits a* PV *proof of correctness, and assuming* iO *satisfies (ρ, τ)-*EF*-security where $\rho = \ell \cdot \mathsf{poly}(\lambda)$ and τ is a sufficiently large polynomial, for all efficient adversary $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_5(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_6(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_7$: Same as $\mathcal{H}_6$, except that the key-mode commitment Z_{ek} is changed back to commit to 1. The indistinguishability between $\mathcal{H}_6$ and $\mathcal{H}_7$ is identical to the indistinguishability between $\mathcal{H}_4$ and $\mathcal{H}_5$.

Lemma 6.17. *Assuming* Com *is computationally hiding, for every efficient adversary $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_6(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_7(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_8$: Same as $\mathcal{H}_7$, except that in this hybrid, all encryption keys ek_i for $i \in [n]$ are generated, and the R_{ek} NIWI proofs $\pi_{\mathsf{ek},i}$ are generated using the real witness containing σ_{ek}. The indistinguishability between $\mathcal{H}_7$ and $\mathcal{H}_8$ is identical to the indistinguishability between $\mathcal{H}_3$ and $\mathcal{H}_4$.

Lemma 6.18. *Assuming* NIWI *is computationally WI, for every efficient adversary $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_7(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_8(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_9$: Same as $\mathcal{H}_8$, except that in this hybrid, all challenge ciphertexts $\mathsf{ct}_{i,q} = (c_{i,q,0}, c_{i,q,1}, \pi_{i,q})$ are computed as encryption of $(\mathbf{x}^{(1)}_{i,q}, \mathbf{x}^{(1)}_{i,q})$, and the proofs $\pi_{i,q}$ are generated using the real witness containing the R_{ek} NIWI proof $\pi_{\mathsf{ek},i}$. The indistinguishability between $\mathcal{H}_8$ and $\mathcal{H}_9$ follows from the same sequence of hybrids between $\mathcal{H}_1$ and $\mathcal{H}_3$, but in the reverse order.

Lemma 6.19. *Assuming* NIWI *is computationally WI,* PKE *is semantically secure, and* Com *is computationally hiding, for every efficient adversary $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_8(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_9(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_{10}$: In this hybrid, the challenge commitment Z_{ch} is changed back to commit to 0^ℓ. The indistinguishability between $\mathcal{H}_9$ and $\mathcal{H}_{10}$ is identical to the indistinguishability between $\mathcal{H}_0$ and $\mathcal{H}_1$.

Lemma 6.20. *Assuming* Com *is computationally hiding, for every efficient adversary $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_9(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_{10}(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

$\mathcal{H}_{11}$: Same as $\mathcal{H}_{10}$, except that each decryption key $\mathsf{sk}_{f_q} = \mathsf{iO}(M'[\mathsf{pub}])$ is changed back to $\mathsf{sk}_{f_q} = \mathsf{iO}(M[\mathsf{pub}])$. This is the real view of the adversary when the challenge bit is 1.

We claim that the Turing machines $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$ with $\mathsf{pub} = (f_q, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}})$ are (ρ, τ)-EF-equivalent. (Note that the commitment values in pub are *different* from the ones in $\mathcal{H}_5$ and $\mathcal{H}_6$!) In this hybrid, Z_{ch} commits 0^ℓ, which does not correspond to any valid challenge, and Z_{td} does not commit to any valid index. Therefore, any ciphertext $\mathsf{ct}_i = (c_0, c_1, \pi)$ on any index that passes the R_{ct} verification must have a witness in the first branch, i.e., (c_0, c_1) encrypts the same message. With this, the computation $M[\mathsf{pub}]$ that computes f_q over $\mathbf{x}_{i,0}$ decrypted from c_0 is equivalent to the computation $M'[\mathsf{pub}]$ that computes f_q over $\mathbf{x}_{i,1}$ decrypted from c_1 for all inputs that passes the verification. Similar to the argument between $\mathcal{H}_5$ and $\mathcal{H}_6$, by relying on the EF proofs of NIWI soundness, commitment extractability, and encryption correctness, we obtain an EF proof of the equivalence between $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$. With this, the indistinguishability between $\mathcal{H}_{10}$ and $\mathcal{H}_{11}$ follows from the EF-security of iO.

Lemma 6.21. *Assuming* NIWI *admits a* PV *proof of extractability,* Com *admits a* PV *proof of extractability, and* PKE *admits a* PV *proof of correctness, and assuming* iO *satisfies (ρ, τ)-*EF*-security where $\rho = \ell \cdot \mathsf{poly}(\lambda)$ and τ is a sufficiently large polynomial, for all efficient adversary $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\left|\Pr[\mathcal{H}_{10}(1^\lambda, \mathcal{A}) = 1] - \Pr[\mathcal{H}_{11}(1^\lambda, \mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

Combining all the hybrids, we conclude that the adversary's view in the case where the challenge bit $b = 0$ is indistinguishable from the case where $b = 1$ for every admissible challenge ch with an EF-proof of admissibility π_{EF} such that $\boldsymbol{K}^t(\mathsf{ch}\|\pi_{\mathsf{EF}}) \leq \ell(|\mathsf{ch}| + |\pi_{\mathsf{EF}}|)$. This concludes the proof. □

7 Non-Adaptive EF-SNARGs with Short CRS

Consider some NP language $\mathcal{L}$ with relation given by a polynomial time relation $\mathcal{R}$. Suppose that any witness for x has size at most $m(|x|)$, where m is some polynomial. Let $C_x : \{0,1\}^{m(|x|)} \to \{0,1\}$ correspond to the $s(|x|)$-sized circuit evaluating $\mathcal{R}(x, \cdot)$. Let ϕ_x denote the statement that $\forall w \in \{0,1\}^{m(n)}, C_x(w) = 0$. We say that $x \notin \mathcal{L}$ has an L-proof of non-membership if ϕ_x has a length L EF proof.

Definition 7.1 (SNARG with L-EF-soundness)**.** We say that a SNARG scheme SNARG_L satisfies L-EF-soundness if soundness only holds for all $x \notin \mathcal{L}$ which have a $L(|x|)$ proof of non-membership.

In this section, we show the following.

Theorem 7.2. *Let L be a polynomial. Assuming*

- *polynomially secure proof complexity generators with large stretch $\ell(n) = n^{o(1)}$ (Assumption 3),*
- *polynomially secure unleveled fully homomorphic encryption,*
- *polynomially secure batch arguments for NP,*

there exists a SNARG scheme SNARG_L with L-EF-soundness. Moreover, the CRS has size $\mathsf{poly}(\lambda)$ independent of L.

We first recap the main tools from prior work in Section 7.1. Then, we present our construction and analysis in Section 7.2.

7.1 Tools and Definitions from Prior Work

In this section, we will be precise about how our Boolean circuits are expressed. This section is almost verbatim from [JKLV24, Section 3.1].

Circuits. We formally describe such a circuit with s gates via a string $\langle C \rangle$ given by $(g_1, \ldots, g_s)$, where each gate $g_i = (i, j, k, f)$ is a tuple consisting of:

- The output wire name i.
- The two children wires j and k denote the children wires.
- A boolean function $f : \{0,1\} \times \{0,1\} \to \{0,1\}$.

Moreover, the last m gates $g_{s-m+1}, \ldots, g_m$ correspond to the output of the circuit. To evaluate $C(x)$, we define:

- for $g_1, \ldots, g_n$, set $g_i(x) = x_i$.
- for $g_i = (i, j, k, f)$ where $i > n$, iteratively evaluate $g_i(x) = f(g_j(x), g_k(x))$.
- Output $g_{s-m+1}(x), \ldots, g_m(x)$.

Universal circuits. Consider a family $\mathcal{C}_{s,n,m}$ of size s circuits with n-bit inputs and m-bit outputs. We then define the following explicit universal circuit U for $\mathcal{C}_{s,n,m}$ such that $U(x, \langle C \rangle) = C(x)$. We construct U via the following sequence of subcircuits $\{U_i\}_{i\in[s]}$, where we define $U_i : \{0,1\}^{i-1} \times \mathcal{G} \to \{0,1\}$ (where $\mathcal{G}$ denotes the gate description alphabet) as follows: For every $1 \le j < i$, let w_j^{out} denote the output wire of U_j. This will be the jth input to U_i. U_i will additionally take as input the ith gate of $\langle C \rangle$.

- Suppose the input to U_i is $(w_1, \ldots, w_{i-1}, g = (i, j_0, j_1, f))$.
- For $b \in \{0,1\}, \alpha \in [i-1]$, construct a subcircuit $\Lambda_{b,\alpha}$ which takes the same input $(w_\alpha, g = (i, j, k, f))$ and outputs w_α if $\alpha = j_b$ (and $\perp$ otherwise). Denote the output wire of $\Lambda_{b,\alpha}$ by $\omega_{b,\alpha}$ (and $\perp$ otherwise).
- Let Λ_b take as input $(\omega_{b,1}, \ldots, \omega_{b,i-1})$ and outputs the single input which is not equal to $\perp$.
- Construct a subcircuit Λ_{out} which takes as input ω_0, ω_1 and $g = (i, j_0, j_1, f)$, and outputs $f(\omega_0, \omega_1)$.
- The output of U_i corresponds to the output of Λ_{out}.

The output wires correspond to the output wires of $U_{s-m+1}, \ldots, U_s$. It is easy to see that the above universal circuit has size and depth $\mathsf{poly}(s)$. Moreover, each U_i has depth at most $O(\log s)$.

We will also define the following notation for augmenting a circuit.

Definition 7.3 (Augmented circuit)**.** Let C be a circuit with k wires, and let D be a circuit that takes as input k bits, and outputs nothing (one can think of this as a circuit which always outputs 0). Let $\mathsf{Aug}(C, D)$ denote the following circuit: on input x,

- Compute $C(x)$, and let τ be the corresponding wire assignment.
- Evaluate $D(\tau)$.
- Output $C(x)$.

Local assignment generators. In our analysis, we will construct local assignment generators [PR17]. Definition 7.4 are taken nearly verbatim from [JKLV24].

Definition 7.4 (Local assignment generator)**.** Let $C : \{0,1\}^n \to \{0,1\}^m$ denote a Boolean circuit of size s. For parameters ℓ and ε, we say that C has an (ℓ, ε)-local assignment generator if there is an algorithm $\mathsf{LocalGen}$ with the following properties.

- **Syntax:** The input to $\mathsf{LocalGen}$ is a subset of wires $T \subseteq [s]$ of size at most ℓ and its output is an assignment $(\sigma_i)_{i\in T} \in \{0,1\}^T$ to the corresponding wires of C.
- **Local consistency:** for any gate $g_i = (i, j, k, f)$ of C, and any set $T \supseteq \{i, j, k\}$ of size at most ℓ, we have
$$\Pr\left[\ \sigma_i \neq f(\sigma_j, \sigma_k) \ \ : \ \ (\sigma_\alpha)_{\alpha\in T} \leftarrow \mathsf{LocalGen}(T)\ \right] \le \varepsilon.$$
- **Computational non-signaling:** for any sets $T_0, T_1 \subseteq [s]$ of size at most ℓ, the following distributions are ε-computationally indistinguishable:
$$(\{\sigma_i\}_{i\in U} : \{\sigma_i\}_{i\in T_0} \leftarrow \mathsf{LocalGen}(T_0)) \approx_\varepsilon (\{\sigma_i\}_{i\in U} : \{\sigma_i\}_{i\in T_1} \leftarrow \mathsf{LocalGen}(T_1))$$
where $U = T_0 \cap T_1$.

Definition 7.5 (Accepting Local Assignment Generator)**.** We say that an (ℓ, ε)-local assignment generator $\mathsf{LocalGen}$ for circuit C is δ-*almost accepting* if

$$\Pr[\sigma \neq 1 : \sigma \leftarrow \mathsf{LocalGen}(\{\mathsf{out}\})] \leq \delta.$$

where out corresponds to the output wire of C.

We need the following theorem from prior work.

Theorem 7.6 ([HJLM26, Section 11])**.** *Let p, p_1, p_2 be polynomials. Consider some $x \in \{0,1\}^n$ and circuit C of size s such that there exists an* EF *proof ϕ_x of size ℓ proving that $\forall w, C_x(w) = 0$. Then, there exists a circuit E_x of size $p(\ell)$ with the following property: For any $(p_1(\log n, \log \ell), \varepsilon)$-local assignment generator of size $\mathsf{poly}(n, s, \lambda)$ for $\mathsf{Aug}(C_x, E_x)$, it holds that it is accepting with probability at most $\varepsilon \cdot p_2(n, \ell)$. Moreover, E_x can be computed efficiently from ϕ_x.*

Let Expand denote the algorithm in Theorem 7.6 which takes as input ϕ_x and outputs $\langle E_x \rangle$ - the string describing the circuit E_x.

7.2 Construction and analysis

Fix a function $\ell(n) = n^{o(1)}$ and a polynomial t.

Let m, L be polynomials. Consider some NP language $\mathcal{L}$ with relation given by a polynomial time relation $\mathcal{R}$ which has an L proof of non-membership. Suppose that any witness for x has size at most $m(|x|)$. We say that x has a (ℓ, t, L)-EF-proof of non-membership if there exists a machine $\mathcal{M}_\Pi$ of size ℓ which in time t outputs a proof ϕ_x of length L, i.e. $\boldsymbol{K}^t(\phi_x) \leq \ell$.

We now present a construction parameterized by a polynomial L, a function ℓ and a polynomial t. The following construction is almost verbatim taken from [HJLM26] (which itself is variant of [JKLV24]), with minor modifications. These modifications will be highlighted in blue.

Construction 3 (EF-SNARG)**.** Fix an NP language $\mathcal{L}$ where:

- a witness for $x \in \mathcal{L}$ has length $m(|x|)$,
- and a proof of non-membership for $x \notin \mathcal{L}$ has length $L(|x|)$,

where m, L are polynomials. We will use the following ingredients:

- An unleveled FHE scheme $\mathsf{FHE}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ with malicious gate correctness.
- A somewhere extractable BARG scheme $\mathsf{BARG}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{TGen}, \mathsf{Ext})$.
- A somewhere extractable hash scheme $\mathsf{SEHash}.(\mathsf{Gen}, \mathsf{TGen}, \mathsf{Hash}, \mathsf{Open}, \mathsf{Ver}, \mathsf{Ext})$.

We now define the algorithm $\mathsf{TGen}, \mathsf{Gen}, \mathsf{Prove}$ and Ver. Note that while the $\mathsf{SNARG}_{L,\ell,t}$ itself does not need a trapdoor mode, we define TGen for the security analysis.

- **Trapdoor CRS algorithm.** $\mathsf{TGen}(1^\lambda, 1^n, \mathcal{R}, T_{\mathsf{SEH}}, S_{\mathsf{BARG}}, \mathsf{aux}; \mathsf{sd})$ operates as follows: On input a security parameter 1^λ, an input length 1^n, a tuple $T_{\mathsf{SEH}} \subseteq [2^\lambda]$ of size loc (where $\mathsf{loc} = \mathsf{poly}(\lambda, \log n, \log m)$, a set $S_{\mathsf{BARG}} \subseteq [2^\lambda]$ of size at most loc, and the explicit value will be chosen in the proof), a string $\mathsf{aux} \in \{0,1\}^{\ell(n)}$, and randomness $\mathsf{sd} = (\mathsf{sd}_{\mathsf{FHE}}, \mathsf{sd}_{\mathsf{ct}_0}, \mathsf{sd}_{\mathsf{SEH}}, \mathsf{sd}_{\mathsf{BARG}})$

1. **Generate FHE key:** Sample FHE keys $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda; \mathsf{sd}_{\mathsf{FHE}})$.
2. **Generate FHE ciphertext:** Sample a ciphertext $\mathsf{ct}_0 \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, \mathsf{aux}; \mathsf{sd}_{\mathsf{ct}_0})$.
3. **Generate SEH keys:** Let $T_{\mathsf{SEH}} = \{i_1, \ldots, i_{\mathsf{loc}}\}$. Let N^* be as defined in Step (7) of the prover algorithm.
   - Parse $\mathsf{sd}_{\mathsf{SEH}} = \{\mathsf{sd}_j\}_{j \in [\mathsf{loc}]}$.
   - For $j \in [\mathsf{loc}]$, sample a hash key $(\mathsf{hk}_j, \mathsf{td}_j) \leftarrow \mathsf{SEH.TGen}(1^\lambda, i_j; \mathsf{sd}_j)$.
   - Set $\mathsf{hk} = \{\mathsf{hk}_1, \ldots, \mathsf{hk}_{\mathsf{loc}}\}$ and $\mathsf{td}_{\mathsf{SEH}} = \{\mathsf{td}_1, \ldots, \mathsf{td}_{\mathsf{loc}}\}$.
   - In the remaining algorithms, we use the shorthand $\mathsf{SEH.Hash}(\mathsf{hk}, \cdot), \mathsf{SEH.Open}(\mathsf{hk}, \cdot), \mathsf{SEH.Ver}(\mathsf{hk}, \cdot)$ to denote the algorithms which carry out the respective algorithms loc times with respect to each of $\mathsf{hk}_1, \ldots, \mathsf{hk}_{\mathsf{loc}}$. We use the shorthand $\mathsf{SEH.Ext}(\mathsf{hk}, \mathsf{td}, \cdot)$ to denote running the Ext algorithm loc times with respect to each of $(\mathsf{hk}_1, \mathsf{td}_1), \ldots, (\mathsf{hk}_{\mathsf{loc}}, \mathsf{td}_{\mathsf{loc}})$.
4. **Generate BARG crs:** Sample $(\mathsf{crs}_{\mathsf{BARG}}, \mathsf{td}_{\mathsf{BARG}}) \leftarrow \mathsf{BARG.TGen}(1^\lambda, 1^{k^*}, 1^{s^*}, 1^{\mathsf{loc}}, S_{\mathsf{BARG}}; \mathsf{sd}_{\mathsf{BARG}})$, where we define k^* and s^* in Step (8) of the prover algorithm.
5. Output $(\mathsf{crs} = (\mathsf{hk}, \mathsf{ek}, \mathsf{ct}_0, \mathsf{crs}_{\mathsf{BARG}}), \mathsf{td} = (\mathsf{sk}, \mathsf{td}_{\mathsf{SEH}}, \mathsf{td}_{\mathsf{BARG}}))$.

- **CRS algorithm.** $\mathsf{Gen}(1^\lambda, 1^n, \mathcal{R})$ operates as follows:
  1. Let C denote the relation circuit for the language $\mathcal{L}$ on inputs of length n, and witnesses of length $m(n)$.
  2. Let $\ell := \ell(n)$.
  3. Set $T_{\mathsf{SEH}} = (0, \ldots, 0)$, $S_{\mathsf{BARG}} = \varnothing$ and $\mathsf{aux} = 1^\ell$.
  4. Sample $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^n, \mathcal{R}, T_{\mathsf{SEH}}, S_{\mathsf{BARG}}, \mathsf{aux})$, and output crs.
- **Prove algorithm.** $\mathcal{P}(\mathsf{crs}, \mathcal{R}, x, w)$ operates as follows:
  1. Parse $(\mathsf{hk}, \mathsf{ek}, \mathsf{ct}_0, \mathsf{crs}_{\mathsf{BARG}}) \leftarrow \mathsf{crs}$.
  2. Let $C_x(\cdot)$ denote the circuit evaluate $\mathcal{R}(x, \cdot)$.
  3. Compute the transcript of $C_x(w)$, and let $\omega = \omega_x(w)$ denote the wire assignment for this computation.
  4. Let $t := t(n)$. Homomorphically compute $\mathsf{ct}_1 \leftarrow \mathsf{FHE.Eval}_{\mathsf{ek}}(\mathsf{U}_{\mathsf{TM}}(\cdot, 1^t), \mathsf{ct}_0)$. Now, homomorphically evaluate $\mathsf{ct} \leftarrow \mathsf{FHE.Eval}_{\mathsf{ek}}(\mathsf{Expand}, \mathsf{ct}_1)$, where Expand is the efficient algorithm guaranteed in Theorem 7.6.
  5. Run the circuit $\mathsf{FHE.Eval}_{\mathsf{ek}}(U_\omega, \cdot)$ on ct gate-by-gate, where $U(\omega, \langle D \rangle)$ denotes the universal circuit the interprets $\langle D \rangle$ as a circuits and runs it on ω (see Section 7.1 for details on the description of U). Denote this circuit by $\Gamma_{\mathsf{ek},\mathsf{ct}}$.
  6. Let $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}} = \mathsf{Aug}(C_x, \Gamma_{\mathsf{ek},\mathsf{ct}})$ denote the computation done in steps (3) and (5) (i.e. the gate-by-gate FHE evaluation), and let $\tilde{\omega}$ denote the wire assignment.
  7. Let N^* be the length $\tilde{\omega}$. Let $v_{\mathcal{P}} = \mathsf{SEH.Hash}(\mathsf{hk}, \tilde{\omega})$.
  8. Compute a BARG proof that essentially proves that every gate of $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$ was computed correctly. Formally, let $R_{\mathsf{hk},v_{\mathcal{P}}}$ be the following relation circuit:
     - **Instance:** Description of some gate $g_i = (i, j_0, j_1, f)$, along with $\mathsf{aux}_i = (\beta_0, \beta_1)$ where β_b is the hardcoded value for wire j_b if any (e.g. if gate g_i takes as input any bit of $x, \mathsf{ek}, \mathsf{ct}$), and $\perp$ otherwise.

- **Witness:** Wire values w_i, w_{j_0}, w_{j_1} and openings $\rho_i, \rho_{j_0}, \rho_{j_1}$.

Then, $R_{\mathsf{hk}}((g_i, \mathsf{aux}_i), (w_i, w_{j_0}, w_{j_1}, \rho_i, \rho_{j_0}, \rho_{j_1}))$ does the following:

- Parse $g_i = (i, j_0, j_1, f)$.
- Verify that $\mathsf{SEH.Ver}(\mathsf{hk}, v_{\mathcal{P}}, i, w_i, \rho_i) = 1$.
- Parse $\mathsf{aux}_i = (\beta_0, \beta_1)$.
- For $b \in \{0, 1\}$, if $\beta_b \neq \bot$, check that $j_b = \beta_b$.
- Else, verify that $\mathsf{SEH.Ver}(\mathsf{hk}, v_{\mathcal{P}}, j_b, w_{j_b}, \rho_{j_b})$.
- Verify that $f(w_{j_0}, w_{j_1}) = w_i$.
- Output 1 if all of the above checks pass, and output 0 otherwise.

Let k^* denote the number of gates in $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$. Let s^* to be the size of the circuit $R_{\mathsf{hk},v}$. It is easy to see that $s^* = \mathsf{poly}(\lambda, \log |C|)$ since it simply verifies SEH openings. Note that one can derive the batch of instances $\{(g_i, \mathsf{aux}_i)\}_{i\in[k^*]}$ from the description of $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$ efficiently, and the prover can compute the corresponding $\{(w_i, w_{j_0}, w_{j_1}, \rho_i, \rho_{j_0}, \rho_{j_1})\}_{i\in[k^*]}$ efficiently.
Let

$$\pi_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Prove}\left(\mathsf{crs}, R_{\mathsf{hk},v_{\mathcal{P}}}, \{(g_i, \mathsf{aux}_i)\}_{i\in[k^*]}, \{(w_i, w_{j_0}, w_{j_1}, \rho_i, \rho_{j_0}, \rho_{j_1})\}_{i\in[k^*]}\right)$$

be the corresponding proof.

9. Let ρ_{out} be the opening to the output wire of $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$.
10. Output $(v_{\mathcal{P}}, \pi_{\mathsf{BARG}}, \rho_{\mathsf{out}})$.

- **The Verifier Algorithm.** $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}, \mathcal{R}, x, \pi)$ does the following:

1. Parse $(\mathsf{hk}, \mathsf{ek}, \boxed{\mathsf{ct}_0}, \mathsf{crs}_{\mathsf{BARG}}) \leftarrow \mathsf{crs}$ and $(v_{\mathcal{P}}, \pi_{\mathsf{BARG}}, \rho_{\mathsf{out}}) \leftarrow \pi$.
2. Let $t := t(n)$. Homomorphically compute $\mathsf{ct}_1 \leftarrow \mathsf{FHE.Eval}_{\mathsf{ek}}(\mathsf{U}_{\mathsf{TM}}(\cdot, 1^t), \mathsf{ct}_0)$. Now, homomorphically evaluate $\mathsf{ct} \leftarrow \mathsf{FHE.Eval}_{\mathsf{ek}}(\mathsf{Expand}, \mathsf{ct}_1)$.
3. Check that ρ_{out} corresponds to an opening to the bit 1.
4. Parse $\widehat{C_{x,\mathsf{ek},\mathsf{ct}}}$ as the batch of instances $\{(g_i, \mathsf{aux}_i)\}_{i\in[k^*]}$, and let $R = R_{\mathsf{hk},v_{\mathcal{P}}}$.
5. $\mathsf{BARG.Ver}(\mathsf{crs}_{\mathsf{BARG}}, R, \{(g_i, \mathsf{aux}_i)\}_{i\in[k^*]}, \pi_{\mathsf{BARG}}) = 1$.
6. Reject if any of the above checks fail, accept otherwise.

We will denote by $\mathsf{SNARG} = \mathsf{SNARG}.(\mathsf{Gen}, \mathcal{P}, \mathcal{V})$, and we will use $\mathsf{SNARG.TGen}$ only in the security proof.

Efficiency. It is easy to see that $\mathsf{crs} \leftarrow \mathsf{SNARG.Gen}(1^\lambda, 1^n, 1^m, \mathcal{R})$ has size $\mathsf{poly}(\lambda, \log n, \log m, \rho)$, prover runtime is $\mathsf{poly}(\lambda, T(n), s(n))$ and the proof size is $\mathsf{poly}(\lambda, \log n, \log m)$.

Remark 13. While the verifier runtime in the above scheme is also $\mathsf{poly}(\lambda, T(n), s(n))$, as in Remark 7, we can improve the efficiency to $\mathsf{poly}(\lambda, n)$ via the use of RAM SNARGs. We omit this optimization in the above construction for simplicity of analysis.

Consider the following security game $\boldsymbol{G}_{n,m,L} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ which formalizes the security game for SNARGs with L-EF-soundness:

- Define the language $\mathcal{U}$ of instances $x \in \{0,1\}^n$ where there exists a length at most $L(n)$ EF proof ϕ_x of the fact that for all $w \in \{0,1\}^{m(|x|)}$, $C_x(w) = 0$. This language can be verified in time $T = \mathsf{poly}(n, L)$.
- The challenger $\mathcal{C}$ takes as input $(1^T, x \in \{0,1\}^n, \phi_x, \mathcal{A}; (r, r_A))$ where $T = \mathsf{poly}(n, L)$:
  - Check that ϕ_x is an EF proof of the fact that $\forall w, C_x(w) = 0$.
  - Sample $\mathsf{crs} \leftarrow \mathsf{SNARG}(1^\lambda, 1^n, 1^m, \mathcal{R}; r)$.
  - Sample $\pi \leftarrow \mathcal{A}(\mathsf{crs})$.
  - If $\mathcal{V}(\mathsf{crs}, \mathcal{R}, x, \pi) = 1$, output 1. Else, output 0.
- Set $\gamma = 0$.

We say that SNARG is an EF-SNARG if $\boldsymbol{G}_{n,m,L}$ with respect to SNARG_L is polynomially secure.

Theorem 7.7. *Let $n := n(\lambda)$ and let m and L be polynomials. Assuming*

- *polynomially secure unleveled fully homomorphic encryption,*
- *polynomially secure somewhere extractable BARGs,*

$\mathcal{G}^{t,\ell}_{n,m,L}$ *when instantiated via* $\mathsf{SNARG}_{L,\ell,t}$ *is polynomially secure assuming t-time ℓ compression, i.e. for every $x \in \{0,1\}^n$ with a $(\ell(n), t(n), L(n))$-*EF *proof of non-membership, for every adversary $\mathcal{A}$ of size* $\mathsf{poly}(\lambda)$,

$$\Pr\left[\ \mathcal{V}(\mathsf{crs}, x, \pi) = 1 \ \ : \ \begin{array}{c}\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^m, \mathcal{R}) \\ \pi \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\ \right] \leq \mathsf{negl}(\lambda).$$

Proof sketch. The proof follows near identically to the proof in [HJLM26][29]. Consider some $x \in \{0,1\}^n$ such that

$$\Pr\left[\ \mathcal{V}(\mathsf{crs}, x, \pi) = 1 \ \ : \ \begin{array}{c}\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^m, \mathcal{R}) \\ \pi \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\ \right] \geq \varepsilon(\lambda)$$

where ε is some function. By FHE, SEH and BARG security, for all aux,

$$\Pr\left[\ \mathcal{V}(\mathsf{crs}, x, \pi) = 1 \ \ : \ \begin{array}{c}\mathsf{crs} \leftarrow \mathsf{TGen}(1^\lambda, 1^n, 1^m, \mathcal{R}, \varnothing, \varnothing, \mathsf{aux}) \\ \pi \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\ \right] \geq \varepsilon(\lambda) - \mathsf{negl}(\lambda).$$

For a fixed aux, by an averaging argument, one can pick $\mathsf{sd}_{\mathsf{FHE}}, \mathsf{sd}_{\mathsf{ct}}$ (FHE key and ciphertext randomness) so that the above probability is at least $\varepsilon(\lambda) - \mathsf{negl}(\lambda)$. In the rest of the proof, we will fix aux, $\mathsf{sd}_{\mathsf{FHE}}$ and $\mathsf{sd}_{\mathsf{ct}}$ in TGen and only specify T_{SEH} and S_{BARG}.

Suppose that $\mathsf{Expand}(\boldsymbol{K}^T(\mathsf{aux})) = \langle E \rangle$, where E is some circuit. Then, following the proof of [HJLM26], we can construct a $(p(\log n, \log L), \mathsf{poly}(\lambda, n, s, \varepsilon))$-local assignment generator of size $\mathsf{poly}(n, s, \lambda)$ for $\mathsf{Aug}(C_x, E_x)$ which is $\mathsf{poly}(\lambda, n, s, \varepsilon)$-almost accepting, where p is some polynomial.

On the other hand, suppose that ϕ_x is a proof of non-membership of x such that $\boldsymbol{K}^{T(n)}(\phi_x) \leq \ell(n)$. Suppose the machine $\mathcal{M}_\Pi$ witnesses this fact. Let $\mathsf{aux} = \mathcal{M}_\Pi$. Then, by definition, we have that $\mathsf{Expand}(\boldsymbol{K}^T(\mathcal{M}_\Pi)) = \langle E_x \rangle$, where E_x is the circuit given by Theorem 7.6.

By combining the above, one can choose $\mathsf{aux} = \mathcal{M}_\Pi$ to deduce that $\varepsilon = \mathsf{negl}(\lambda)$, as desired. □

[29] Looking forward, we are referencing [HJLM26] here rather than [JKLV24] because we will rely on their APC_1 formalization of the analysis in Section 10.

By applying Corollary 4.11 to the above and observing that $n, L = \mathsf{poly}(\lambda)$, we directly obtain Theorem 7.2.

Remark 14 (Removing the bound on n)**.** Let SNARG_L denote the scheme in Theorem 7.2. If the CRS does not grow with n or m, one can then remove 1^n and 1^m from the input to the CRS generation algorithm, and instead provide n and m as inputs in binary. Moreover, one can remove n and m entirely from the construction as follows. Consider some NP language $\mathcal{L}$ with relation given by a polynomial time relation $\mathcal{R}$. Suppose that any witness for x has size at most $m(|x|)$, and every $x \notin \mathcal{L}$ has a $L(|x|)$ EF proof of the fact that $\forall w, C_x(w) = 0$. Then, one can construct a SNARG for $\mathcal{L}$ without any bound on n via a standard powers-of-two trick as follows:

- $\mathsf{SNARG.Gen}(1^\lambda, \mathcal{R})$: For $i \in [\lambda]$, compute $\mathsf{crs}_i \leftarrow \mathsf{SNARG}_L.\mathsf{Gen}(1^\lambda, 2^i, m(2^i), \mathcal{R})$, and output $\mathsf{crs} = (\mathsf{crs}_1, \ldots, \mathsf{crs}_\lambda)$.
- $\mathsf{SNARG}.\mathcal{P}(\mathsf{crs}, \mathcal{R}, x, w)$:
  - Parse $\mathsf{crs} = (\mathsf{crs}_1, \ldots, \mathsf{crs}_\lambda)$.
  - Let $i = \lceil \log_2 |x| \rceil$.
  - Let $\pi \leftarrow \mathsf{SNARG}_L(\mathsf{crs}_i, \mathcal{R}, x, w)$ and output π.
- $\mathsf{SNARG}.\mathcal{V}(\mathsf{crs}, \mathcal{R}, x, \pi)$:
  - Parse $\mathsf{crs} = (\mathsf{crs}_1, \ldots, \mathsf{crs}_\lambda)$.
  - Let $i = \lceil \log_2 |x| \rceil$.
  - Output $\mathsf{SNARG}_L(\mathsf{crs}_i, \mathcal{R}, x, \pi)$.

Note that for polynomial-sized x, it suffices to rely on polynomial security of the corresponding crs_i. Finally, one can use the same RAM delegation composition as in Remark 7 to boost the verifier efficiency to be quasilinear in $(\mathcal{R}, x, \pi)$.

Part III
Formalization in Bounded Arithmetic

8 Proof Complexity Generators against Feasible Cryptography

8.1 Notation and Basic Concepts

We follow closely the notation and concepts built in [HJLM26] for the formalization of cryptography. For the convenience of readers, we give a brief overview and pointers to specific sections in [Jeř07a] and [HJLM26].

- (*Formalization of Approximate Counting*). We use Jeřábek's formalization and notation for approximate counting [Jeř07a]. A (bounded) definable set $X \subseteq \{0,1\}^n$ is the set of accepted inputs of a given circuit; for definable sets $X_1, X_2 \subseteq \{0,1\}^n$. Jeřábek introduced a formalization of size comparison $X_1 \lesssim_\varepsilon X_2$ (see, e.g., [HJLM26, Definition 3.1]) with the intended meaning that $|X_2| \leq |X_1| + \varepsilon 2^n$; two set are approximately of the same size, denoted by $X_1 \approx_\varepsilon X_2$, if $X_1 \lesssim_\varepsilon X_2$ and $X_1 \lesssim_\varepsilon X_2$.

- (*Basic Properties*). Jeřábek proves in APC_1 that for every bounded definable set $X \subseteq \{0,1\}^n$ and any $\varepsilon^{-1} \in \mathsf{Log}$, there is a number $s \leq 2^n$ (the size of X) such that $X \approx_\varepsilon s$. It means that the size of every definable set can be counted approximately in APC_1. Moreover, under this formalization of approximate counting, many basic properties (such as the inclusion-exclusion principle) can be proved in APC_1, see [HJLM26, Section 3.2].

- (*Axiom* $\mathsf{prBPP} = \mathsf{prP}$). We use the formalization of $\mathsf{prBPP} = \mathsf{prP}$ in the language of APC_1 proposed in [HJLM26]. Specifically, for a PV function symbol $\mathsf{capp}(\cdot,\cdot)$, the axiom formalizes the following: For every circuit $C : \{0,1\}^n \to \{0,1\}$, string t, number $s \leq 2^n$, and $\varepsilon^{-1} \in \mathsf{Log}$, let $X := \{x \in \{0,1\}^n \mid C(x) = 1\}$, then if $X \approx_\varepsilon s$, then

  $$|\mathsf{capp}(C, 1^t) - s| \leq (\varepsilon + t^{-1}) \cdot 2^n;$$

  see [HJLM26, Definition 4.3].

- (*Formalization of Probability*). The axiom $\mathsf{prBPP} = \mathsf{prP}$ gives a natural formalization of (approximate) acceptance probability of circuits: For a circuit C and a precision parameter $\delta^{-1} \in \mathsf{Log}$, we use $\Pr_\delta[C]$ to denote $\mathsf{capp}(C, 1^{\delta^{-1}})/2^n$. Using the approximate conting toolkits in APC_1, we can prove properties of (approximate) probability in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$", e.g., the union bound

  $$\Pr_\delta[C_1 \vee C_2] \leq \Pr_\delta[C_1] + \Pr_\delta[C_2] + 3\delta$$

  is provable (see [HJLM26, Proposition 7.5]). More results about approximate probability can be found in [HJLM26, Section 7.1 to 7.3].

In addition to approximate probability $\Pr_\delta[\cdot]$ and basic tools in probability theory, a few more abbreviations are developed [HJLM26] to avoid (to a large extent) carrying the precision parameter δ around in formalizations:

- (*Indistinguishability*). One of the most useful abbreviations is computational indistinguishability. For two distributions $\mathcal{D}_1, \mathcal{D}_2$ over $\{0,1\}^n$ (defined by their sampler circuits), $t \in \mathsf{Log}$, and $\varepsilon \in [0,1]$, we

say $\mathcal{D}_1 \approx_{t,\varepsilon} \mathcal{D}_2$ if for every (probabilistic) circuit $\mathcal{A} : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}$ of size t and every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$|\Pr_\delta[C(\mathcal{D}_1)] - \Pr_\delta[C(\mathcal{D}_2)]| \leq 2\delta + \beta + \varepsilon.$$

Here, the additive terms $2\delta + \beta$ accommodate the precision for approximate counting; see [HJLM26, Section 4.2]. Several inner theorems and meta-theorems, including the standard hybrid arguments, are proved within $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$; see [HJLM26, Section 7.4].

- (*Search Games*). Another useful concept is search security games: a probabilistic circuit $\mathcal{C}$ called challenger proposes a challenge to an adversary $\mathcal{A}$, the adversary outputs an answer, and the challenger verifies whether the answer is correct. An abbreviation $\mathsf{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon$ is defined, with the intended interpretation that the adversary $\mathcal{A}$ has advantage at most ε in the game $\mathcal{G}$. The formal definition is given in [HJLM26, Section 5.1]. A collection of meta-theorems are proved in [HJLM26, Section 7.5].

In this work, we will only use the notion of *indistinguishability*.

8.2 Infinitely-Often Provability

For cryptographic applications, it is crucial to consider *almost everywhere* assumptions, namely the adversary cannot succeed infinitely often. This is because we want the security guarantees to hold almost everywhere, or equivalent, we have to make use of adversaries that only break the scheme infinitely often. Subsequently, unprovability assumptions also need to consider a notion of “infinitely often” provability as the “adversary” aims to prove certain mathematical statements.

A straightforward formalization and its limitation. One way to formalize infinite often provability is to use quantifier alternations. Namely, $\psi(x)$ holds for infinitely many x is formalized as the following sentence in the language of PV_1:

$$\Phi := \forall n_0 \in \mathsf{Log}\ \exists x \in \{0,1\}^*\ (|x| > n_0 \wedge \psi(x)).$$

However, this formalization is insufficient for our purpose.

Recall that a bounded quantifier is of form $\forall x \leq t$ or $\exists x \leq t$, where t is a term. A sentence is said to be a $\forall\Sigma^b_i$-sentence if it is logical equivalent to a formula with outermost universal quantifiers and i alternations of bounded quantifiers, and is said to be a $\forall\Sigma^b_\infty$-sentence if it is a $\forall\Sigma^b_i$-sentence for some $i \in \mathbb{N}$.

Theorem 8.1 (Implicit in [Par71]). *Let $\mathcal{T}$ be any theory in the language of PV_1, whose axioms are $\forall\Sigma^\infty_b$-formulas. Then for any formula $\psi(x,y)$, if $\mathcal{T} \vdash \forall x\ \exists y\ \psi(x,y)$, then there is a term t in the language of PV_1 such that $\mathcal{T} \vdash \forall x\ \exists y \leq t(x)\ \psi(x,y)$.*

Note that Parikh’s theorem applies to almost all bounded theories, including PV_1, the S_2 hierarchy corresponding to polynomial-time hierarchy, and their extensions with $\mathsf{WPHP}(\mathsf{PV}), \mathsf{dWPHP}(\mathsf{PV})$, or falsifiable cryptographic assumptions.

Assume that $\mathcal{T}$ is a sound theory that satisfies the premise of Parikh’s theorem, and we have $\mathcal{T} \vdash \Phi$. Then by Parikh’s theorem, there is a term t in the language of PV_1 such that

$$\mathcal{T} \vdash \forall n_0 \in \mathsf{Log}\ \exists x \leq t(1^{n_0})\ (|x| > n_0 \wedge \psi(x)).$$

In particular, as t is a polynomial-time function, $t(1^{n_0})$ is of length at most $\mathsf{poly}(n_0)$. By the soundness of $\mathcal{T}$, there must exist a string x with length between $[n_0, \mathsf{poly}(n_0)]$ satisfying $\psi(x)$ for any n_0. This implies

that, in addition to having infinitely many x satisfying $\psi(x)$, the length of such good x's is polynomially *dense*.

This is a property that will cause some trouble in security proofs. In general, an adversary wins if it succeeds on infinitely many instances, and these the lengths of these instances are not necessarily polynomially dense. However, $\mathcal{T} \vdash \Phi$ mandates provability on polynomially dense instance lengths, so it captures a *less general* winning condition for the adversary.

Our formalization. We consider a more general definition for provability of infinite often statements. Instead of using quantifiers, we consider an explicit NP verifier $V(x, y)$ that certifies "good" x's, i.e., $\exists y\, V(x, y) = 1$ implies $\psi(x)$. We require that (1) it must certify infinitely many good strings, and (2) its correctness must be provable in the theory $\mathcal{T}$.

Let t be a time bound. Formally:

Definition 8.2 (Provability of i.o. statements)**.** Let $\psi(x)$ be a formula with x being the only free variable in the language of PV_1, and $\mathcal{T}$ be a theory that contains PV_1. We say that $\mathcal{T}$ proves $\psi(x)$ for infinitely many x in time t, denoted by $\mathcal{T} \vdash (\text{i.o.})^t x\ \psi(x)$, if there is a deterministic polynomial time algorithm $V(x, w)$ and $t'(m) := t(m) \cdot \mathsf{poly}(m)$ such that the following two conditions hold:

- (*Completeness*). For infinitely many x, there exists $y \in \{0,1\}^{t'(|x|)}$ such that $V(x, y) = 1$.
- (*Provable Soundness*). $\mathcal{T} \vdash \forall x\, \forall w\ (V(x, w) = 1 \to \psi(x))$.

In particular, we say that $\mathcal{T}$ proves $\psi(x)$ for infinitely many x (in polynomial time), denoted by $\mathcal{T} \vdash (\text{i.o.})x\ \psi(x)$, if $\mathcal{T} \vdash (\text{i.o.})^1 x\ \psi(x)$.

Note that in particular, when $\psi(x)$ is a Π^b_1-formula, i.e., there exists $\phi(x, y)$ and a PV-term t such that $\psi(x) := \forall y \in \{0,1\}^{|t(x)|}\ \phi(x, y)$, the sentence $\forall x\, \forall y\ (V(x, y) = 1 \to \psi(x))$ is a universal sentence, as it is equivalent to

$$\forall x\, \forall w\, \forall y \le t(x)\ (V(x, w) = 1 \to \phi(x, y)).$$

We will be mostly interested in this special case.

8.3 Proof Complexity Generators against Bounded Arithmetic

Now we are ready to introduce our assumption of proof complexity generators against bounded arithmetic proofs. Let $\mathcal{T}$ be an extension of PV_1, and $\ell(\cdot) : \mathsf{Log} \to \mathsf{Log}$ be a PV function (called the *seed length* of the PCG).

Definition 8.3 (Proof complexity generator against bounded arithmetic)**.** Let $\mathcal{C} = \{C_m : \{0,1\}^{\ell(m)} \to \{0,1\}^m\}_{m \in \mathbb{N}}$ is a polynomial-time uniform family of circuits, $t(n)$ be a time bound, and $\varphi(y)$ is a straightforward $\Pi^b_1(\mathsf{PV})$-formula that states:

- For $m := |y|$ and every string x of length $\ell(m)$, $C_m(x) \neq y$.

Then $\mathcal{C}$ is said to be a t-secure proof complexity generator against $\mathcal{T}$ if $\mathcal{T} \nvdash (\text{i.o.})^t y\ \varphi(y)$. In particular, it is said to have polynomial security if $t = 1$, and is said to be subexponential security if $t(n) = 2^{n^\varepsilon}$ for some constant $\varepsilon \in (0, 1)$.

In more details, the family of circuits is generated by a PV function $g(\cdot)$, i.e., $g(1^m)$ outputs the description of C_m for $m \in \mathsf{Log}$. We omit the generator g and write C_m directly for simplicity.

The following is an equivalent formalization that is closer to Definition 4.1.

Definition 8.4 (proof complexity generator against NP with provable soundness)**.** Let $\mathcal{C} = \{C_m : \{0,1\}^{\ell(m)} \to \{0,1\}^m\}_{m\in\mathbb{N}}$ is a polynomial-time uniform family of circuits. It is said to be a t-secure proof complexity generator against $\mathcal{T}$ provable soundness, if for any deterministic polynomial-time algorithm $\mathcal{A}(1^m, y, \tau)$ and any $t'(m) := t(m) \cdot \mathsf{poly}(m)$, the following conditions cannot hold simultaneously:

- (*Completeness*): For infinitely many $m \in \mathbb{N}$, there exists a string $y \in \{0,1\}^m \setminus \mathsf{Range}(C_m)$ and a string $\tau \in \{0,1\}^{t'(m)}$ such that $\mathcal{A}(1^m, y, \tau) = 1$.
- (*Provable Soundness*): The following sentence is unprovable in the theory $\mathcal{T}$: For every $m \in \mathbb{N}$, every string $y \in \mathsf{Range}(C_m)$ and every τ, $\mathcal{A}(1^m, y, \tau) = 0$.

Proposition 8.5. *Assume straightforward formalization of sentences in the language of* PV_1*, Definition 8.3 and Definition 8.4 are equivalent.*

Proof Sketch. This immediately follows from the definition of infinite-often provability. In particular, the verifier V in Definition 8.2 (from Definition 8.3) corresponds to the adversary $\mathcal{A}$ in Definition 8.4, and the completeness and provable soundness properties in Definition 8.2 and 8.4 tightly match with each other. □

8.4 Time-bounded Chaitin Incompleteness

Similar to Section 4.1.2, we consider an alternative formalization of PCG against bounded arithmetic. Fix a PV function $\mathsf{U}_{\mathsf{TM}}(N, x, 1^T)$ that serves as a universal Turing machine. It takes as input the encoding of a machine N and its input x, as well as a time bound T. It simulates $N(x)$ for T steps and outputs its configuration; in particular, if $N(x)$ halts in T steps, it outputs $N(x)$.

Let $t : \mathsf{Log} \to \mathsf{Log}$ be a PV function. For $x \in \{0,1\}^*$ and $\ell \in \mathsf{Log}$, we say $\boldsymbol{K}^t(x) \le \ell$ if there exists $\langle N, x\rangle$ of length at most ℓ such that $\mathsf{U}_{\mathsf{TM}}(N, x, 1^{t(|x|)}) = x$. Note that "$\boldsymbol{K}^t(x) \le \ell$" is $\Sigma^b_1(\mathsf{PV})$-formula with x and $\ell \in \mathsf{Log}$ as free variables.

Definition 8.6 (Time-bounded Chaitin against bounded arithmetic)**.** Let $t(\cdot), \ell(\cdot) : \mathsf{Log} \to \mathsf{Log}$ be PV functions, and $t'(n)$ be a time bound. Let $\varphi(y)$ be a straightforward $\Pi^b_1(\mathsf{PV})$-formula that $\neg(\boldsymbol{K}^t(y) \le \ell(|y|))$, that is:

- For $m := |y|$ and every string σ of length $\ell(m)$, $\mathsf{U}_{\mathsf{TM}}(\sigma, 1^{t(m)}) \neq y$.

We say (t, ℓ)-Chaitin incompleteness holds against $\mathcal{T}$ in t'-time if $\mathcal{T} \nvdash (\text{i.o.})^{t'} y\ \varphi(y)$.

As described in Section 4.1.2, the universal Turing machine U_{TM} (for fixed $t(\cdot), \ell(\cdot)$) can be viewed as a candidate proof complexity generator, where the circuit $C_m : \{0,1\}^{\ell(m)} \to \{0,1\}^m$ is defined as $C_m(\sigma) := \mathsf{U}_{\mathsf{TM}}(\sigma, 1^{t(m)})$. On the other hand, the following proposition is needed to prove that the generator is universal:

Proposition 8.7. *For uniform family of circuits $C_m : \{0,1\}^{\ell(m)} \to \{0,1\}^m$ with a* PV *generator g, there exists a* PV *function $t(\cdot) : \mathsf{Log} \to \mathsf{Log}$ that is a polynomial and a constant $c \in \mathbb{N}$ such that the following is provable in* PV_1*. For every $m \in \mathsf{Log}$ and $y = C_m(x) \in \{0,1\}^m$, there exists a string $\sigma \in \{0,1\}^{c\cdot\ell(m)+c}$ such that $\mathsf{U}_{\mathsf{TM}}(\sigma, 1^{t(m)}) = y$.*

Proof Sketch. Pick a sufficiently large constant c, and set $t(m) = cm^c$, so that $t(m)$ is larger than the total time complexity of generating C_m and evaluating $C_m(x)$. Then the proposition follows by setting $\sigma = \langle N, x\rangle$, where $N(x)$ evaluates $C_m(x)$. □

Theorem 8.8 (counterpart of Theorem 4.7)**.** *Let $k(m), t'(m) : \mathsf{Log} \to \mathsf{Log}$ be functions, $k(m) = o(m)$, and $\mathcal{T}$ be an extension of PV_1. Then the following two statements are equivalent:*

(1). *There exists a t'-secure PCG against $\mathcal{T}$ with seed length $c \cdot k(m)$, for some constant $c \in \mathbb{N}$.*

(2). *There is a polynomial $t(m)$ such that $(t, c \cdot k(m))$-Chaitin incompleteness holds against $\mathcal{T}$ in t' time, for some constant $c \in \mathbb{N}$.*

Proof Sketch. The proof follows closely the proof of Theorem 4.7, where (1) $\Rightarrow$ (2) is proved via Proposition 8.7 using the fact that $\mathcal{T}$ is an extension of PV_1. □

9 Upgrading Security via PCG against Feasible Cryptography

The purpose of the section is to develop a master theorem similar to Theorem 4.10 that, instead of assuming PCG against NP, works with PCG against bounded arithmetic.

9.1 Formalizations in Bounded Arithmetic

In this subsection, we explain the formalization of security games and their security under compression. These formalization would be used in the statement and proof of the master theorem (see Theorem 9.3). We stress that our formalizations follows standard methodology in bounded arithmetic literature, and the results does not exploit specific formalization details. Therefore, these results should hold under any "natural formalizations".

Formalizing security games. Let $\mathcal{C}$ be a PV function (and thus a polynomial time algorithm), $\mathcal{U}$ be an $\mathsf{NTIME}[T]$ language with witness length $p(n) \leq T(n)$, where T is a PV-formalizable size bound (see Definition 3.2). Let $\mathcal{U}_{\mathsf{pad}}(x, y, w)$ be PV function such that the following conditions hold:

- (*Completeness*). If $x \in \mathcal{U}$, there exists a witness w of length $p(n)$ such that for every $m \geq T(n)$, $\mathcal{U}_{\mathsf{pad}}(x, 1^t, w) = 1$.
- (*Soundness*). If $x \notin \mathcal{U}$, the for every w of length $p(n)$ and every $t \geq T(n), \mathcal{U}_{\mathsf{pad}}(x, 1^t, w) = 0$.

We can thus define "$x \in \mathcal{U}$" in PV_1 provided that there exists $t \in \mathsf{Log}$ such that $t \geq T(n)$. (Note that this padding is not necessary if $T(n) = \mathsf{poly}(n)$.)

Security under compression. We are now ready to formalize the *security under compression* of a game $\mathcal{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ in the above encoding, starting from a parametrized version.

Definition 9.1 (Parametrized security of $\boldsymbol{G}$ under compression, in $\mathsf{APC}_1 + $ "$\mathsf{prBPP} = \mathsf{prP}$")**.** Let $t(\cdot), \ell(\cdot) : \mathsf{Log} \to \mathsf{Log}$ be PV functions, where t is a polynomial. Let $\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T$ be PV functions mentioned above, $\gamma \in (0, 1)$ be a constant, $n, s, \tau, \varepsilon^{-1} \in \mathsf{Log}$.

We define $(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)$-$\mathsf{GSecure}^{s,\varepsilon}_{n,\tau}$ as the following formula (with variables $n, s, \tau, \varepsilon^{-1}$ in unary encoding): Suppose that $\tau \geq T(n)$. Let $x \in \{0,1\}^n$ and w be strings such that $\mathcal{U}_{\mathsf{pad}}(x, 1^\tau, w) = 1$, A be a circuit of size at most s, $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$, and $m = |x\|w| \in \mathsf{Log}$. Then if $\boldsymbol{K}^t(x\|w) \leq \ell(m)$,

$$\Pr_\delta[D] \leq \gamma + \varepsilon + \delta + \beta,$$

where D is the circuit $D(r) := \mathcal{C}(1^\tau, x, w, A; r)$.

Note that $(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)$-$\mathsf{GSecure}^{s,\varepsilon}_{n,\tau}$ is a $\forall\Pi^b_1(\mathsf{PV})$-formula; the formula $\boldsymbol{K}^t(x\|w) \leq \ell(m)$ is a $\Sigma^b_1(\mathsf{PV})$-formula, but it appears as the antecedent of an implication. It formalizes that the security game defined by $(\mathcal{C}, \mathcal{U}, T, \gamma, n)$ (using the formalization described above) is ε-secure against any adversary of size at most s assuming t-time ℓ-compression.

We now formalize asymptotic security of games, which tightly matches Definition 4.9.

Definition 9.2 (Security of $\boldsymbol{G}$ under compression, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Fix PV functions $\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, \gamma, t, \ell$ and size bound T as mentioned above, and $S(\lambda), \mathcal{E}(\lambda), N(\lambda) = \lambda^{\Omega(1)}$ be formalizable size bounds, and $\lambda_0(\cdot) : \mathbb{N} \to \mathbb{N}$ be a function (not necessarily in PV).

We define $(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)$-$\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0]$ as the set of the following sentence for each $k \in \mathbb{N}$:

- For every $\lambda, n, \tau, s, \varepsilon^{-1} \in \mathsf{Log}$, satisfying that $\lambda > \lambda_0(k)$, $n \leq N(\lambda)$, $s \leq S(\lambda) \cdot \lambda^k$, $\varepsilon^{-1} \leq \mathcal{E}(\lambda)$, $\tau \geq T(n)$, we have $(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)$-$\mathsf{GSecure}^{s,\varepsilon}_{n,\tau}$.

Note that each sentence in the set $(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)$-$\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0]$ is a universal (i.e. $\forall\Pi^b_0$) sentence with no existential quantifier. It is also worth noting that the formalization allows us to deal with polynomial and subexponential security in a similar manner. To see this, notice that $S, \mathcal{E}, N$ are formalizable size bounds (see Section 3.1.3) and could be superpolynomial in general, and it is easy to verify that standard subexponential functions 2^{n^ε} is formalizable. More explicitly, there are PV functions $N_{\leq}(n, m)$ formalizing $m \leq N(n)$, and the bounded quantifier $\forall n \leq N(\lambda)$ is the abbreviation of $\forall n\, N_{\leq}(\lambda, n)$.

9.2 Master Theorem with Bounded Arithmetic

Theorem 9.3 (Master Theorem with Bounded Arithmetic)**.** *Let $t(\cdot), \ell(\cdot) : \mathsf{Log} \to \mathsf{Log}$ be PV functions, where t is a polynomial. Let $\mathcal{T}$ be a sound extension of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$, $\mathcal{C}, \mathcal{U}_{\mathsf{pad}}$ be PV functions, T be a PV-formalizable size bound, and $\mathcal{U} \in \mathsf{NTIME}[T]$ as described in Section 9.1. Let $p(n)$ be the witness length of $\mathcal{U}$, and $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ be the security game defined by these functions. Let $S(\lambda), \mathcal{E}(\lambda), N(\lambda) = \lambda^{\Omega(1)}$ be PV-formalizable size bounds, and $\lambda_0 : \mathbb{N} \to \mathbb{N}$ be a function.*

Suppose that $\mathcal{T}$ is sound and (t, ℓ)-Chaitin incompleteness holds against $\mathcal{T}$ in time

$$t'(m) = \mathsf{poly}(T(N(\lambda)), S(\lambda), \mathcal{E}(\lambda), \lambda),$$

where $m := N(\lambda) + p(N(\lambda))$. If for some $\lambda_0(\cdot)$, $\mathcal{T}$ proves every sentence in

$$(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)\text{-}\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0],$$

then $\boldsymbol{G}$ is $(S(\lambda), 2/\mathcal{E}(\lambda))$-secure.

At a high level, the theorem is proved by using the same adversary construction for the proof complexity generator and formalizing the soundness analysis in the proof of Theorem 4.10 in the theory $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

Proof of Theorem 9.3. Fix functions t, ℓ, PV-formalizable size bounds $T, S, \mathcal{E}, N$, PV functions $\mathcal{C}, \mathcal{U}_{\mathsf{pad}}$ (and $\mathcal{U} \in \mathsf{NTIME}[T]$ corresponding to $\mathcal{U}_{\mathsf{pad}}$), and $\gamma, p, \boldsymbol{G}, \lambda_0(\cdot)$ as described above. Suppose that $\mathcal{T}$ is an extension of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ that is sound, and (t, ℓ)-Chaitin incompleteness holds against $\mathcal{T}$ in time t' on input length $m := N(\lambda) + p(N(\lambda))$. Moreover, suppose that $\mathcal{T}$ proves every sentence in $(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)$-$\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0]$.

Assume, towards a contradiction, that the game $\boldsymbol{G}$ is not $(S(\lambda), 2/\mathcal{E}(\lambda))$-secure. By definition, there is a polynomial $q = q(\lambda)$ and an infinite sequence $\Lambda = \{\lambda_i\}_{i \in \mathbb{N}}$ of security parameters such that for every

$\lambda \in \Lambda$, there is an adversary that has winning probability at least $\gamma + 2\varepsilon$ in $\boldsymbol{G}_{N(\lambda),S(\lambda)\cdot q(\lambda)}$. In other words, there is a string $x^*_\lambda \in \mathcal{U}$ of length $N(\lambda)$, a witness w^*_λ for $x^*_\lambda \in \mathcal{U}$, and a circuit A^*_λ of size $S(\lambda) \cdot q(\lambda)$ such that

$$\Pr_r[\mathcal{C}(1^T, x^*_\lambda, w^*_\lambda, A^*_\lambda; r) = 1] \geq \gamma + 2\varepsilon, \tag{9.1}$$

where $T = T(N(\lambda))$. Any such tuple $x^*_\lambda, w^*_\lambda, A^*_\lambda$ is said to be a *strong insecurity witness* for the security parameter λ.

Construction of Adversary. To reach a contradiction, we will show that (t, ℓ)-Chaitin incompleteness with aforementioned parameters do not hold. In other words, we will prove that $\mathcal{T} \vdash (\text{i.o.})^{t'} y\ \varphi(y)$, where $\varphi(y)$ is the straightforward $\Pi^b_1(\mathsf{PV})$-formula that states $\neg(\boldsymbol{K}^t(y) \leq \ell(|y|))$:

- for $m := |y|$ and every string σ of length $\ell(m)$, $\mathsf{U}_{\mathsf{TM}}(\sigma, 1^{t(m)}) \neq y$.

Furthermore, by Definition 8.2, we will need to construct a deterministic polynomial time algorithm $V(y, \pi)$ such that for a time bound $t'' = t' \cdot \mathsf{poly}(m)$, the following two conditions hold:

- (*Completeness*). For infinitely many y, there exists π of length $t''(|y|)$ such that $\mathcal{A}(y, \pi) = 1$.
- (*Provable Soundness*). $\mathcal{T} \vdash \forall y\ \forall \pi\ (V(y, \pi) = 1 \to \varphi(y))$.

Let $k, \lambda_1 \in \mathbb{N}$ be constants such that $q(\lambda) \leq \lambda^k$ for any $\lambda > \lambda_1$. We will define $V(y, \pi)$ as the adversary $\mathcal{A}$ described in the proof of Theorem 4.10, with slight modifications (marked in blue) so that it can be formalized as a PV function. The algorithm V parses π as a tuple of form $(1^\lambda, 1^n, 1^\tau, 1^s, 1^{\varepsilon^{-1}}, A)$, and rejects immediately if one of the tests fails:

- (*Size Bound*). $n = N(\lambda)$, $s \cdot \lambda^k = S(\lambda)$, $\varepsilon^{-1} = \mathcal{E}(\lambda)$, and $\tau = T(n)$; see Definition 3.2.
- (*Instance Size*). The algorithm hard-codes $\lambda_0(k), \lambda_1 \in \mathbb{N}$, and checks $\lambda > \max\{\lambda_0(k), \lambda_1\}$.
- (*Circuit Size*). A is a circuit of size at most s.
- (*Instance*). $y = x \| w$, where $|x| = n$ and w is a witness of $x \in \mathcal{U}$, or equivalently, $\mathcal{U}_{\mathsf{pad}}(x, 1^\tau, w) = 1$. Note that this step is feasible as 1^τ is given as a part of π.

Recall that $\mathcal{T}$ is an extension of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$", where "$\mathsf{prBPP} = \mathsf{prP}$" allows the formalization of approximate acceptance probability of circuits as PV functions. Let $D(r) := \mathcal{C}(1^\tau, y, w, A; r)$ be a circuit, the adversary works as follows:

- If $\Pr_{\varepsilon/10}[D] \geq \gamma + 5\varepsilon/3$, the algorithm $V(y, \pi)$ accepts.
- If $\Pr_{\varepsilon/10}[D] \leq \gamma + 4\varepsilon/3$, the algorithm $V(y, \pi)$ rejects.

It can be verified that the algorithm runs in polynomial time (in its input length).

Completeness. Recall that there are infinitely many λ and strong insecurity witnesses $x^*_\lambda, w^*_\lambda, A^*_\lambda$. Fix any such $\lambda \in \mathbb{N}$. Let $y^*_\lambda := x^*_\lambda \| w^*_\lambda$, and

$$\pi^*_\lambda := (1^\lambda, 1^{N(\lambda)}, 1^{T(N(\lambda))}, 1^{S(\lambda)}, 1^{\mathcal{E}(\lambda)}, A^*_\lambda).$$

Note that the length of π^*_λ is at most $t' \cdot \mathsf{poly}(m)$, where $m := |y^*_\lambda|$. As $N, T, S, \mathcal{E}$ are PV-formalizable size bounds (see Definition 3.2) and $\mathcal{T}$ is a sound extension of PV, $V(y^*_\lambda, \pi^*_\lambda)$ passes the (*Size Bound*) test in the standard model. For sufficiently large λ, it also passes the (*Instance Size*) test. Moreover, it is clear that $V(y^*_\lambda, \pi^*_\lambda)$ also passes the (*Circuit Size*) test and (*Instance*) test.

Furthermore, as $\mathcal{T}$ is a sound extension of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$, $\Pr_{\varepsilon/10}[\cdot]$ is a correct algorithm for the Circuit Acceptance Probability Problem within an additive error $\varepsilon/10$, and subsequently

$$\Pr_{\varepsilon/10}[D] \geq \Pr_{r \leftarrow \mathcal{U}}[D(r) = 1] - \frac{\varepsilon}{10} \geq \gamma + 2\varepsilon - \frac{\varepsilon}{10} \geq \gamma + \frac{5\varepsilon}{3},$$

where the first inequality follows from the correctness of $\Pr_{\varepsilon/10}[\cdot]$, and the second inequality follows from the fact that $x^*_\lambda, w^*_\lambda, A^*_\lambda$ is a strong insecurity witness. It then follows from the definition of $V(y^*_\lambda, \pi^*_\lambda)$ that the algorithm accepts.

Provable Soundness. We will show that the theory $\mathcal{T}$ proves the $\forall\Sigma^b_0(\mathsf{PV})$-sentence: $\forall y\, \forall \pi\ (V(y,\pi) = 1 \to \varphi(y))$. Recall that the theory $\mathcal{T}$ proves every sentence in $(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)$-$\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0]$. In particular, by the definition of the set, it proves the following sentence:

- For every $\lambda, n, \tau, s, \varepsilon^{-1} \in \mathsf{Log}$, satisfying that $\lambda > \lambda_0(k)$, $n \leq N(\lambda)$, $s \leq S(\lambda) \cdot \lambda^k$, $\varepsilon^{-1} \leq \mathcal{E}(\lambda)$, $\tau \geq T(n)$, we have
$$(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)\text{-}\mathsf{GSecure}^{s,\varepsilon}_{n,\tau}.$$

In the rest of the proof, we argue in the theory $\mathcal{T}$.

Fix y and π such that $V(y,\pi) = 1$, our goal is to prove that $\varphi(y)$ holds; in other words, $\neg(\boldsymbol{K}^t(y) \leq \ell(|y|))$. Notice that by the definition of V, as $V(y,\pi)$ accepts, it necessarily passes all the tests. In other words, we have:

- π is a tuple of form $(1^\lambda, 1^n, 1^\tau, 1^s, 1^{\varepsilon^{-1}}, A)$, and thus $\lambda, n, \tau, s, \varepsilon^{-1} \in \mathsf{Log}$.
- (*Size Bound*). $n = N(\lambda)$, $s \cdot \lambda^k = S(\lambda)$, $\varepsilon^{-1} = \mathcal{E}(\lambda)$, and $\tau = T(n)$.
- (*Instance Size*). The algorithm hard-codes $\lambda_0(k), \lambda_1 \in \mathbb{N}$, and checks $\lambda > \max\{\lambda_0(k), \lambda_1\}$.
- (*Circuit Size*). A is a circuit of size at most $s \cdot \lambda^k$.
- (*Instance*). $y = x \| w$, where $|x| = n$ and $\mathcal{U}_{\mathsf{pad}}(x, 1^T, w) = 1$.

Moreover, let D be the circuit $D(r) := \mathcal{C}(1^\tau, x, w, A; r)$, it must satisfy that

$$\Pr_{\varepsilon/10}[D] > \gamma + \frac{4\varepsilon}{3}, \tag{9.2}$$

as otherwise $V(y,\pi)$ would reject.

As $\lambda > \lambda_0(k)$, $n \leq N(\lambda)$, $s \leq S(\lambda) \cdot \lambda^k$, $\varepsilon^{-1} \leq \mathcal{E}(\lambda)$, $\tau \geq T(n)$, it follows that the formula $(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)$-$\mathsf{GSecure}^{s,\varepsilon}_{n,\tau}$ holds, which means that:

- Suppose that $\tau \geq T(n)$. Let x, w be strings such that $\mathcal{U}_{\mathsf{pad}}(x, 1^\tau, w) = 1$, A be a circuit of size at most s, $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$, and $m = |x\|w| \in \mathsf{Log}$. Then if $\boldsymbol{K}^t(x\|w) \leq \ell(m)$,
$$\Pr_\delta[D] \leq \gamma + \varepsilon + \delta + \beta, \tag{9.3}$$
where D is the circuit $D(r) := \mathcal{C}(1^\tau, x, w, A; r)$.

Subsequently, for $\delta = \beta := \varepsilon/10$, we concludes that if $\boldsymbol{K}^t(y) \leq \ell(m)$, then

$$\Pr_{\varepsilon/10}[D] \leq \gamma + \varepsilon + \frac{\varepsilon}{10} + \frac{\varepsilon}{10} \leq \gamma + \frac{6\varepsilon}{5}, \tag{9.4}$$

Combining this with Equation (9.2), we immediately conclude that $\neg(\boldsymbol{K}^t(y) \leq \ell(m))$, i.e., $\varphi(y)$ holds. This concludes the provable soundness in $\mathcal{T}$ and thus completes the proof. □

9.3 Typical Parameter Regimes

Similar to Section 4.4, the following two corollaries directly follow from the master theorem and Theorem 8.8. The proofs are omitted.

Corollary 9.4. *Let $\gamma \in (0,1)$ be a constant, $\mathcal{C}$ be a polynomial-time algorithm, $\mathcal{U} \in \mathsf{NTIME}[T]$ for $T(n) = \mathsf{poly}(n)$, and $\mathcal{T}$ be a sound extension of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Assume that $\mathsf{prBPP} = \mathsf{prP}$, and there is a proof complexity generator against $\mathcal{T}$ with polynomial security and $n^{o(1)}$ compression rate, then there is a polynomial $t(n)$ and a function $\ell(n) = n^{o(1)}$ such that the following holds.*

Suppose that for every $k \in \mathbb{N}$, there is a function $\lambda_0 : \mathbb{N} \to \mathbb{N}$ such that

$$\mathcal{T} \vdash (\mathcal{C}, \mathcal{U}, T, \gamma, t, \ell)\text{-}\mathsf{GSecure}[1, \lambda^k, \lambda^k, \lambda_0].$$

Then for every polynomial $n(\lambda)$, $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is polynomially secure.

Corollary 9.5. *Let $\gamma \in (0,1)$ be a constant and $S(\lambda), \mathcal{E}$ be PV-definable size bound such that $S(\lambda) = \lambda^{\omega(1)}$, $\mathcal{E}(\lambda) = \lambda^{\omega(1)}$. Let $\mathcal{C}$ be a polynomial-time algorithm, $\mathcal{U} \in \mathsf{NTIME}[T]$ for $T(n) = \mathsf{poly}(n)$, and $\mathcal{T}$ be a sound extension of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Assume that $\mathsf{prBPP} = \mathsf{prP}$, and there is a proof complexity generator against $\mathcal{T}$ with polynomial security and $n^{o(1)}$ compression rate, then there is a polynomial $t(n)$ and a function $\ell(n) = n^{o(1)}$ such that the following holds.*

Suppose that for every $k \in \mathbb{N}$, there is a function $\lambda_0 : \mathbb{N} \to \mathbb{N}$ such that

$$\mathcal{T} \vdash (\mathcal{C}, \mathcal{U}, T, \gamma, t, \ell)\text{-}\mathsf{GSecure}[S, \mathcal{E}, \lambda^k, \lambda_0].$$

Then for every polynomial $n(\lambda)$, $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, n)$ is $(S(\lambda), 2/\mathcal{E}(\lambda))$-secure.

Corollary 9.6. *Let $\mathcal{T}$ be a sound extension of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Assume that $\mathsf{prBPP} = \mathsf{prP}$, and there is a proof complexity generator against $\mathcal{T}$ with subexponential security and $o(n)$ compression rate, then for every constant $c > 0$ and every sufficiently small $\delta \in (0,1)$, there exists a polynomial $t(n)$ and a function $\ell(n) = o(n)$ such that the following holds.*

Let $\gamma \in (0,1)$ be any constants, $T(n) = 2^{n^\delta}$, $\mathcal{C}$ be a $\mathsf{poly}(T)$-time algorithm, and $\mathcal{U} \in \mathsf{DTIME}[T]$ (i.e. the verifier runs in time $T(n)$ and there is no witness). Let $\mathcal{U}_{\mathsf{pad}}$ be the padded language described in Section 9.1. Suppose that for $N(\lambda) = \lambda^\delta$ and any constants $k \in \mathbb{N}$, there is a function $\lambda_0 : \mathbb{N} \to \mathbb{N}$ such that

$$\mathcal{T} \vdash (\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)\text{-}\mathsf{GSecure}[1, \lambda^k, N, \lambda_0].$$

Then the security game $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, N)$ is polynomially secure.

10 Applications

In this section, we verify that all our previous constructions can be built on proof complexity generators against feasible cryptography, namely the extension of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" by axioms related to security of cryptographic primitives.

We assume readers have basic familiarity with the theory APC_1, its formal definitions and capability; see, e.g., [Jeř07a, HJLM26] for detailed expositions. Given reasonable experiences in working with formal theories, it becomes unnecessary (and infeasible) to write APC_1 proofs in very detail provided that the formalization of statements are robust. In this paper, we will provide detailed exposition on the *formalization of security definitions* of cryptographic schemes, which will be served as axioms for Assumption 1, but will only sketch the *formalization of security proofs*.

10.1 Cryptographic Primitives Used as Axioms

Utilizing the toolkits in [HJLM26], we can formalize the syntax and security of cryptographic primitives needed in the theory $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$".

10.1.1 Overview of Our Methodology

Before proceeding, we provide a general guideline on how we formalize the syntax and semantics of cryptographic primitives in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$".

The syntax of a cryptographic primitive is usually formalized in the language of PV, as demonstrated in a recent sequence of works [JJMP25, JKLM25] (see also [Mat25]). In particular, a probabilistic machine is formalized with a PV function (computing a deterministic polynomial-time function) that takes both its input and the random seed. Similarly, *perfect correctness* of cryptographic primitives can usually be formalized as a $\forall\Pi^b_0$-sentence (i.e. sentences with no existential quantifiers) in the language of PV_1, or alternatively an equation in the language of PV.

Following [HJLM26], we aim to formalize security definitions as universal sentences (i.e. $\forall\Pi^b_1$-sentences) in the language of APC_1 using either *indistinguishability* or *search games*. We formalize polynomial security as *a set of sentences* rather than a sentence. For instance, suppose that we will formalize that $\mathcal{H}_1$ and $\mathcal{H}_2$ are indistinguishability for any $\mathsf{poly}(\lambda)$ sized adversary, we will use the set of sentences where for every constant k, the sentences

$$\mathcal{H}_1 \approx_{\lambda^k,\lambda^{-k}} \mathcal{H}_2$$

is in the set. The intended meaning of the formalization is that every sentence in the set is true; in particular, when we say that the security of a primitive is used as axioms, we add every sentence in the set as an axiom.

We emphasize that our results hold for any natural and straightforward formalization, provided that it is straightforward and follows the general principles we mentioned here.

10.1.2 Fully Homomorphic Encryption

We will need (public-key) FHE with malicious gate correctness in constructions. Recall that the syntax of FHE is as follows.

- $\mathsf{Gen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$: a randomized algorithm generating keys.
- $\mathsf{Enc}_{\mathsf{pk}}(m) \to \mathsf{ct}$: a randomized algorithm the encrypts a string $m \in \{0,1\}^*$.
- $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}) \to m$: decrypts the ciphertext ct.
- $\mathsf{GateEval}_{\mathsf{pk}}(f, c_1, c_2) \to c^*$: A deterministic algorithm that takes the public key, the truth table of a *two input bit* function $f\colon \{0,1\} \times \{0,1\} \to \{0,1\}$, and two ciphertexts c_1, c_2. It outputs a ciphertext c^*.
- $\mathsf{Eval}_{\mathsf{pk}}(f, c_1, \ldots, c_n) \to c^*$. This is a deterministic algorithm defined by GateEval. It algorithm takes as input the public key pk, a circuit representing a function $f\colon \{0,1\}^n \to \{0,1\}^m$, along with n ciphertexts $c_1, \ldots, c_n$. It runs GateEval to evaluate the circuit gate-by-gate and outputs the ciphertexts c^* of the output of it.

Correctness. The FHE correctness states the (perfect) correctness of decryption and evaluation; both of which do not involve probability can be formalized as a $\forall\Pi^b_1$-sentence in PV_1 (or equivalently, a PV equation). Similar to [HJLM26], we will need malicious gate correctness. Specifically:

- (*Decryption*). It states that for every $(\mathsf{sk}, \mathsf{pk})$ generated by $\mathsf{Gen}(1^\lambda)$, every string m, the for any internal randomness of $\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}$, then $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(m)) = m$.
- (*Honest Evaluation Correctness*). It states that for any $(\mathsf{sk}, \mathsf{pk})$ generated by $\mathsf{Gen}(1^\lambda)$, strings $b_1, \ldots, b_n \in \{0,1\}$ and any circuit $f : \{0,1\}^n \to \{0,1\}^m$,

$$f(b_1, \ldots, b_n) = \mathsf{Dec}_{\mathsf{sk}}(\mathsf{Eval}_{\mathsf{pk}}(f, \mathsf{Enc}_{\mathsf{pk}}(b_1), \ldots, \mathsf{Enc}_{\mathsf{pk}}(b_n))).$$

- (*Malicious Gate Correctness*). For any choice of $(\mathsf{sk}, \mathsf{pk})$ given by $\mathsf{Gen}(1^\lambda)$, any function $f : \{0,1\}^2 \to \{0,1\}$, and any ciphertexts c_1, c_2, if $\mathsf{Dec}_{\mathsf{sk}}(c_1) = b_1$ and $\mathsf{Dec}_{\mathsf{sk}}(c_2) = b_2$, then

$$\mathsf{Dec}_{\mathsf{sk}}(i, \mathsf{GateEval}_{\mathsf{pk}}(f, c_1, c_2)) = f(b_1, b_2).$$

- (*Malicious Evaluation Correctness*). For any choice of $(\mathsf{sk}, \mathsf{pk})$ given by $\mathsf{Gen}(1^\lambda, 1^d)$ any ciphertexts $c_1, \ldots, c_n$ such that $b_i = \mathsf{Dec}_{\mathsf{sk}}(c_i)$, and any circuit $f\colon \{0,1\}^n \to \{0,1\}^m$, if we set $c = \mathsf{Eval}_{\mathsf{pk}}(f, c_1, \ldots, c_n)$ then $\mathsf{Dec}_{\mathsf{sk}}(c) = f(b_1, \ldots, b_n)$.

Security. The semantic security of the scheme states that the encryption of different strings are computationally indistinguishable given pk.

We first formalize a Π^b_0-formula $\mathsf{FHESecure}^{t,\varepsilon}_{\ell,\lambda}$ in the language of $\mathsf{APC}_1 +$"$\mathsf{prBPP} = \mathsf{prP}$" with variables $\lambda, \ell, \varepsilon^{-1}, t \in \mathsf{Log}$. It states that no adversary of size t distinguishes the encryption of two different strings of length at most ℓ with advantage at least ε. Specifically, let $\mathcal{H}_m$ be the distribution $(\mathsf{pk}, \mathsf{Enc}_{\mathsf{pk}}(m))$, where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^\lambda)$,

$$\mathsf{FHESecure}^{t,\varepsilon}_{\ell,\lambda} : \quad \forall m_1, m_2 \in \{0,1\}^{\leq \ell} \, (|m_1| = |m_2| \to \mathcal{H}_{m_1} \approx_{t,\varepsilon} \mathcal{H}_{m_2}).$$

Note that this is a Π^b_1-formula as the quantifier over m_1, m_2 is bounded.

Fix a function $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$. We define $\mathsf{FHESecure}[\lambda_0]$ as the set where **for every** $k \in \mathbb{N}$, it contains

$$\phi_k := \forall \lambda \in \mathsf{Log} \, (\lambda > \lambda_0(k) \to \mathsf{FHESecure}^{\lambda^k, \lambda^{-k}}_{\lambda^k, \lambda}).$$

Note that the quantifier in **bold** is in the meta-theory; each sentence ϕ_k that is in the set $\mathsf{FHESecure}$ is a $\forall\Pi^b_1$-sentence.

Definition 10.1 (Security Axioms of FHE)**.** Fix any secure FHE scheme $\mathsf{FHE}.*$, there exists a function λ_0 such that every sentence in $\mathsf{FHESecure}[\lambda_0]$ is true. If so, $\mathsf{FHESecure}[\lambda_0]$ is said to be the security axioms of this scheme $\mathsf{FHE}.*$.

10.1.3 Public Key Encryption

The formalization for PKE is identical to that for FHE but with no $\mathsf{GateEval}$ and Eval functions. Specifically, it involves $\mathsf{PKE}.(\mathsf{Gen}, \mathsf{Enc}_{\mathsf{pk}}, \mathsf{Dec}_{\mathsf{sk}})$. The correctness and security of a PKE scheme are defined in the same way as FHE:

- A $\forall\Pi^b_1$-formula $\mathsf{PKESecure}^{t,\varepsilon}_{\ell,\lambda}$ that formalizes the indistinguishability of the encryption for $m_1, m_2 \in \{0,1\}^{\leq \ell}$ such that $|m_1| = |m_2|$.
- For a function $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$, $\mathsf{PKESecure}[\lambda_0]$ is defined as the set of sentences

$$\phi_k := \forall \lambda \in \mathsf{Log} \, (\lambda > \lambda_0(k) \to \mathsf{PKESecure}^{\lambda^k, \lambda^{-k}}_{\lambda^k, \lambda}).$$

  for all constants $k \in \mathbb{N}$.

Definition 10.2 (Security Axioms of PKE)**.** Fix any secure PKE scheme $\mathsf{PKE}.*$, there exists a function λ_0 such that every sentence in $\mathsf{PKESecure}[\lambda_0]$ is true. If so, $\mathsf{PKESecure}[\lambda_0]$ is said to be the security axioms of this scheme $\mathsf{PKE}.*$.

10.1.4 Pseudorandom Generators

Recall that pseudorandom generator is a stretching function $G : \{0,1\}^n \to \{0,1\}^m$, where $m = m(n) > n$ is a polynomial, such that $G(\mathcal{U}_n)$ is computationally indistinguishable with $\mathcal{U}_m$.

Polynomially-secure PRG. We now formalize its security in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". For any fixed PRG G, assume that the output length is a PV function $m(n) : \mathsf{Log} \to \mathsf{Log}$, we define a Π^b_0-formula $\mathsf{PRGSecure}^{t,\varepsilon}_n$ with variables $n, t, \varepsilon^{-1} \in \mathsf{Log}$ to formalize $\mathcal{U}_{m(n)} \approx_{t,\varepsilon} G(\mathcal{U}_n)$. Let $n_0(k) : \mathbb{N} \to \mathbb{N}$ be any function, we define $\mathsf{PRGSecure}[n_0]$ as the set of sentences where for every $k \in \mathbb{N}$, it includes the $\forall\Pi^b_0$-sentence

$$\phi_k := \forall n \in \mathsf{Log}\ \left(n > n_0(k) \to \mathsf{PRGSecure}^{n^k, n^{-k}}_n\right).$$

Definition 10.3 (Security Axioms of PRG)**.** Fix any secure PRG G with stretch $m(n) > n$, there exists a function n_0 such that every sentence in $\mathsf{PRGSecure}[n_0]$ is true. If so, $\mathsf{PRGSecure}[n_0]$ is said to be the security axioms of G.

For technical reasons, we will need to formalize "slightly superpolynomial security" of PRGs. Namely, there is a superpolynomial function $s(\cdot)$ such that no adversary of size $s(n)$ can break it with advantage $1/s(n)$. Formally, let $S(\cdot)$ be a PV-formalizable size bound such that $S(n) = n^{\omega(1)}$, and $\lambda_0 \in \mathbb{N}$, we define $\mathsf{PRGSecure}[S, n_0]$ be the sentence

$$\forall n \in \mathsf{Log}\ \forall s, \varepsilon^{-1} \in \mathsf{Log}\ \left(n > n_0 \wedge s \le S(n) \wedge \varepsilon^{-1} \le S(n) \to \mathsf{PRGSecure}^{s,\varepsilon}_n\right).$$

Definition 10.4 (Security Axioms of Slightly Superpolynomially Secure PRGs.)**.** Let $S(\cdot)$ be a PV-formalizable size bound such that $S(n) = n^{\omega(1)}$. For any PRG G with stretch $m(n) > n$ that is $S(n)$-secure, there exists a constant $n_0 \in \mathbb{N}$ such that $\mathsf{PRGSecure}[S, n_0]$ is true. If so, $\mathsf{PRGSecure}[S, n_0]$ is said to be the security axiom of G.

In particular, we may use $\mathsf{PRGSecure}[n^{\omega(1)}, n_0]$ as the shorthand of $\mathsf{PRGSecure}[S, n_0]$ for some superpolynomial formalizable size bound S, if we do not need to refer to the specific function S.

Remark 15. Note that PRG can be construct from FHE by first showing that $\mathsf{FHE.Enc}$ is a OWF and then using the construction of PRG from OWF [HILL99]. However, it is unclear whether [HILL99] can be formalized in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". For this reason, we need to include the security axioms of PRG as axioms to formalize the security analysis of constructions in Part II.

10.1.5 Non-Interactive Witness-Indistinguishable Proof of Knowledge

Recall that for succinct MIFE, we will need NIWIPoK as a building block (see, Section 6.1). The syntax of NIWIPoK consists of the following probabilistic polynomial-time algorithms:

- $\mathsf{Gen}(1^\lambda, R) \to (\mathsf{crs}, \mathsf{td})$, where R is a relation circuit;
- $\mathsf{Prove}(\mathsf{crs}, x, w) \to \pi$: generates the NIWI proof π for $R(x, w) = 1$;
- $\mathsf{Ver}(\mathsf{crs}, x, \pi) \to \{0,1\}$: verifies the NIWI proof;

- $\mathsf{Ext}(\mathsf{td}, x, \pi) \to w$: extracts an NP witness w from the NIWI proof π.

The perfect soundness and completeness of the NIWIPoK scheme do not involve probability, and can be easily formalized as $\forall\Pi_1^b$-sentences in PV_1 (or equivalently, PV equations). It remains to formalize the *witness indistinguishability*, namely any polynomial-time adversary cannot distinguish between the NIWI proofs obtained from two different NP witnesses.

Witness Indistinguishability. As usual, we formalize witness indistinguishability against polynomial sized adversary as a set of sentences. Let $\mathsf{WI}_{\ell,\lambda}^{t,\varepsilon}$ be a formula with variables $\lambda, \ell, t, \varepsilon^{-1} \in \mathsf{Log}$, it states that for every circuit R of size at most ℓ, every x, w_0, w_1 such that $R(x, w_0) = R(x, w_1) = 1$, let $\mathcal{H}_i$ be the distribution

$$(\mathsf{crs}, x, \mathsf{Prove}(\mathsf{crs}, x, w_i))$$

where $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, R)$, then $\mathcal{H}_0 \approx_{t,\varepsilon^{-1}} \mathcal{H}_1$. Note that this is a Π_1^b-formula in the language of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$", where the universal quantifiers over R, x, w_0, w_1 are bounded as the size is bounded by $\ell \in \mathsf{Log}$.

Fix a function $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$, we define $\mathsf{WISecure}[\lambda_0]$ as the set where for every $k \in \mathbb{N}$, it contains the $\forall\Pi_1^b$-sentence in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$":

$$\phi_k := \forall \lambda \in \mathsf{Log}\ \left(\lambda > \lambda_0(k) \to \mathsf{WI}_{\lambda^k,\lambda}^{\lambda^k,\lambda^{-k}}\right)$$

Definition 10.5 (Security Axioms of NIWIPoK)**.** Fix any secure NIWI Proof of Knowledge scheme $\mathsf{NIWI}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Ext})$, there exists a function λ_0 such that every sentence in $\mathsf{WISecure}[\lambda_0]$ is true. If so, $\mathsf{WISecure}[\lambda_0]$ is said to be the security axioms of this scheme $\mathsf{NIWI}.*$.

10.1.6 Succinct Obfuscation with Extended Frege Proofs

Now we show how to formalize succinct obfuscation (constructed in [JJMP25]) in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". For reference, the syntax and semantics of succinct obfuscation is defined in Section 3.3.7.

Following the notation above, we use $M[\mathsf{pub}](\cdot)$ to denote a pair (M, pub), where M is parsed as a Turing machine and pub is parsed as a string.

Formalization of EF-equivalence. We start by explaining the formalization of EF equivalence in PV_1 (and thus also in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$").

Let $\rho, \tau, m \in \mathsf{Log}$. As mentioned in Definition 3.16, programs $\underline{M_0[\mathsf{pub}](\cdot)}$ and $\underline{M_1[\mathsf{pub}](\cdot)}$ are said (ρ, τ)-EF-equivalent up to input length m if **there exists** a machine $\mathcal{M}_\Pi$ of size ρ, such that *for every* $n < m$, $\mathcal{M}_\pi$ takes as input a length bound 1^n, along with M_0, M_1, pub and outputs an EF proof that $[\![M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x)]\!]_n$ in $\tau(n)$ time. Specifically:

- "(ρ, τ)-EF-equivalent" is a $\Sigma_1^b(\mathsf{PV})$-formula with variables M_0, M_1, pub and $\rho, m \in \mathsf{Log}$.
- The **existential quantifier** (in blue and bold) is a bounded existential quantifier, while the *universal quantifier* in red and italic is a sharply bounded quantifier as $m \in \mathsf{Log}$ (alternatively, it can be implemented as a PV function that enumerates $n \le m$ and verifies the condition).

Syntax and Succinctness. The syntax of EF-iO consists of a PPT machine $\mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$ as well as two functions $\rho(\lambda), \tau(\lambda, n) : \mathsf{Log} \to \mathsf{Log}$ that are polynomials. The input length bound n is given in binary, and $n \le 2^\lambda$. It is required to be fully succinct: the size of the obfuscated machine $\tilde{M}$ is

bounded by $\mathsf{poly}(\lambda, |M|)$, i.e., the size of $\tilde{M}$ does not grow with $|\mathsf{pub}|$ and n. However, we will not need the formalization of succinctness in the theory $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$".

Formalization of Correctness. The formalization of its correctness has been discussed in detail in [JJMP25] (see also [Mat25]). The correctness asserts that for every security parameter λ, machine M, input length $n \leq 2^\lambda$, and string pub, $M[\mathsf{pub}]$ and the obfuscated machine $\tilde{M}[\mathsf{pub}] = \mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$ agrees with each other on input length at most n. This can be formalized as a $\forall\Pi_1^b$-sentence in the language of PV_1, where $n, \lambda \in \mathsf{Log}$, or alternatively as a PV equation. (Note that the predicate $n \leq 2^\lambda$ with variables $n, \lambda \in \mathsf{Log}$ is definable in polynomial time.)

Formalization of Polynomial Security. We need both polynomial and subexponential secure EF-iO; we start with polynomial security. As usual, we will define security axioms as a set of sentences to formalize the existential quantification over a negligible function.

Fix a PV function $\mathsf{iO}_{\rho,\tau}$ and functions $\rho(\lambda) : \mathsf{Log} \to \mathsf{Log}$ and $\tau(\lambda, n) : \mathsf{Log} \times \mathsf{Log} \to \mathsf{Log}$ that are polynomials. We define a formula $\mathsf{EFIOSecure}^{t,\varepsilon}_{\lambda,p}$ with variables $t, \varepsilon^{-1}, \lambda, p \in \mathsf{Log}$ that formalizes the follwing statement: For every machines $M_0[\mathsf{pub}], M_1[\mathsf{pub}], \mathcal{M}_\Pi$ with split description of size at most p and $n \in \mathsf{Log}$, $n \leq p$, assume that

- $\mathsf{EQRT}(M_0[\mathsf{pub}], M_1[\mathsf{pub}]) = 1$, i.e., $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$ have the same clock, and
- $M_0[\mathsf{pub}]$ and $M_1[\mathsf{pub}]$ are $(\rho(\lambda), \tau(\lambda, n))$-equivalent on inputs up to length n, witnessed by the machine $\mathcal{M}_\Pi$ with description size at most ρ.

Then it follows that

$$\mathsf{iO}_{\rho,\tau}(1^\lambda, M_0[\mathsf{pub}], 1^n) \approx_{t,\varepsilon} \mathsf{iO}_{\rho,\tau}(1^\lambda, M_1[\mathsf{pub}], 1^n)$$

Note that this formula is a Π_1^b-formula in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". To see this, notice that:

- The quantifiers over $M_0[\mathsf{pub}], M_1[\mathsf{pub}], \mathcal{M}_\Pi, n$ are bounded as $|M_0[\mathsf{pub}]|, |M_1[\mathsf{pub}]|, n \leq p$ and $|\mathcal{M}_\Pi| \leq \rho(\lambda)$ for $\rho(\lambda), p \in \mathsf{Log}$.
- The second bullet is a Σ_1^b-formula, while it appears as the premise of an implication, and thus becomes a Π_1^b-formula after prenexification.

Let $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$ be any function, we define $\mathsf{PolyEFIOSecure}[\lambda_0]$ as the following set of sentences: For every $k \in \mathbb{N}$, it includes the $\forall\Pi_1^b$-sentence

$$\phi_k := \forall\lambda \in \mathsf{Log}\ \left(\lambda > \lambda_0(k) \to \mathsf{EFIOSecure}^{\lambda^k, \lambda^{-k}}_{\lambda,\lambda^k}\right).$$

Definition 10.6 (Polynomial Security Axioms of EFIO)**.** Fix any polynomially secure EF-iO for Turing machines with split description $\mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$, where ρ, τ are PV functions that are polynomials, there exists a function λ_0 such that every sentence in $\mathsf{PolyEFIOSecure}[\lambda_0]$ is true. If so, $\mathsf{PolyEFIOSecure}[\lambda_0]$ is said to be the security axioms of $\mathsf{iO}_{\rho,\tau}(\cdot,\cdot,\cdot)$.

Formalization of Subexponential Security. The subexponential security for $\mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$ can be formalized as one sentence rather than a set of sentences. Let $\delta \in (0, 1)$ and $\lambda_0 \in \mathbb{N}$ be constants, we define $\mathsf{ExpEFIOSecure}[\delta, \lambda_0]$ as the sentences:

$$\forall\lambda, t, \varepsilon^{-1} \in \mathsf{Log}\ \left(\lambda > \lambda_0 \wedge (t, \varepsilon^{-1} \leq 2^{\lambda^\delta}) \to \mathsf{EFIOSecure}^{t,\varepsilon}_{\lambda,\lambda^k}\right).$$

Note that this sentence is a $\forall\Pi_1^b$ formula as $t, \varepsilon^{-1} \leq 2^{\lambda^\delta}$ can be verified in $\mathsf{poly}(t, \varepsilon^{-1}, \lambda)$ time for $t, \varepsilon^{-1}, \lambda \in \mathsf{Log}$.

Definition 10.7 (Subexponential Security Axioms of EFIO)**.** Fix any subexponentially secure EF-iO for Turing machines with split description $\mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$, where ρ, τ are PV functions that are polynomials, there are constants $\lambda_0 \in \mathbb{N}$ and $\delta \in (0,1)$ such that $\mathsf{ExpEFIOSecure}[\delta, \lambda_0]$ is true. If so, $\mathsf{ExpEFIOSecure}[\delta, \lambda_0]$ is said to be the security axiom of $\mathsf{iO}_{\rho,\tau}(\cdot,\cdot,\cdot)$.

Formalization of Slightly Superpolynomial Security. Similar to the case for PRGs, we will need EF-iO that is secure against slightly superpolynomial sized adversaries. Let $S(\lambda)$ be a PV-formalizable size bound such that $S(\lambda) = \lambda^{\omega(1)}$, and $\lambda_0 \in \mathbb{N}$, we define $\mathsf{EFIOSecure}[S, \lambda_0]$ as the sentence

$$\forall \lambda, t, \varepsilon^{-1} \in \mathsf{Log}\ \left(\lambda > \lambda_0 \wedge (t, \varepsilon^{-1}) \leq S(\lambda) \to \mathsf{EFIOSecure}^{t,\varepsilon}_{\lambda,\lambda^k}\right).$$

Definition 10.8 (Superpolynomial Security Axioms of EFIO)**.** Fix any slightly superpolynomially secure secure EF-iO for Turing machines with split description $\mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$, where ρ, τ are PV functions that are polynomials, there are constants $\lambda_0 \in \mathbb{N}$ and PV-formalizable size bound $S(\lambda) = \lambda^{\omega(1)}$ such that $\mathsf{EFIOSecure}[S, \lambda_0]$ is true. If so, $\mathsf{EFIOSecure}[S, \lambda_0]$ is said to be the security axiom of $\mathsf{iO}_{\rho,\tau}(\cdot,\cdot,\cdot)$.

In particular, we may use $\mathsf{EFIOSecure}[\lambda^{\omega(1)}, \lambda_0]$ as the shorthand of $\mathsf{EFIOSecure}[S, \lambda_0]$ for some superpolynomial formalizable size bound S, if we do not need to refer to the specific function S.

10.1.7 Batch Argument for NP

Following [JKLV24, HJLM26], we will need Batch Argument (see Section 3.3.5 for syntax and semantics). In particular, we will use the same formalization in [HJLM26] in APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$"; we assume perfect correctness and extractability, so both statements can be formalized as a $\forall\Pi_1^b$-statement in PV_1. We will also formalize the CRS indistinguishability of BARG following [HJLM26]; for completeness, we review the formalization.

Recall that the syntax of BARG consists of functions $\mathsf{BARG}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{TGen}, \mathsf{Extract})$. We first formalize a Π_1^b-formula $\mathsf{BARGSecure}^{t,\varepsilon}_{\lambda,m,s,L}$ in the language of APC_1 + "$\mathsf{prBPP} = \mathsf{prP}$" with variables $\lambda, m, s, L, t, \varepsilon^{-1} \in \mathsf{Log}$. Let $\mathcal{D}$ be the distribution $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^m, 1^s, 1^L)$, and $\mathcal{D}_\mathcal{S}$ be the distribution crs for $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^m, 1^s, 1^L, \mathcal{S})$, we define

$$\mathsf{BARGSecure}^{t,\varepsilon}_{\lambda,m,s,L}: \quad \forall \mathcal{S} \subseteq [m]\ (|\mathcal{S}| \leq L \to \mathcal{D} \approx_{t,\varepsilon} \mathcal{D}_\mathcal{S}).$$

Fix a function $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$, we define $\mathsf{BARGSecure}[\lambda_0]$ as the set where for every $k \in \mathbb{N}$, it contains the $\forall\Pi_1^b$-sentence

$$\phi_k := \forall \lambda \in \mathsf{Log}\ \left(\lambda > \lambda_0(k) \to \mathsf{BARGSecure}^{\lambda^k, \lambda^{-k}}_{\lambda,\lambda^k,\lambda^k,\lambda^k}\right).$$

Definition 10.9 (Security Axioms of BARG)**.** Fix any secure $\mathsf{BARG}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{TGen}, \mathsf{Ext})$, there exists a function λ_0 such that every sentence in $\mathsf{BARGSecure}[\lambda_0]$ is true. If so, $\mathsf{BARGSecure}[\lambda_0]$ is said to be the security axioms of this scheme $\mathsf{BARG}.*$.

10.2 Cryptographic Primitives Constructed from Axioms

We will also need several cryptographic primitives to formalize our security analysis in Part II, including PKE, extractable commitment, SSB Hash, and description-succinct iO. These primitives can be constructed

from FHE or PRGs[30], and the security of these primitives can be proved from the security axioms of FHE. For completeness, we provide the formalization of these primitives and sketch the construction from FHE.

10.2.1 Extractable Commitment

The MIFE construction requires an extractable commitment scheme. As mentioned in Section 3.3.1, it is well-known to be constructible from any PKE scheme (and thus also from any FHE scheme). We now formalize extractable commitment in the language of APC_1, and show that its security can be proved from that of FHE in APC_1.

Syntax and Binding Property. The syntax of an extractable commitment involves the following polynomial-time algorithms:

- $\mathsf{Gen}(1^\lambda) \to (\mathsf{crs}, \mathsf{td})$: a randomized algorithm generating CRS and extraction trapdoor.
- $\mathsf{Com}(\mathsf{crs}, b; \sigma) \to Z$: produces the commitment Z of $b \in \{0,1\}$ with the randomness σ.
- $\mathsf{Ext}(\mathsf{td}, Z) \to b$: outputs the bit commited to Z.

As mentioned above, we formalize these randomized algorithms as PV functions that take an additional input for random seeds.

The extractability of the scheme can be formalized as a $\forall\Pi_1^b$-sentence in PV_1 (or equivalently, a PV equation). Specifically, it can be formalized as: for every $\lambda \in \mathbb{N}$, any internal randomness r of Gen, any Z, σ and $b \in \{0,1\}$, if $(\mathsf{crs}, \mathsf{td})$ is the output of $\mathsf{Gen}(1^\lambda; r)$ and Z is the output of $\mathsf{Com}(\mathsf{crs}, b; \sigma) = Z$, then $\mathsf{Ext}(\mathsf{td}, Z) = b$. It remains to formalize the computational hiding property. Fix any extractable commitment scheme, and let $k \in \mathbb{N}$ be a constant.

Hiding Property. We first formalize a Π_0^b-formula $\mathsf{ComHiding}_\lambda^{t,\varepsilon}$ in the language of $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" with variables $\lambda, \varepsilon^{-1}, t \in \mathsf{Log}$. It state that it is computationally hiding against t size adversary with advantage ε on the security parameter λ. That is, let $\mathcal{H}_b$ be the distribution that, over the internal randomness r, σ of Gen and Com, outputs

$$(\mathsf{crs}, Z_b)$$

where $Z_b := \mathsf{Com}(\mathsf{crs}, b; \sigma)$, and $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Gen}(1^\lambda, r)$, we define

$$\mathsf{ComHiding}_\lambda^{t,\varepsilon}: \quad \mathcal{H}_0 \approx_{t,\varepsilon} \mathcal{H}_1.$$

Fix a function $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$. We define $\mathsf{ComHiding}[\lambda_0]$ as the set where **for every** $k \in \mathbb{N}$, it contains the sentence

$$\phi_k := \forall \lambda \in \mathsf{Log}\left(\lambda > \lambda_0(k) \to \mathsf{ComHiding}_\lambda^{\lambda^k, \lambda^{-k}}\right).$$

Note that the quantifier in **bold** is in the meta-theory; each sentence ϕ_k that is in the set ComHiding is a $\forall\Pi_0^b$-sentence.

Definition 10.10 (Security Axioms of Extractable Commitment)**.** Fix any secure extractable commitment scheme $\mathsf{Com}.(\mathsf{Gen}, \mathsf{Com}, \mathsf{Ext})$, there exists a function λ_0 such that every sentence in $\mathsf{ComHiding}[\lambda_0]$ is true. If so, $\mathsf{ComHiding}[\lambda_0]$ is said to be the security axioms of this scheme $\mathsf{Com}.*$.

[30] Indeed, these primitives can be constructed from PKE or even weaker primitives; nevertheless, as we will include FHE axioms anyway, we present constructions directly from FHE.

Construction from FHE. It is well-known that the commitment can be constructed from PKE (and thus also FHE). Recall that the construction works as follows:

- Let $\mathsf{PKE.Gen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$, $\mathsf{PKE.Enc}_{\mathsf{pk}}(b) \to c$, $\mathsf{PKE.Dec}_{\mathsf{sk}}(c) \to b$ be the key generation, encryption, and decryption algorithms for the FHE scheme.
- $\mathsf{Com.Gen}(1^\lambda)$ runs $\mathsf{PKE.Gen}(1^\lambda)$ and outputs $\mathsf{crs} := \mathsf{pk}$ and $\mathsf{td} := \mathsf{sk}$.
- $\mathsf{Com.Com}(b; r) := \mathsf{Enc}_{\mathsf{pk}}(b; r)$.
- $\mathsf{Com.Ext}(Z, \mathsf{td}) := \mathsf{Dec}_{\mathsf{td}}(z)$.

Note that the extractability is exactly the correctness of the PKE scheme, so provided that the PKE scheme has PV proof of correctness, the commitment scheme will have PV proof of extractability. Moreover, the hiding property is exactly the security of PKE, and is thus provable in $\mathsf{APC}_1 +$ "prBPP = prP". Therefore, given the security of FHE (i.e. the set FHESecure of sentences) as axioms, there exists a function $\lambda_0(\cdot)$ such that every sentence is ComHiding is provable in $\mathsf{APC}_1 +$ "prBPP = prP".

Claim 10.11. *If there is a secure PKE scheme* $\mathsf{PKE}.*$*, there is also a secure extractable commitment scheme* $\mathsf{Com}.*$*. Moreover, let* $\mathsf{PKESecure}[\lambda_0]$ *be the security axioms of* $\mathsf{PKE}.*$*, there exists a function* $\lambda_0'(k)$ *and security axioms* $\mathsf{ComHiding}[\lambda_0']$ *of* $\mathsf{Com}.*$ *such that*

$$\mathsf{APC}_1 + \mathsf{PKESecure}[\lambda_0] \vdash \mathsf{ComHiding}[\lambda_0'].$$

Furthermore, if $\mathsf{PKE}.*$ *admits* PV *proof of correctness, then* $\mathsf{Com}.*$ *admits* PV *proof of extractability. The same result holds when* $\mathsf{PKE}.*$ *is replaced by a secure FHE scheme* $\mathsf{FHE}.*$*.*

10.2.2 Somewhere Extractable Hash Function

We will need somewhere extractable hash function, or SEH (see Section 3.3.3). We follow the formalization in [HJLM26, Section 9]. Note that the correctness (i.e. opening completeness and extraction) can be formalized as $\forall\Pi_1^b$-statements in PV_1, and the security (i.e. key indistinguishability) can be formalized as a $\forall\Pi_1^b$-statement in $\mathsf{APC}_1 +$ "prBPP = prP". For completeness, we sketch the formalization of its security and the construction from FHE.

Recall that the syntax of SEH consists of functions $\mathsf{SEHash.}(\mathsf{Gen}, \mathsf{TGen}, \mathsf{Hash}, \mathsf{Open}, \mathsf{Ver}, \mathsf{Ext})$. We formalize a Π_1^b-formula $\mathsf{SEHSecure}_{\lambda,N}^{t,\varepsilon}$ with variables $t, \varepsilon^{-1}, \lambda, N \in \mathsf{Log}$. Let $\mathcal{D}$ be the distribution of hk where $\mathsf{hk} \leftarrow \mathsf{Gen}(1^\lambda, 1^N)$, and $\mathcal{D}_i$ be the distribution of hk where $(\mathsf{hk}, \mathsf{td}) \leftarrow \mathsf{TGen}(1^\lambda, 1^N, i)$, we define

$$\mathsf{SEHSecure}_{\lambda,N}^{t,\varepsilon}: \quad \forall i \in [N]\ \mathcal{D} \approx_{t,\varepsilon} \mathcal{D}_i.$$

Let $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$ be any function, we define $\mathsf{SEHSecure}[\lambda_0]$ as the following set of sentences: for every constant $k \in \mathbb{N}$, it includes the $\forall\Pi_1^b$-sentence

$$\phi_k := \forall\lambda \in \mathsf{Log}\ \left(\lambda > \lambda_0(k) \to \mathsf{SEHSecure}_{\lambda,\lambda^k}^{\lambda^k,\lambda^{-k}}\right)$$

in the language of $\mathsf{APC}_1 +$ "prBPP = prP".

Definition 10.12 (Security Axioms of SSB Hash)**.** Fix any SSB Hash scheme $\mathsf{SEHash}.*$, there exists a function λ_0 such that every sentence in $\mathsf{SEHSecure}[\lambda_0]$ is true. If so, $\mathsf{SEHSecure}[\lambda_0]$ is said to be the security axioms of this scheme $\mathsf{SEHash}.*$.

Theorem 10.13 ([HJLM26, Section 9]). *If there is a secure FHE scheme* $\mathsf{FHE}.*$*, there is also a secure SEH scheme* $\mathsf{SEHash}.*$*. Moreover, let* $\mathsf{FHESecure}[\lambda_0]$ *be the security axioms of* $\mathsf{FHE}.*$*, there exists a function* $\lambda_0'(k)$ *and security axioms* $\mathsf{SEHSecure}[\lambda_0']$ *such that*

$$\mathsf{APC}_1 + \mathsf{FHESecure}[\lambda_0] \vdash \mathsf{SEHSecure}[\lambda_0'].$$

Furthermore, if $\mathsf{FHE}.*$ *admits* PV *proof of correctness, then* $\mathsf{SEHash}.*$ *admits* PV *proof of correctness (i.e. completeness and extractability).*

10.2.3 Description-Succinct Indistinguishability Obfuscation

A key primitive in the construction of rate-1 indistinguishability obfuscation is a description-succinct indistinguishability obfuscation for Turing machines with split description (see Section 3.3.7).

Recall that the syntax of a description-succinct iO is a function $\mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n)$, where $M[\mathsf{pub}]$ is a clocked Turing machine with split description (see Definition 3.13). It should satisfy the correctness, description-succinctness, as well as polynomial security for all pairs of equivalent machines.

Formalization of Security. The security of description-succinct iO will be formalized in a slightly different way: we will use $n \in \mathsf{LogLog}$, i.e., 1^{2^n} is feasible, so we can formalize it as a $\forall\Pi_1^b$-sentence in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$".

Let $\mathsf{IOSecure}^{t,\varepsilon}_{\lambda,p,m}$ be the following formula with variables $\lambda, t, \varepsilon^{-1}, p \in \mathsf{Log}$ and $m \in \mathsf{LogLog}$: For every machine $M_0[\mathsf{pub}], M_1[\mathsf{pub}]$ of size at most p, assume that $\mathsf{EQRT}(M_0[\mathsf{pub}], M_1[\mathsf{pub}]) = 1$ and for every $n \le m$,

$$\forall x \in \{0,1\}^n \; M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x) \tag{10.1}$$

then it follows that

$$\mathsf{iO}(1^\lambda, M_0[\mathsf{pub}], 1^n) \approx_{t,\varepsilon} \mathsf{iO}(1^\lambda, M_1[\mathsf{pub}], 1^n).$$

Note that this is a Π_1^n-formula; this is because Equation (10.1) is a sharply bounded quantifier for $n \le m \in \mathsf{LogLog}$, or equivalently it can be formalized as a PV function that enumerates all possible $x \in \{0,1\}^n$. This is impossible for the more standard $m \in \mathsf{Log}$ formalization.

Let $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$, we define $\mathsf{PolyIOSecure}[\lambda_0]$ as the set of sentences that for every k, it includes

$$\phi_k : \quad \forall \lambda \in \mathsf{Log}\, \forall m \in \mathsf{LogLog} \left(\lambda > \lambda_0(k) \wedge m \le \lambda^k \to \mathsf{IOSecure}^{\lambda^k, \lambda^{-k}}_{\lambda, \lambda^k, m}\right).$$

Note that this is a $\forall\Pi_1^n$-formula in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". It is worth noting that we do not require $\lambda \in \mathsf{LogLog}$: when we instantiate with $m = \lambda^\varepsilon$ for a small constant $\varepsilon < 1$, it could be that $m \in \mathsf{LogLog}$ but $\lambda \notin \mathsf{LogLog}$ as $2^\lambda = 2^{m^{1/\varepsilon}} \gg \mathsf{poly}(2^m)$.

Construction from EF-iO. It is proved in [JJMP25] that for sufficiently large polynomials ρ, τ, a fully-succinct subexponentially-secure EF-iO scheme $\mathsf{iO}_{\rho,\tau}$ is indeed a description-succinct polynomiall-secure iO (see also Theorem 3.20). For our purpose, we will need to formalize its security in the theory $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Formally:

Theorem 10.14 (Theorem 3.20, in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$"). *The following is true for some* PV *functions* ρ, τ *that are polynomials. If* $\mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$ *is a fully-succinct subexponentially secure* EF*-iO, then for a constant* $d \in \mathbb{N}$*,*

$$\mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n) := \mathsf{iO}_{\rho,\tau}(1^{\lambda+n^d}, M[\mathsf{pub}], 1^n)$$

is a description-succinct polynomially secure iO. Moreover, let $\mathsf{ExpEFIOSecure}[\delta, \lambda_0]$ *be the security axiom of* $\mathsf{iO}_{\rho,\tau}$ *as an* EF*-iO scheme, there exists a function* $\lambda_0'(k) : \mathbb{N} \to \mathbb{N}$ *such that*

$$\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{ExpEFIOSecure}[\delta, \lambda_0] \vdash \mathsf{PolyIOSecure}[\lambda_0'],$$

where $\mathsf{PolyIOSecure}[\lambda_0']$ *is the security axioms of* $\mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n)$.

This is proved by formalizing the proof of Theorem 3.20 in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$; specifically, we use a hybrid argument over intermediate machines $N_0, N_1, \ldots, N_{2^{n+1}-1}$ where $N_0 = M_0$ and $N_{2^{n+1}-1} = M_1$, and for each hybrid, we use the security of EF-iO. This proof can be formalized as $n \in \mathsf{LogLog}$ and thus it is feasible to construct all 2^n hybrids. The details are deferred to Section B.2.

10.3 Formalization of Security Analysis

To apply our master theorem (see Theorem 9.3), we need to formalize the security proof of cryptographic constructions (in the setting where (x, w) in the security game is compressible) in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ with addition axioms for cryptographic primitives. This requires a case-by-case examination on our results in Part II.

Fortunately, the formalization of all our results are relatively straightforward, with the tools developed in [HJLM26]. In this subsection, we will first revisit the general framework and available tools, and then provide a case-by-case guideline on formalizing results in Part II.

10.3.1 Overview of Our Methodology

Our main tools to formalize cryptographic proofs are the meta-theorems developed in [HJLM26, Section 7]. Most importantly, we will use the following inner theorem of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ related to the indistinguishability of distributions.

- (*Hybrid Argument*). $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ proves that for distributions $\mathcal{H}_0, \mathcal{H}_1, \ldots, \mathcal{H}_n$ such that $\mathcal{H}_i \approx_{t,\varepsilon_i} \mathcal{H}_{i+1}$ for every $i < n$, where $t, \varepsilon_i^{-1} \in \mathsf{Log}$, then $\mathcal{H}_0 \approx_{t,\varepsilon} \mathcal{H}_n$ for $\varepsilon := \sum_{i=0}^{n-1} \varepsilon_i$ [HJLM26, Lemma 4.15]. This lemma serves as the skeleton of all our security proofs. For all our applications, we formalize security as the indistinguishability of two distributions, and use a hybrid argument.
- (*Reduction Lemma*). Suppose that $\mathcal{H}_1, \mathcal{H}_2$ be distributions over $\{0,1\}^n$, and $\mathcal{A}(x, r) : \{0,1\}^n \times \{0,1\}^{m_1} \to \{0,1\}^{m_2}$ be a circuit of size at most $t', n, t, t', m_1, m_2, \varepsilon^{-1} \in \mathsf{Log}$, if $\mathcal{H}_1 \approx_{t',\varepsilon} \mathcal{H}_2$, then $\mathcal{A}(\mathcal{H}_1, \mathcal{U}) \approx_{t,\varepsilon} \mathcal{A}(\mathcal{H}_2, \mathcal{U})$. This lemma is the key to utilize axioms for cryptographic security.
- (*Isomorphic Distributions*). $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ proves that for $n, m \in \mathsf{Log}$ and distributions $\mathcal{H}_1, \mathcal{H}_2$ over $\{0,1\}^m$ that are sampled by circuits $C_1, C_2 : \{0,1\}^n \to \{0,1\}^m$, if there are functions $f, g : \{0,1\}^n \to \{0,1\}^n$ such that
  - $C_1(f(x)) = C_2(x)$ for every $x \in \{0,1\}^n$, and
  - $f(g(x)) = x$ for every $x \in \{0,1\}^n$,

  then $\mathcal{H}_1 \approx_{t,\varepsilon} \mathcal{H}_2$ for every $t, \varepsilon^{-1} \in \mathsf{Log}$ [HJLM26, Lemma 4.14].

  In general, it is infeasible to determine whether two distributions are statistically identical, as it requires an exhaustive enumeration of all seeds. This lemma summarizes the most common approach to argue two distributions are statistically identical, and is sufficient for our applications. For instance, it can be used to prove that the distributions x and $x \oplus m$ (for $x \leftarrow \mathcal{H}$ and a fixed string m) are indistinguishable.

The key difference between feasible and infeasible mathematics is that the encoding of mathematical objects matters: By Parikh's theorem [Par71], $\forall x\ \exists y\ |y| = x$ is unprovable in bounded theories as the length of y is superpolynomial in the length of x. As a result, when we utilize the lemmas above, it is crucial to verify that the encoding matches requirements.

For instance, to use the lemma for hybrid argument [HJLM26, Lemma 4.15], it is crucial to verify that the number of hybrid $n \in \mathsf{Log}$, i.e., it is in unary encoding. Similarly, to use axioms PolyIOSecure for the security of description-succinct iO (see Section 10.2.3), we have to verify that the input length bound for machines $m \in \mathsf{LogLog}$.

10.3.2 From Indistinguishability to Indistinguishability Game

One technical problem we will face is that in our applications, it is be easier to define security (under compression) as the indistinguishability of two distributions, while to apply Theorem 9.3, we need to prove its security as a search game (see Definition 9.2) in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. We verify that the correctness of the standard conversion of indistinguishability security to indistinguishability game can be formalized in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

We first define indistinguishability security under compression in the language of $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Recall that for a language $\mathcal{U} \in \mathsf{NTIME}[T]$ for a formalizable size bound T, we define $\mathcal{U}_{\mathsf{pad}} \in \mathsf{NP}$ be the padded language (see Section 9.1). Falsifiable indistinguishability security of cryptographic primitives is defined as a tuple $(G_0, G_1, \mathcal{U}_{\mathsf{pad}})$, where $G_0(1^\tau, x, w, \cdot), G_1(1^\tau, x, w, \cdot) : \{0,1\}^{r_1} \to \{0,1\}^{r_2}$ are polynomial time algorithms that takes an instance x, a witness w for $x \in \mathcal{U}_{\mathsf{pad}}$, a time padding 1^τ, and an additional random seeds $\mathsf{sd} \in \{0,1\}^{r_1}$. It is said to be (s, ε)-secure on input length n, if

- for every $x \in \{0,1\}^n$, witness w for $x \in \mathcal{U}$, and $\tau \geq T(n)$, $G_0(1^\tau, x, w, \mathcal{U}_{r_1})$ and $G_1(1^\tau, x, w, \mathcal{U}_{r_1})$ are (s, ε)-indistinguishable.

Using the formalization of indistinguishability $\mathcal{H}_1 \approx_{s,\varepsilon} \mathcal{H}_2$ formalized in [HJLM26], this can be formalized in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

Definition 10.15. Let $t(\cdot), \ell(\cdot) : \mathsf{Log} \to \mathsf{Log}$ be PV functions, where t is a polynomial. Let $G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T$ be PV functions mentioned above, $n, s, \tau, \varepsilon^{-1} \in \mathsf{Log}$.

We define $(G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T, t, \ell)$-$\mathsf{IndSecure}^{s,\varepsilon}_{n,\tau}$ as the following formula (with variables $n, s, \tau, \varepsilon^{-1} \in \mathsf{Log}$): Suppose that $\tau \geq T(n)$. Let x, w be any strings such that $\mathcal{U}_{\mathsf{pad}}(x, 1^\tau, w) = 1$, let $\mathcal{H}_i$ be the distribution defined by $G_0(1^\tau, x, w, \cdot)$, then if $\boldsymbol{K}^t(x \| w) \leq \ell(|x| + |w|)$, $\mathcal{H}_0 \approx_{s,\varepsilon} \mathcal{H}_1$.

Definition 10.16 (Indistinguishability security under compression, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$)**.** Fix PV functions $G_0, G_1, \mathcal{U}, \mathcal{U}_{\mathsf{pad}}, t, \ell$ and size bound T as mentioned above. Let $S(\lambda), \mathcal{E}(\lambda), N(\lambda) = \lambda^{\Omega(1)}$ be formalizable size bounds, and $\lambda_0(\cdot) : \mathbb{N} \to \mathbb{N}$ be a function (not necessarily in PV). We define $(G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T, t, \ell)$-$\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0]$ as the set of the following sentence for each $k \in \mathbb{N}$:

For every $\lambda, n, \tau, s, \varepsilon^{-1} \in \mathsf{Log}$, satisfying that $\lambda > \lambda_0(k)$, $n \leq N(\lambda)$, $s \leq S(\lambda) \cdot \lambda^k$, $\varepsilon^{-1} \leq \mathcal{E}(\lambda)$, $\tau \geq T(n)$, we have

$$(G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T, t, \ell)\text{-}\mathsf{IndSecure}^{s,\varepsilon}_{n,\tau}.$$

Definition 10.17 (indistinguishability game)**.** Let $(G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T)$ be functions that formalize indistinguishability security, its *indistinguishability game* is defined as the following search game (following Definition 4.8).

- $\mathcal{C}(1^\tau, x, w, A; r)$: The random seed r consists a bit $b \in \{0,1\}$ and a seed sd for G_0 or G_1, it computes $m \leftarrow G_b(1^\tau, x, w, \mathsf{sd})$, and accepts if and only if $A(m) = b$.

- The threshold $\gamma := 1/2$.

It is well-known that in the standard model, the indistinguishability security holds imply that the indistinguishability game is secure. The following lemma shows that the fact can be formalized in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$".

Lemma 10.18. *Let $G_0, G_1, \mathcal{U}, \mathcal{U}_{\mathsf{pad}}, T$ be functions that define indistinguishability security, $t(\cdot), \ell(\cdot) : \mathsf{Log} \to \mathsf{Log}$ be PV functions, and $\mathcal{C}$ be the challenger of its indistinguishability game, $\gamma = 1/2$. Let $S(\lambda), \mathcal{E}(\lambda), N(\lambda) = \lambda^{\Omega(1)}$ be formalizable size bounds. Then for every function $\lambda_0(\cdot) : \mathbb{N} \to \mathbb{N}$, there exists a function $\lambda_0'(\cdot) : \mathbb{N} \to \mathbb{N}$ such that*

$$\begin{aligned}&\mathsf{APC}_1 + \text{"}\mathsf{prBPP} = \mathsf{prP}\text{"} + (G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T, t, \ell)\text{-}\mathsf{IndSecure}[S, \mathcal{E}, N, \lambda_0] \\ &\quad \vdash (\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)\text{-}\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0'].\end{aligned}$$

The proof of the lemma is deferred to Section B.3. With this lemma, it suffices to formalize security with indistinguishability to apply Theorem 9.3.

10.3.3 Multi-Input Functional Encryption

Recall that in Section 6, we construct very-selectively secure succinct EF-MIFE from the strong proof complexity generator assumption. Now we prove security of the same construction under proof complexity generators against feasible cryptography. Formally:

Theorem 10.19. *Suppose that there exist the following cryptographic primitives:*

- *A PKE scheme $\mathsf{PKE}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ that has PV-provable correctness.*
- *A NIWIPoK scheme $\mathsf{NIWI}.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Ext})$ that is PV-provably sound and relation-succinct.*
- *A (ρ, τ)-EF-iO for programs with split description (see Definition 3.18), where $\rho(\lambda) = \mathsf{poly}(\lambda)$ and $\tau(t, \lambda) = \mathsf{poly}(t, \lambda)$ are some fixed functions.*

Let $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$ be a function such that $\mathsf{PKESecure}[\lambda_0]$, $\mathsf{WISecure}[\lambda_0]$, and $\mathsf{PolyEFIOSecure}[\lambda_0]$ be the security axioms of the schemes above. Assume that there is a polynomially secure proof complexity generator against $\mathcal{T}$ provable soundness for

$$\mathcal{T} := \mathsf{APC}_1 + \text{"}\mathsf{prBPP} = \mathsf{prP}\text{"} + \mathsf{PKESecure}[\lambda_0] + \mathsf{WISecure}[\lambda_0] + \mathsf{PolyEFIOSecure}[\lambda_0]$$

with compression rate $\ell(n) = n^{o(1)}$.

Then there exist a very-selectively secure EF-MIFE scheme with $\mathsf{poly}(\lambda, L)$-size public parameters, encryption keys, and ciphertexts, and $|f| + \mathsf{poly}(\lambda, L)$-size decryption keys, where $|f|$ is the size of the function description, and L is the per-slot input length.

The theorem is proved by applying the master theorem (specifically, Corollary 9.4), for which we need to formalize the security proof under t-time ℓ-compression (i.e. Theorem 6.7) in the feasible cryptography theory $\mathcal{T}$.

Constructions. We use the same construction in Theorem 6.7, namely Construction 2. The construction utilizes PKE, NIWIPoK, EF-iO as well as an extractable commitment scheme. The first three primitives are given in assumption (with security axioms available in $\mathcal{T}$), and the last primitive can be constructed from PKE. Moreover, by Claim 10.11, the security axioms of the commitment scheme is provable from $\mathcal{T}$. The efficiency of the scheme has been verified in Section 6.

Formalization of Security Game. Recall that the security of MIFE is defined as an IND-security game. Fix a construction of EF-MIFE. As observed in proof of Theorem 6.7, we can also formalize it using indistinguishability-based security, as follows:

- Fix $t(n) = \mathsf{poly}(n)$ and $\ell(n) = n^{o(1)}$ be the parameters of the proof complexity generators, where $t, \ell : \mathsf{Log} \to \mathsf{Log}$ are PV functions.
- We define $\mathcal{U} \in \mathsf{NP}$ as the language of valid instances (corresponding to the challenges ch in IND-security definition), where $x \in \mathcal{U}$ if it can be parsed as a pair $(\mathsf{ch}, 1^p)$, where ch is a MIFE challenge

$$\mathsf{ch} = \left\{ \{f_q \in \mathcal{F}\}_{q\in[Q]}, \mathcal{I} \subseteq [n], \{\mathbf{x}_{i,q}^{(0)}, \mathbf{x}_{i,q}^{(1)} \in \{0,1\}^L\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]} \right\},$$

  and it is an *admissible* challenge with a size p EF proof π of admissibility; see Definition 3.23 for definitions. The EF proof π is the NP witness for $x \in \mathcal{U}$.
- Let $T(n) = \mathsf{poly}(n)$ such that $\mathcal{U} \in \mathsf{NTIME}[T]$, and the NP witness length is $O(n)$.
- Let $G_b(1^\kappa, x, \pi, \mathsf{sd})$ be the following algorithms for $b \in \{0,1\}$:
  - Let $(\mathsf{msk}, \mathsf{ek}_1, \ldots, \mathsf{ek}_n) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, 1^L, 1^Q, 1^p)$.
  - It computes decryption keys $\mathsf{sk}_{f_q} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f_q)$ for every $q \in [Q]$.
  - It computes ciphertexts $\mathsf{ct}_{i,q} \leftarrow \mathsf{Enc}(\mathsf{ek}_i, \mathbf{x}_{i,q}^{(b)})$ for every $i \in [n] \setminus \mathcal{I}, q \in [Q]$.
  - It outputs $(\{\mathsf{ek}_i\}_{i\in\mathcal{I}}, \{\mathsf{sk}_{f_q}\}_{q\in[Q]}, \{\mathsf{ct}_{i,q}\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]})$.

  (The time padding 1^κ used to deal with superpolynomial runtime is ignored as $\mathcal{U} \in \mathsf{NP}$.)

Let $\mathcal{C}$ be the challenger of the indistinguishability game derived from indistinguishability-based security $(G_0, G_1, \mathcal{U}, T)$; see Section 10.3.2. It can be observed that the formalization is correct:

Claim 10.20. *For any EF-MIFE scheme, it is very-selectively secure if and only if for every polyomial $N(\lambda)$, the search game $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, 1/2, N)$ is polynomially secure (see Definition 4.8).*

Formalization of Security Proof. Next, we show that the security proof (i.e. Theorem 6.7) can be formalized in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Formally:

Lemma 10.21. *Let $\mathcal{T}$ be the theory in Theorem 10.19, $t, \ell : \mathsf{Log} \to \mathsf{Log}$ be PV functions, $t(n) = \mathsf{poly}(n)$, $\ell(n) = n^{o(1)}$. Fix Construction 2 for EF-MIFE using PKE, NIWIPoK, and (ρ, τ)-EF-iO specified in Theorem 10.19. Then for all polynomials $N(\lambda), S(\lambda), \mathcal{E}(\lambda) : \mathsf{Log} \to \mathsf{Log}$, there exists a function $\lambda_0 : \mathbb{N} \to \mathbb{N}$ such that*

$$\mathcal{T} \vdash (G_0, G_1, \mathcal{U}, T, t, \ell)\text{-}\mathsf{IndSecure}[S, \mathcal{E}, N, \lambda_0].$$

Proof Sketch. The proof closely follows the proof of Theorem 6.7, so we only sketch the proof and highlight key points in formalization.

To prove $(G_0, G_1, \mathcal{U}, T, t, \ell)$-$\mathsf{IndSecure}[S, \mathcal{E}, N, \lambda_0]$, we need to prove that for every $k \in \mathbb{N}$, there exists λ_0' such that the following statement is provable in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$".

We argue in $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". For every $\lambda, n, \kappa, s, \varepsilon^{-1} \in \mathsf{Log}$ satisfying that $\lambda > \lambda_0'$, $n \le N(\lambda)$, $s \le S(\lambda) \cdot \lambda^k$, $\varepsilon^{-1} \le \mathcal{E}(\lambda)$, $\kappa \ge T(n)$, let $x = (\mathsf{ch}, 1^p) \in \mathcal{U}$ and π be its witness, suppose that

- $\boldsymbol{K}^t(x\|\pi) \le \ell(|x| + |\pi|)$

then it follows that

- let $\boldsymbol{D}_b$ be the distribution defined by $G_b(1^\kappa, x, \pi, \cdot)$ for $b \in \{0,1\}$, $\mathcal{D}_0 \approx_{s,\varepsilon} \mathcal{D}_1$.

Fix $\lambda, n, \kappa, s, \varepsilon^{-1} \in \mathsf{Log}$, $x = (\mathsf{ch}, 1^p) \in \mathcal{U}$, and π be its witness. Assume that $\mathcal{K}^t(x\|w) \leq \ell(|x| + |\pi|)$, and let $\mathcal{D}_b$ be the distribution defined as above. Following the proof of Theorem 6.7, we define hybrids $\mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_2, \dots \mathcal{H}_{11}$, where $\mathcal{H}_0$ is identical to $\mathcal{D}_0$ and $\mathcal{H}_{11}$ is identical to $\mathcal{D}_1$. By a hybrid argument with [HJLM26, Lemma 4.15], it suffices to prove that for every $i \leq 10$, $\mathcal{H}_i \approx_{s,\varepsilon/11} \mathcal{H}_{i+1}$.

$\mathcal{H}_0 \approx \mathcal{H}_1$**:** This is proved using the security of commitment scheme, i.e., $\mathsf{ComHiding}[\lambda_0]$, with the help of the reduction lemma [HJLM26, Lemma 7.12]. Note that $\mathsf{ComHiding}[\lambda_0]$ consists of sentences $\phi_1, \phi_2, \dots, \phi_d, \dots$ corresponding to $\lambda^{-1}, \lambda^{-2}, \dots, \lambda^{-d}, \dots$ security for $\lambda > \lambda_0$. We choose ϕ_d for a sufficiently large d based on the size of adversary and advantage we need (which depends on $S(\cdot), \mathcal{E}(\cdot), N(\cdot)$) to prove $\mathcal{H}_0 \approx_{s,\varepsilon/11} \mathcal{H}_1$.

$\mathcal{H}_1 \approx \mathcal{H}_2 \approx \mathcal{H}_3$**:** For this step, we define additional hybrids $\mathcal{H}_{2,(i,q),0}, \dots, \mathcal{H}_{2,(i,q),5}$ that runs over all $i \in [n] \setminus \mathcal{I}$ and $q \in [Q]$, where the first hybrid is $\mathcal{H}_1$, and the last hybrid is $\mathcal{H}_3$.

Note that the number of hybrids is at most $100nQ \in \mathsf{Log}$. Therefore, we can apply the hybrid lemma [HJLM26, Lemma 4.15] and it suffices to prove that consecutive hybrids are $(s, \varepsilon/(100nQ))$-indistinguishable.

We order (i, q) in a canonical order.

$\mathcal{H}_{2,(i,q),0} \approx \mathcal{H}_{2,(i,q),1}$**:** This step uses $\mathsf{ComHiding}[\lambda_0]$ and the reduction lemma.

$\mathcal{H}_{2,(i,q),1} \approx \mathcal{H}_{2,(i,q),2}$**:** This step uses $\mathsf{WISecure}[\lambda_0]$ and the reduction lemma.

$\mathcal{H}_{2,(i,q),2} \approx \mathcal{H}_{2,(i,q),3}$**:** This step uses $\mathsf{PKESecure}[\lambda_0]$ and the reduction lemma.

$\mathcal{H}_{2,(i,q),3} \approx \mathcal{H}_{2,(i,q),4}$**:** This step uses $\mathsf{WISecure}[\lambda_0]$ and the reduction lemma.

$\mathcal{H}_{2,(i,q),4} \approx \mathcal{H}_{2,(i,q),5}$**:** This step uses $\mathsf{ComHiding}[\lambda_0]$ and the reduction lemma.

Note that $\mathcal{H}_{2,(i,q),5}$ is identical to the hybrid $\mathcal{H}_{2,(i',q'),0}$, where (i', q') is the next pair in the order. Similar to $\mathcal{H}_0 \approx \mathcal{H}_1$, for each of the hybrid, we choose a sentence in corresponding sets to make sure $(s, \varepsilon/(100nQ))$-indistinguishability. This is always possible as n, Q are fixed polynomial in λ that depend on $S(\cdot), \mathcal{E}(\cdot), N(\cdot)$.

$\mathcal{H}_3 \approx \mathcal{H}_4 \approx \mathcal{H}_5$**:** These hybrids use $\mathsf{ComHiding}[\lambda_0], \mathsf{WISecure}[\lambda_0]$ and the reduction lemma.

$\mathcal{H}_5 \approx \mathcal{H}_6$**:** This step uses $\mathsf{PolyEFIOSecure}[\lambda_0]$ and the reduction lemma. Note that we need to prove within $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$" that the machines $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$ are (ρ, τ)-EF-equivalence. As mentioned in Remark 16, the equivalence is indeed provable in PV_1.

$\mathcal{H}_6 \approx \cdots \approx \mathcal{H}_{10}$**:** These steps use $\mathsf{ComHiding}[\lambda_0], \mathsf{WISecure}[\lambda_0]$ and the reduction lemma.

$\mathcal{H}_{10} \approx \mathcal{H}_{11}$**:** This step uses $\mathsf{PolyEFIOSecure}[\lambda_0]$ and the reduction lemma. The formalization of EF-equivalence is similar to $\mathcal{H}_5 \approx \mathcal{H}_6$.

This completes the proof. □

By combining Lemma 10.21 and Lemma 10.18, we can show that for every polynomials $S, \mathcal{E}, N$ and $\gamma := 1/2$, $\mathcal{T}$ proves $(\mathcal{C}, \mathcal{U}, T, \gamma, t, \ell)$-$\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0'']$ for some function $\lambda_0'' : \mathbb{N} \to \mathbb{N}$. By Corollary 9.4, we further show that the game $\boldsymbol{G} = (\mathcal{C}, \mathcal{U}, T, \gamma, N)$ is a $(S(\lambda), 2/\mathcal{E}(\lambda))$ secure game for every polynomials $S, \mathcal{E}, N$, namely, the game is polynomially secure. Theorem 10.19 then follows from the correctness of the security game formalization (see Claim 10.20).

10.3.4 Rate-1 iO for Circuits

Now we prove the security of rate-1 iO for circuits constructed in Corollary 5.4 based on proof complexity generators against feasible cryptography.

Theorem 10.22. *Suppose that there exist the following cryptographic primitives:*

- *A polynomially secure PRGs.*
- *A subexponentially-secure (ρ, τ)-EF-iO for programs with split description (see Definition 3.18), where $\rho(\lambda) = \mathsf{poly}(\lambda)$ and $\tau(t, \lambda) = \mathsf{poly}(t, \lambda)$ are some fixed functions.*

Let $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$ be a function and $\delta \in (0, 1), \lambda_0' \in \mathbb{N}$ be constants such that $\mathsf{PRGSecure}[\lambda_0]$ and $\mathsf{ExpEFIOSecure}[\delta, \lambda_0]$ be the security axioms of the schemes above. Moreover, assume that there is a subexponentially-secure proof complexity generator against $\mathcal{T}$ provable soundness for

$$\mathcal{T} := \mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{PRGSecure}[\lambda_0] + \mathsf{ExpEFIOSecure}[\delta, \lambda_0']$$

with compression rate $\ell(n) = o(n)$.

Then there exists an indistinguishability obfuscation scheme $\mathcal{O}$ for circuits $C : \{0,1\}^n \to \{0,1\}$ where $|\mathcal{O}(1^\lambda, C, 1^n)| = |C|(1 + o(1)) + \mathsf{poly}(n, \lambda)$.

Construction. We closely follow the construction in Corollary 5.4; our goal is to prove the the game $\boldsymbol{G}_{q,\tau}$ is secure for sufficiently large $c \in \mathbb{N}$ using the assumptions of Theorem 10.22. Note the conclusion of Theorem 10.22 follows immediately from the security of $\boldsymbol{G}_{q,\tau}$ (see the proof of Theorem 5.1).

First, we construct a description-succinct iO from the (ρ, τ)-EF-iO for programs following Section 10.2.3. By Theorem 10.14, we further know that there exists a function $\lambda_0''(k) : \mathbb{N} \to \mathbb{N}$ such that $\mathcal{T} \vdash \mathsf{PolyIOSecure}[\lambda_0'']$. Then fix $t(n) = \mathsf{poly}(n)$ and $\ell(n) = o(n)$ be the time and compression rate of the subexponentially-secure PCG, and let κ, p, q, τ be the functions specified in Section 5.1, we define $\mathcal{O}(1^\lambda, M, 1^n)$ as the function $\mathcal{O}_{t,\ell,\kappa,p}$ in Figure 1.

Formalization of Security. Similar to EF-MIFE, we define the security of rate-1 iO for programs using indistinguishability-based security. Let $T(n) = 2^{n^\delta}$ for a sufficiently small constant δ. Following Section 5.1, we define $\mathcal{U} \in \mathsf{DTIME}[T]$ be the language that consists of $(1^m, M_0, M_1, \mathsf{pub})$ such that

- $m \in \mathbb{N}$, $M_0[\mathsf{pub}], M_1[\mathsf{pub}]$ are TMs with split description,
- $\frac{1}{2}q(m) \le |M_b[\mathsf{pub}]| \le q(m)$,
- $\mathsf{EQRT}(M_0[\mathsf{pub}], M_1[\mathsf{pub}]) = 1$ where the machines have explicit clocks given by τ,
- for all $x \in \{0,1\}^{\le m}$, $M_0(x) = M_1(x)$.

Here, $q(m) = cm^c$ for a sufficiently large $c \ge 1$ so that $\mathcal{U} \in \mathsf{DTIME}[T]$. For $b \in \{0,1\}$, we define $G_b(1^{\tau_G}, x, \pi, \mathsf{sd})$ be the algorithm that:

- Parse x as $(1^m, M_0, M_1, \mathsf{pub})$.
- Ignore τ_G and output $\mathcal{O}(1^\lambda, M_b, 1^m; \mathsf{sd})$.

Let $\mathcal{U}_{\mathsf{pad}} \in \mathsf{NP}$ be the padded version of $\mathcal{U}$. It then follows that the indistinguishability game corresponding to $(G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T)$ (see Definition 10.17) is exactly the game $\boldsymbol{G}_{q,\tau}$ in Corollary 5.4.

Formalization of Security Proof. We now show that the security of rate-1 iO (i.e. Corollary 5.4) can be formalized in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Formally:

Lemma 10.23. *Let $\mathcal{T}$ be the theory in Theorem 10.22, $t, \ell : \mathsf{Log} \to \mathsf{Log}$, $t(n) = \mathsf{poly}(n)$, $\ell(n) = o(n)$. Fix the construction and functions κ, p, q, τ in Corollary 5.4. Fix the constructions of PRGs and (ρ, τ)-EF-iO specified in Theorem 10.22. Then for all polynomials $N(\lambda), S(\lambda), \mathcal{E}(\lambda) : \mathsf{Log} \to \mathsf{Log}$, there exists a function $\lambda_0 : \mathbb{N} \to \mathbb{N}$ such that*

$$\mathcal{T} \vdash (G_0, G_1, \mathcal{U}, T, t, \ell)\text{-}\mathsf{IndSecure}[S, \mathcal{E}, N, \lambda_0].$$

Proof Sketch. We follow the proof of Corollary 5.4 (which uses Theorem 5.3), and formalize the argument in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Note that $\mathcal{H}_{1,0}$ and $\mathcal{H}_{1,1}$ in Theorem 5.3 are exactly the distributions sampled by G_0 and G_1. The proof of Theorem 5.3 uses a hybrid argument, which can be formalized using [HJLM26, Lemma 4.15].

Now we verify that the indistinguishability of consecutive hybrids can be proved within $\mathcal{T}$. Most steps are essentially similar to Lemma 10.21, while a subtle issue (corresponding to the formalization of description-succinct iO) is explained in detail (marked with red and bold notice).

$\mathcal{H}_{1,b} \approx \mathcal{H}_{2,b}$**:** These two distributions are isomorphic (i.e. identical up to an efficiently computable permutation of seeds).

$\mathcal{H}_{2,b} \approx \mathcal{H}_{3,b}$**:** This follows from the security axiom of PRGs (i.e. $\mathsf{PRGSecure}[\lambda_0]$) and the reduction lemma (see [HJLM26, Lemma 7.12]).

$\mathcal{H}_{3,b} \approx \mathcal{H}_{4,b}$**:** This step follows from $\mathsf{PolyIOSecure}[\lambda'']$, which is known to be provable from $\mathcal{T}$ via Theorem 10.14, and the reduction lemma (see [HJLM26, Lemma 7.12]).

Note that: PolyIOSecure requires that the input length m of machines $M_0[\mathsf{pub}], M_1[\mathsf{pub}]$ to satisfy $m \in \mathsf{LogLog}$. In this security game, we use the padded version $\mathcal{U}_{\mathsf{pad}} \in \mathsf{NP}$ of $\mathcal{U} \in \mathsf{DTIME}[T]$. Subsequently, for the instance $x = (1^m, M_0, M_1, \mathsf{pub})$, we have $T(|x|) = 2^{q(m)^\delta}$ and $T(|x|) \in \mathsf{Log}$.[31] Moreover, we have set $q(m) = cm^c$ such that $2^m \leq T(|x|)$, which implies that $m \leq |T(|x|)|$ and thus $m \in \mathsf{LogLog}$.

$\mathcal{H}_{4,b} \approx \mathcal{H}_{5,b}$**:** This follows from $\mathsf{PRGSecure}[\lambda_0]$) and the reduction lemma.

$\mathcal{H}_{5,0} \approx \mathcal{H}_{5,1}$**:** These two distributions are isomorphic.

This completes the proof. □

By combining Lemma 10.23, Lemma 10.18, and the master theorem with subexponential security Corollary 9.6, we know that under the assumptions of Theorem 10.22, the game $\boldsymbol{G}_{q,\tau}$ in Corollary 5.4 is secure. Theorem 10.22 then follows by the proof of Theorem 5.1; note that this step does not involve the application of the master theorem, and thus there is no need to formalize it in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

10.3.5 Rate-1 iO for EF Equivalent Programs

Theorem 10.24. *Suppose that there exist the following cryptographic primitives:*

- *A superpolynomially secure PRGs.*

[31] To see that $T(|x|) \in \mathsf{Log}$, recall that in Definition 10.16, the variable $\tau \in \mathsf{Log}$ is given and we assume $\tau \geq T(|x|)$ (note that this variable τ is not the same as the function τ in this particular instantiation).

- *A superpolynomially secure (ρ,τ)-EF-iO for programs with split description (see Definition 3.18), where $\rho(\lambda)=\mathsf{poly}(\lambda)$ and $\tau(t,\lambda)=\mathsf{poly}(t,\lambda)$ are some fixed functions.*

Let $n_0,\lambda_0\in\mathbb{N}$ be constants such that $\mathsf{PRGSecure}[n^{\omega(1)},n_0]$ and $\mathsf{EFIOSecure}[n^{\omega(1)},n_0]$ are security axioms of the schemes above. Moreover, assume that there is a $n^{\omega(1)}$-secure proof complexity generator against $\mathcal{T}$ provable soundness for

$$\mathcal{T}:=\mathsf{APC}_1+\text{“}\mathsf{prBPP}=\mathsf{prP}\text{”}+\mathsf{PRGSecure}[n^{\omega(1)},n_0]+\mathsf{EFIOSecure}[\lambda^{\omega(1)},\lambda_0]$$

with compression rate $\ell(n)=n^{o(1)}$.

Then there exists an superpolynomially secure (ρ,τ)-EF-iO for programs with split description where $|\mathcal{O}(1^\lambda,M[\mathsf{pub}],1^n)|=|M[\mathsf{pub}]|+\mathsf{poly}(\lambda)$.

Formalization of Security. We define the security of (ρ,τ)-EF-iO for programs using indistinguishability-based security. Following Section 5.2, we define the language $\mathcal{U}_{\rho,\tau}$ of tuples $(1^m,M_0,M_1,\mathsf{pub})$ where the following hold:

- $m\in\mathbb{N}$, $M_0[\mathsf{pub}],M_1[\mathsf{pub}]$ are TMs with split description,
- for all $x\in\{0,1\}^{\le m}$, $\mathsf{EQRT}(M_0[\mathsf{pub}])=\mathsf{EQRT}(M_1[\mathsf{pub}])$ where the clock is given by τ,
- there exists a machine $\mathcal{M}_\Pi$ which on input $(1^k,M_0,M_1,\mathsf{pub})$, in $\tau(k)$ time outputs $[\![M_0[\mathsf{pub}](x)=M_1[\mathsf{pub}](x)]\!]_k$ for all $k\le m$.

Formalization of Security Proof. We now show that the security of (ρ,τ)-EF-secure rate-1 iO (i.e. Theorem 5.2) can be formalized in $\mathsf{APC}_1+\text{“}\mathsf{prBPP}=\mathsf{prP}\text{”}$. Formally:

Lemma 10.25. *Let $\mathcal{T}$ be the theory in Theorem 10.24, $t,\ell:\mathsf{Log}\to\mathsf{Log}$, $t(n)=\mathsf{poly}(n)$, $\ell(n)=n^{o(1)}$. Fix the construction and functions κ,p,q,τ in Theorem 5.2. Fix the constructions of PRGs and (ρ,τ)-EF-iO specified in Theorem 10.24. Then for every polynomial $N(\lambda)$, there are slightly superpolynomial functions $S(\lambda),\mathcal{E}(\lambda):\mathsf{Log}\to\mathsf{Log}$ and a function $\lambda_0:\mathbb{N}\to\mathbb{N}$ such that*

$$\mathcal{T}\vdash(G_0,G_1,\mathcal{U},T,t,\ell)\text{-}\mathsf{IndSecure}[S,\mathcal{E},N,\lambda_0].$$

Proof Sketch. Note that the proof complexity generator is slightly superpolynomially secure against $\mathsf{APC}_1+\text{“}\mathsf{prBPP}=\mathsf{prP}\text{”}$ extended by axioms for slightly superpolynomially secure PRG and succinct EF-iO. We choose $S(\lambda),\mathcal{E}(\lambda)$ to be superpolynomial functions that are bounded by the superpolynomial functions in $\mathcal{T}$.

To prove the IndSecure sentences from $\mathcal{T}$, we follow the proof in Theorem 5.3, which uses a hybrid argument that can be formalized with [HJLM26, Lemma 4.15]. We then prove the all consecutive hybrids can be proved to be indistinguishable in the theory $\mathcal{T}$:

$\mathcal{H}_{1,b}\approx\mathcal{H}_{2,b}$: These two distributions are isomorphic (i.e. identical up to an efficiently computable permutation of seeds).

$\mathcal{H}_{2,b}\approx\mathcal{H}_{3,b}$: This follows from the security axiom of PRGs (i.e. $\mathsf{PRGSecure}[n^{\omega(1)},n_0]$) and the reduction lemma (see [HJLM26, Lemma 7.12]), provided that the functions $S,\mathcal{E}$ are much smaller than functions hidden in $n^{\omega(1)}$.

$\mathcal{H}_{3,b} \approx \mathcal{H}_{4,b}$: This step follows from $\mathsf{EFIOSecure}[\lambda^{\omega(1)}, \lambda_0]$ and the reduction lemma, provided that the functions $S, \mathcal{E}$ are much smaller than functions hidden in $n^{\omega(1)}$. Note that the $(O(1), \mathsf{poly}(\cdot))$-equivalence of the machines is indeed derived from a PV proof of equivalence of these two machines. Subsequently, by Theorem 3.1, it can be proved in PV_1 (and also $\mathcal{T}$) that these two machines are $(O(1), \mathsf{poly}(\cdot))$-equivalence in any straightforward formalization.

$\mathcal{H}_{4,b} \approx \mathcal{H}_{5,b}$: This follows from $\mathsf{PRGSecure}[n^{\omega(1)}, n_0]$ and the reduction lemma.

$\mathcal{H}_{5,0} \approx \mathcal{H}_{5,1}$: These two distributions are isomorphic.

We note that although the argument involves superpolynomial functions $\lambda^{\omega(1)}$ and $n^{\omega(1)}$, our formalization using PV-formalizable size bounds ensures that it can be formalized within $\mathsf{APC}_1 +$ "$\mathsf{prBPP} = \mathsf{prP}$". Specifically, $\mathsf{IndSecure}$, $\mathsf{PRGSecure}$, $\mathsf{EFIOSecure}$ will use the superpolynomial functions such as $S(\lambda), \mathcal{E}(\lambda)$ only in the following restricted way:

- $s \leq S(\lambda)$ for $s, \lambda \in \mathsf{Log}$, or
- $\varepsilon^{-1} \leq \mathcal{E}(\lambda)$ for $\varepsilon^{-1}, \lambda \in \mathsf{Log}$.

It is then formalized using the corresponding PV functions that define these size bounds (see Definition 3.2). This completes the proof. □

The proof of Theorem 10.24 then follows from the same template as in Section 10.3.3 and 10.3.4. We formalize the security using indistinguishability-based definition, prove the $\mathsf{IndSecure}$ sentences from $\mathcal{T}$ (see Lemma 10.25), and apply the master theorem via Section 10.3.2. We will apply Corollary 9.5 that utilizes PCG with slightly superpolynomial security.

10.3.6 Non-Adaptive SNARG with Short CRS

Theorem 10.26. *Let L be a polynomial. Suppose that there exist the following cryptographic primitives:*

- *a polynomially secure PRG.*
- *a polynomially secure unleveled fully homomorphic encryption scheme,*
- *a polynomially secure batch argument scheme for* NP*,*

Let $\lambda_0(k) : \mathbb{N} \to \mathbb{N}$ be a function and $\lambda_0' \in \mathbb{N}$ be a constant such that $\mathsf{PRGSecure}[\lambda_0']$, $\mathsf{FHESecure}[\lambda_0]$, and $\mathsf{BARGSecure}[\lambda_0]$ be the security axioms of the schemes above. Moreover, assume that there is a polynomially secure proof complexity generator against $\mathcal{T}$ provable soundness for

$$\mathcal{T} := \mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{PRGSecure}[\lambda_0'] + \mathsf{FHESecure}[\lambda_0] + \mathsf{BARGSecure}[\lambda_0]$$

with compression rate $\ell(n) = n^{o(1)}$.

Then there exists an SNARG scheme satisfying L-EF-soundness (Definition 7.1) where the CRS size is independent of L.

As usual, we will prove the security under compression of our SNARG construction in $\mathcal{T}$, and apply the master theorem using polynomially secure PCG (see Corollary 9.4).

The proof of security under compression is different from previous examples, as the security of SNARG is not defined using indistinguishability but with search security games. Fortunately, the formalization of SNARG security game as well as the security proof of the JKLV construction [JKLV24] was thoroughly studied in [HJLM26]. As our construction makes only minor changes to the JKLV construction, it suffices

to pinpoint how our changes will affect the security proof, and verify that it remains formalizable in APC_1. First, recall that by Theorem 10.13, the security axioms of the underlying SEH scheme is provable from $\mathcal{T}$.

Moreover, [HJLM26] also showed how to formalize Theorem 7.6 in $\mathcal{T}$. Finally, after application of FHE security, the construction of the local assignment generator from a cheating prover follows an identical argument as was done in [HJLM26]. Therefore the soundness proof assuming compression can therefore be formalized in $\mathcal{T}$. This allows us to use our master theorem.

Note that once we prove soundness, one can invoke Remark 7 to improve the verifier efficiency. Since this step does not require the master theorem, there is no need to formalize this in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$.

Acknowledgements

A. Jain was supported in part by NSF CAREER 1942789, Johns Hopkins University Catalyst award, JP Morgan Faculty Award and research gifts from Ethereum Foundation, Stellar Development Foundation, and Cisco. J. Li received support from National Science Foundation under Grant No. CCF-2420092. S. Mathialagan was supported in part by Jane Street and the Simons Institute for the Theory of Computing.

Part of this work was done while the authors were visiting the Simons Institute for the Theory of Computing, UC Berkeley, and while Yao-Ching Hsieh and Jiatu Li were interns at NTT Research, Sunnyvale.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. 114, 115

[ABG+13] Prabhanjan Ananth, Dan Boneh, Sanjam Garg, Amit Sahai, and Mark Zhandry. Differing-inputs obfuscation and applications. Cryptology ePrint Archive, Report 2013/689, 2013. 6, 15

[ABSRW04] Michael Alekhnovich, Eli Ben-Sasson, Alexander A Razborov, and Avi Wigderson. Pseudorandom generators in propositional proof complexity. *SIAM Journal on Computing*, 34(1):67–88, 2004. 3

[AJS17] Prabhanjan Ananth, Abhishek Jain, and Amit Sahai. Indistinguishability obfuscation for Turing machines: Constant overhead and amortization. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 252–279. Springer, Cham, August 2017. 2, 6, 15

[BCCT12] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In Shafi Goldwasser, editor, *ITCS 2012*, pages 326–349. ACM, January 2012. 16

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 111–120. ACM Press, June 2013. 16

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 52–73. Springer, Berlin, Heidelberg, February 2014. 8, 21

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Berlin, Heidelberg, August 2001. 1

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, Berlin, Heidelberg, March 2014. 26

[BGJS15] Saikrishna Badrinarayanan, Divya Gupta, Abhishek Jain, and Amit Sahai. Multi-input functional encryption for unbounded arity functions. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 27–51. Springer, Berlin, Heidelberg, November / December 2015. 9, 15

[BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13(4):850–864, 1984. 4

[BM15] Christina Brzuska and Arno Mittelbach. Universal computational extractors and the superfluous padding assumption for indistinguishability obfuscation. Cryptology ePrint Archive, Report 2015/581, 2015. 1

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Berlin, Heidelberg, March 2011. 9

[BSW16] Mihir Bellare, Igors Stepanovs, and Brent Waters. New negative results on differing-inputs obfuscation. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 792–821. Springer, Berlin, Heidelberg, May 2016. 15

[Bus85] Samuel R Buss. *Bounded arithmetic*. Princeton University, 1985. 28

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In Venkatesan Guruswami, editor, *56th FOCS*, pages 171–190. IEEE Computer Society Press, October 2015. 2, 6, 15

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, Berlin, Heidelberg, December 2013. 26

[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. 62

[CJ25] Liyan Chen and Zhengzhong Jin. On the impossibility of SNARGs with short CRS : (or: Revisiting gentry-wichs barrier in the non-adaptive setting). In *66th FOCS*, pages 1741–1760. IEEE Computer Society Press, December 2025. 2, 8

[CJJ21] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Non-interactive batch arguments for NP from standard assumptions. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 394–423, Virtual Event, August 2021. Springer, Cham. 4

[CL24] Yilei Chen and Jiatu Li. Hardness of range avoidance and remote point for restricted circuits via cryptography. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *56th ACM STOC*, pages 620–629. ACM Press, June 2024. 10

[CLOW26] Lijie Chen, Jiatu Li, Igor C. Oliveira, and Ryan Williams. A theory for probabilistic polynomial-time reasoning. *Electron. Colloquium Comput. Complex.*, TR26, 2026. 13, 117

[CN10] Stephen Cook and Phuong Nguyen. *Logical foundations of proof complexity*, volume 31. Cambridge University Press Cambridge, 2010. 13, 28

[Coo75] Stephen A. Cook. Feasibly constructive proofs and the propositional calculus (preliminary version). In William C. Rounds, Nancy Martin, Jack W. Carlyle, and Michael A. Harrison, editors, *Proceedings of the 7th Annual ACM Symposium on Theory of Computing, May 5-7, 1975, Albuquerque, New Mexico, USA*, pages 83–97. ACM, 1975. 7, 12, 13, 16, 28, 29

[CR79] Stephen A Cook and Robert A Reckhow. The relative efficiency of propositional proof systems. *The journal of symbolic logic*, 44(1):36–50, 1979. 2, 3, 10, 28, 42

[DHJQ26] Jesko Dujmovic, Yao-Ching Hsieh, Abhishek Jain, and Willy Quach. How to use polynomially-hard io: Turing machine obfuscation and more. In *Advances in Cryptology – CRYPTO 2026*, pages 317–349, 2026. 62

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009. 4

[GGG+14] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 578–602. Springer, Berlin, Heidelberg, May 2014. 9, 18, 24, 25, 39

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013. 18

[GGHW14] Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 518–535. Springer, Berlin, Heidelberg, August 2014. 7, 15

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984. 26

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476. ACM Press, June 2013. 7, 8, 34

[GJO16] Vipul Goyal, Aayush Jain, and Adam O'Neill. Multi-input functional encryption with unbounded-message security. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 531–556. Springer, Berlin, Heidelberg, December 2016. 9

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Berlin, Heidelberg, August 2013. 4, 30, 31

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 99–108. ACM Press, June 2011. 2, 8

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. 90

[HJLM26] Yao Ching Hsieh, Abhishek Jain, Jiatu Li, and Surya Mathialagan. SNARGs for NP from unprovability of mathematical theorems (or: How to use the simplicity of cryptographic reasoning). *STOC 2026 (to appear).*, 2026. The full version is available at this Google Drive link. 4, 5, 13, 14, 15, 16, 23, 26, 28, 38, 74, 77, 79, 80, 87, 88, 93, 95, 96, 97, 98, 101, 103, 104, 105, 106, 116, 117, 118

[Hol15] Justin Holmgren. On necessary padding with IO. Cryptology ePrint Archive, Report 2015/627, 2015. 1, 15, 27

[Ila25a] Rahul Ilango. Gödel in cryptography: Effectively zero-knowledge proofs for NP with no interaction, no setup, and perfect soundness. In *FOCS*, 2025. 16

[Ila25b] Rahul Ilango. The oracle derandomization hypothesis is false (and more) assuming no natural proofs. *Electron. Colloquium Comput. Complex.*, TR25-190, 2025. 10, 11, 14, 42

[ILW23] Rahul Ilango, Jiatu Li, and R. Ryan Williams. Indistinguishability obfuscation, range avoidance, and bounded arithmetic. In Barna Saha and Rocco A. Servedio, editors, *55th ACM STOC*, pages 1076–1089. ACM Press, June 2023. 10, 11

[IPS15] Yuval Ishai, Omkant Pandey, and Amit Sahai. Public-coin differing-inputs obfuscation and its applications. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 668–697. Springer, Berlin, Heidelberg, March 2015. 6, 15

[IW97] Russell Impagliazzo and Avi Wigderson. *P = BPP* if *E* requires exponential circuits: Derandomizing the XOR lemma. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 220–229. ACM, 1997. 120, 121

[Jeř05] Emil Jeřábek. *Weak Pigeonhole Principle, and Randomized Computation.* PhD thesis, Faculty of Mathematics and Physics, Charles University, Prague, 2005. 12

[Jeř07a] Emil Jeřábek. Approximate counting in bounded arithmetic. *Journal of Symbolic Logic*, 72(3):959–993, 2007. 4, 5, 13, 28, 79, 87

[Jer07b] Emil Jerábek. On independence of variants of the weak pigeonhole principle. *J. Log. Comput.*, 17(3):587–604, 2007. 13

[JJ22] Abhishek Jain and Zhengzhong Jin. Indistinguishability obfuscation via mathematical proofs of equivalence. In *63rd FOCS*, pages 1023–1034. IEEE Computer Society Press, October / November 2022. 2, 6, 7, 13, 16, 44

[JJ25] Abhishek Jain and Zhengzhong Jin. Sometimes-decryptable homomorphic encryption from sub-exponential DDH. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part III*, volume 16002 of *LNCS*, pages 406–439. Springer, Cham, August 2025. 1

[JJMP25] Abhishek Jain, Zhengzhong Jin, Surya Mathialagan, and Omer Paneth. On succinct obfuscation via propositional proofs. In *66th FOCS*, pages 1703–1740. IEEE Computer Society Press, December 2025. The full version is available at this Google Drive link. 2, 4, 6, 7, 13, 14, 15, 16, 17, 20, 34, 35, 37, 38, 88, 91, 92, 96

[JKLM25] Zhengzhong Jin, Yael Tauman Kalai, Alex Lombardi, and Surya Mathialagan. Universal SNARGs for NP from proofs of correctness. In Michal Koucký and Nikhil Bansal, editors, *57th ACM STOC*, pages 933–943. ACM Press, June 2025. 1, 13, 16, 88

[JKLV24] Zhengzhong Jin, Yael Kalai, Alex Lombardi, and Vinod Vaikuntanathan. SNARGs under LWE via propositional proofs. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *56th ACM STOC*, pages 1750–1757. ACM Press, June 2024. 1, 2, 7, 8, 9, 14, 16, 22, 23, 26, 72, 73, 74, 77, 93, 105

[JLL23] Aayush Jain, Huijia Lin, and Ji Luo. On the optimal succinctness and efficiency of functional encryption and attribute-based encryption. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 479–510. Springer, Cham, April 2023. 2, 6, 15

[JLP26] Zhengzhong Jin, Mingqi Lu, and Bo Peng. Snarks from lwe via non-black-box reductions. *To appear at STOC 2026.*, 2026. 8, 15, 16

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 60–73. ACM Press, June 2021. 1

[JLS22] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 670–699. Springer, Cham, May / June 2022. 1

[Kha22] Erfan Khaniki. Nisan-wigderson generators in proof complexity: New lower bounds. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, Philadelphia, PA, USA, July 20-23, 2022*, LIPIcs, pages 17:1–17:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. 12

[Kor22] Oliver Korten. The hardest explicit construction. In *62nd FOCS*, pages 433–444. IEEE Computer Society Press, February 2022. 11

[Kor25] Oliver Korten. Range avoidance and the complexity of explicit constructions. *Bull. EATCS*, 145, 2025. 10, 11

[KP89] Jan Krajícek and Pavel Pudlák. Propositional proof systems, the consistency of first order theories and the complexity of computations. *J. Symb. Log.*, 54(3):1063–1079, 1989. 16

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 669–684. ACM Press, November 2013. 26

[Kra95] Jan Krajíček. *Bounded arithmetic, propositional logic and complexity theory*, volume 60. Cambridge University Press, 1995. 12, 13, 28

[Kra01] Jan Krajíček. On the weak pigeonhole principle. *Fundamenta Mathematicae*, 170:123–140, 2001. 3

[Kra19] Jan Krajíček. *Proof complexity*, volume 170. Cambridge University Press, 2019. 13, 16, 28

[Kra24] Jan Krajíček. On the existence of strong proof complexity generators. *Bulletin of Symbolic Logic*, 30(1):20–40, 2024. 10, 42, 120

[KS25] Oliver Korten and Rahul Santhanam. How to construct random strings. In *CCC*, volume 339 of *LIPIcs*, pages 35:1–35:32. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. 11, 113

[KvM02] Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002. 121, 122

[Li25] Jiatu Li. An introduction to feasible mathematics and bounded arithmetic for computer scientists. *Electron. Colloquium Comput. Complex.*, TR25-086, 2025. 13, 28, 29, 36

[LMOP26] Zhenjian Lu, Noam Mazor, Igor C. Oliveira, and Rafael Pass. Lower bounds on the overhead of indistinguishability obfuscation. In Joan Daemen and Emmanuel Thomé, editors, *EUROCRYPT 2026, Part V*, volume 16545 of *LNCS*, pages 154–183. Springer, Cham, May 2026. 2, 6, 15

[LO23] Jiatu Li and Igor C. Oliveira. Unprovability of strong complexity lower bounds in bounded arithmetic. In Barna Saha and Rocco A. Servedio, editors, *55th ACM STOC*, pages 1051–1057. ACM Press, June 2023. 13

[Mat25] Surya Mathialagan. *Succinct Cryptography via Propositional Proofs*. PhD thesis, MIT, 2025. 13, 28, 30, 31, 88, 92

[MDS25] Yaohua Ma, Chenxin Dai, and Elaine Shi. Quasi-linear indistinguishability obfuscation via mathematical proofs of equivalence and applications. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part III*, volume 15603 of *LNCS*, pages 157–186. Springer, Cham, May 2025. 6, 9, 15, 16, 39

[Mic94] Silvio Micali. CS proofs (extended abstracts). In *35th FOCS*, pages 436–453. IEEE Computer Society Press, November 1994. 8

[Nao03] Moni Naor. On cryptographic assumptions and challenges (invited talk). In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 96–109. Springer, Berlin, Heidelberg, August 2003. 1

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. 120, 121

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pages 427–437. ACM Press, May 1990. 18

[Oli25] Igor C. Oliveira. SIGACT news complexity theory column 124 meta-mathematics of computational complexity theory. *SIGACT News*, 56(1):41–68, 2025. 13, 28, 115

[Par71] Rohit Parikh. Existence and feasibility in arithmetic. *J. Symb. Log.*, 36(3):494–508, 1971. 13, 80, 98

[Pic15] Ján Pich. Logical strength of complexity theory and a formalization of the PCP theorem in bounded arithmetic. *Log. Methods Comput. Sci.*, 11(2), 2015. 13

[PR17] Omer Paneth and Guy N. Rothblum. On zero-testable homomorphic encryption and publicly verifiable non-interactive arguments. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part II*, volume 10678 of *LNCS*, pages 283–315. Springer, Cham, November 2017. 23, 73

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 89–114. Springer, Cham, August 2019. 62

[PS21] Ján Pich and Rahul Santhanam. Strong co-nondeterministic lower bounds for NP cannot be proved feasibly. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 223–233. ACM Press, June 2021. 13

[Raz15] Alexander A Razborov. Pseudorandom generators hard for k-dnf resolution and polynomial calculus resolution. *Annals of Mathematics*, pages 415–472, 2015. 12

[RSW22] Hanlin Ren, Rahul Santhanam, and Zhikun Wang. On the range avoidance problem for circuits. In *63rd FOCS*, pages 640–650. IEEE Computer Society Press, October / November 2022. 3, 10, 11, 43

[Rud97] Steven Rudich. Super-bits, demi-bits, and np/qpoly-natural proofs. In José D. P. Rolim, editor, *Randomization and Approximation Techniques in Computer Science, International Workshop, RANDOM'97, Bolognna, Italy, July 11-12. 1997, Proceedings*, volume 1269 of *Lecture Notes in Computer Science*, pages 85–93. Springer, 1997. 10, 14

[RVV24] Seyoon Ragavan, Neekon Vafa, and Vinod Vaikuntanathan. Indistinguishability obfuscation from bilinear maps and LPN variants. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part IV*, volume 15367 of *LNCS*, pages 3–36. Springer, Cham, December 2024. 1

[RWZ26] Hanlin Ren, Yichuan Wang, and Yan Zhong. Hardness of range avoidance and proof complexity generators from demi-bits. In *ITCS*, volume 362 of *LIPIcs*, pages 111:1–111:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2026. 10, 11

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Berlin, Heidelberg, May 2005. 9

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014. 26

[WW24] Brent Waters and David J. Wu. Adaptively-sound succinct arguments for NP from indistinguishability obfuscation. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *56th ACM STOC*, pages 387–398. ACM Press, June 2024. 38, 39

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd FOCS*, pages 80–91. IEEE Computer Society Press, November 1982. 4

A The Hardness Assumption of Korten and Santhanam

In a recent work, Korten and Santhanam [KS25] construct, under a natural complexity-theoretic assumption, an algorithm that generates string with high $\boldsymbol{K}^t$-complexity. Subsequently, it breaks any proof complexity generator. One may wonder whether this result should be considered as an attack of the proof complexity generator assumption.

We show that the *assumption* of Korten and Santhanam [KS25] almost directly breaks our assumption on PCGs. Therefore, their results should be interpreted as a reduction when there is no PCG, rather than a candidate attack to PCGs.

Assumption 7 (Korten-Santhanam Hardness Assumption)**.** There is an absolute constant $\varepsilon > 0$ such that the following holds. For any time-constructible $T(n), m(n)$, with $T(n) \geq n$ having at most d-th order exponential growth rate and $n \leq m(n) \leq \mathsf{poly}(n)$, there is a function $f : \mathbb{N} \to \{0,1\}^*$, $|f(n)| = m(n)$ computable in $\mathsf{poly}(T(n))$ time, which cannot be computed in time $T(n)^\varepsilon$ with $m(n)^\varepsilon$ bits of advice by any NP-oracle machine for more than finitely many n.

Proposition A.1. *Under Assumption 7, Assumption 2 (with polynomial security and stretch $\ell(m) = m^{o(1)}$) does not hold.*

Proof. Towards a contradiction, assume that both Assumption 7 and Assumption 2 hold. Fix any proof complexity generator $C_m : \{0,1\}^\ell \to \{0,1\}^m$ for $\ell(m) = m^{o(1)}$.

Let $c > 1$ be a constant to be determined later. We fix $d = 0$, $T(n) = 2^{cn}$, $m(n) = 2^n$ in Assumption 7 and let $f : \mathbb{N} \to \{0,1\}^*$ be computable in $2^{O(n)}$-time, $|f(n)| = m(n)$, that cannot be computed in time $2^{\varepsilon cn}$-time with $2^{\varepsilon n}$ bits of advice by NP-oracle machines. Consider the following NP verifier $V(y, \pi)$: Given $y \in \{0,1\}^m$, it ignores π and accepts if and only if $m = 2^n$ is a power of two, $n > c$, and $y = f(n)$.

By choosing $c > 1$ to be sufficiently large, the verifier is sound and complete, and thus breaks C_m as a PCG. Note that the completeness is trivial, as the verifier accepts infinitely many strings unconditionally. The soundness of V follows from the hardness of the function f. To see this, consider the polynomial-time algorithm A that given $n \in \mathbb{N}$ and an advice x_n of length $\ell(2^n)$, outputs $y := C_m(x)$. The algorithm takes an advice of length $\ell(2^n) < 2^{\varepsilon n}$ and runs in time $\mathsf{poly}(2^n)$; by taking $c > 1$ to be sufficiently large, its running time is at most $2^{\varepsilon cn}$. It then follows that $A_{x_n}(n) = f(n)$ for only finitely many $n \in \mathbb{N}$, and subsequently when c is sufficiently large, V will not accept $A_{x_n}(n)$. By the definition of A, we can see that V will not accept any string in the range of C_m. □

B Missing Formalizations

B.1 Circuit SAT Reduces to Hamiltonian Cycle

In this section, we verify that the Karp reduction from Circuit SAT to Hamiltonian Cycle can be formalized in PV_1. This is straightforward and folklore; we include it for completeness.

Lemma B.1. *There are* PV *functions $f, \mathcal{R}_s, \mathcal{R}_c$ such that the following sentences are provable in* PV_1*:*

- *(Syntax). For every circuit $C : \{0,1\}^n \to \{0,1\}$, $f(C)$ outputs a directed graph $G = (V, E)$.*
- *(Completeness). For every circuit $C : \{0,1\}^n \to \{0,1\}$ and input $x \in \{0,1\}^n$, if $C(x) = 1$, then $\mathcal{R}_c(C, x)$ outputs a Hamiltonian cycle in the graph $f(C)$.*
- *(Soundness). For every circuit $C : \{0,1\}^n \to \{0,1\}$ and any Hamiltonian cycle P in the graph $f(C)$, $\mathcal{R}_s(P) \in \{0,1\}^n$ and $C(\mathcal{R}_s(C, P)) = 1$.*

Proof Sketch. We formalize the proof that Circuit SAT is in NP (see, e.g., [AB09, Lemma 6.11]) and the NP-hardness of Hamiltonian Cycle (see, e.g., [AB09, Theorem 2.17]).

Reducing Circuit SAT to 3SAT. We first prove that the following sentences are true for some PV_1 function symbol $f_1(\cdot)$:

- For every circuit C, $f_1(C)$ outputs a formula in 3-CNF.
- For every circuit $C : \{0,1\}^n \to \{0,1\}$, $\exists x \in \{0,1\}^n\ C(x) = 1 \iff f_1(C)$ is satisfiable.

The construction follows from [AB09, Lemma 6.11], where we introduce auxiliary variables for wires and clauses for simulation of gates. To prove the second bullet above, we need to show that:

- (*Completeness*): Given a circuit C, an input x such that $C(x) = 1$, there is a satisfying assignment of $f_1(C)$. The satisfying assignment can be obtained by evaluating the circuit and fixing the auxiliary variables based on values on wires. The correctness of the satisfying assignment can be formalized by an induction on the execution of the algorithm.
- (*Soundness*): Given a circuit C and a satisfying assignment ρ of $f_1(C)$, there is an input x such that $C(x) = 1$. This is obtained by ignoring the assignment for auxiliary variables. To prove that $C(x) = 1$, we perform induction on the topological order of the circuit that for each gate G, the evaluation of G is identical to the assignment of the auxiliary variable corresponding to its output wire in ρ.

Reducing 3SAT to Hamiltonian Cycle. Next, we prove the following sentences are true for some PV_1 function symbol $f_2(\cdot)$:

- For every 3-CNF formula ϕ, $f_2(\phi)$ outputs a directed graph.
- For every 3-CNF formula $C : \{0,1\}^n \to \{0,1\}$, ϕ is satisfiable $\iff$ there is a Hamiltonian cycle in $f_2(\phi)$.

The construction follows from [AB09, Theorem 2.17]. At a high level, for a 3-CNF formula ϕ with n variables and m clauses, we introduce n bi-directional chains of length m corresponding to n variables, with the intended meaning that:

- if x_i is assigned to be 0, the Hamiltonian cycle visits the i-th chain from left to right;

- if x_i is assigned to 1, the Hamiltonian cycle visits the i-th chain from right to left.

Moreover, we introduce a start node, an end node, and m additional nodes corresponding to clauses (see [AB09, Figure 2.6]). (Note that [AB09, Theorem 2.17] reduces 3SAT to Hamiltonian Path Problem, while by adding an edge from the end node to the start node, we can easily obtain a reduction to Hamiltonian Cycle Problem.

It is straightforward to prove that if ϕ is satisfiable and x is a satisfying assignment, there is a polynomial time algorithm (with PV_1 correctness proof) that finds a Hamiltonian cycle, based on the correspondence above. Now we prove the converse direction: Suppose that ϕ is a formula and P is a Hamiltonian cycle of $f_2(\phi)$, then ϕ is satisfiable. The proof works as follows:

- We prove that any Hamiltonian cycle must include the edge from the end node to the start node. We remove that edge and make it a Hamiltonian path starting from the start node.
- We prove that following P, the first visited node in the i-th chain is either the leftmost or the rightmost node. Moreover, if the first visited node is the rightmost (resp. leftmost) one, nodes are visited from right to left (resp. from left to right) - denote this property by $(\diamond)$. This can be proved by an induction on i, which is an induction on polynomial time predicate and thus can be formalized in PV_1.
- For each $i \in [n]$, the assignment for the i-th variable is 1 if and only if following the path P, the first visited node in the i-th chain is the rightmost node.
- Assume, towards a contradiction, that this is not a satisfying assignment, then there is a clause that is not satisfied. Let v be the node corresponding to the clause and find its occurrence in P. By its local topology and the fact that it is not satisfied, it will necessarily violate $(\diamond)$.

Wrapping Up. Combining these two mappings, we can obtain a function f such that:

- For every circuit C, $f(C)$ outputs a directed graph.
- For every $C : \{0,1\}^n \to \{0,1\}$, $\exists x \in \{0,1\}^n\ C(x) = 1 \iff f(C)$ has a Hamiltonian cycle. In other words, we have
  - (*Completeness*). For every circuit $C : \{0,1\}^n \to \{0,1\}$ and input x such that $C(x) = 1$, there is a Hamiltonian cycle in $f(C)$.
  - (*Soundness*). For every $C : \{0,1\}^n \to \{0,1\}$ and any Hamiltonian cycle P in $f(C)$, there is an input x such that $C(x) = 1$.

Note that both completeness and soundness properties above are $\forall\Sigma_2^b$-sentences in the language of PV_1. The theorem follows by the witnessing theorem of PV_1, see, e.g., [Oli25, Theorem 3.1]. □

B.2 Description Succinct Obfuscation

Theorem 10.14 (Theorem 3.20, in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$). *The following is true for some* PV *functions* ρ, τ *that are polynomials. If* $\mathsf{iO}_{\rho,\tau}(1^\lambda, M[\mathsf{pub}], 1^n)$ *is a fully-succinct subexponentially secure* EF*-iO, then for a constant* $d \in \mathbb{N}$*,*

$$\mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n) := \mathsf{iO}_{\rho,\tau}(1^{\lambda+n^d}, M[\mathsf{pub}], 1^n)$$

is a description-succinct polynomially secure iO. Moreover, let $\mathsf{ExpEFIOSecure}[\delta, \lambda_0]$ *be the security axiom of* $\mathsf{iO}_{\rho,\tau}$ *as an* EF*-iO scheme, there exists a function* $\lambda_0'(k) : \mathbb{N} \to \mathbb{N}$ *such that*

$$\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + \mathsf{ExpEFIOSecure}[\delta, \lambda_0] \vdash \mathsf{PolyIOSecure}[\lambda_0'],$$

where $\mathsf{PolyIOSecure}[\lambda_0']$ *is the security axioms of* $\mathsf{iO}(1^\lambda, M[\mathsf{pub}], 1^n)$.

Proof. We work in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Fix $\delta \in (0,1)$ and $\lambda_0 \in \mathbb{N}$, and assume that the security axiom $\mathsf{ExpEFIOSecure}[\delta, \lambda_0]$ holds. Fix any constant $k \in \mathbb{N}$, and let $\lambda_0' \in \mathbb{N}$ be a constant to be determined later. Our goal is to prove that:

$$\forall \lambda \in \mathsf{Log}\ \forall m \in \mathsf{LogLog}\ \left(\lambda > \lambda_0' \wedge m \le \lambda^k \to \mathsf{IOSecure}^{\lambda^k, \lambda^{-k}}_{\lambda, \lambda^k, m}\right).$$

Fix $\lambda \in \mathsf{Log}$ and $m \in \mathsf{LogLog}$ such that $\lambda > \lambda_0'$ and $m \le \lambda^k$. To prove the formula $\mathsf{IOSecure}^{\lambda^k, \lambda^{-k}}_{\lambda, \lambda^k, m}$, fix two machines $M_0[\mathsf{pub}], M_1[\mathsf{pub}]$ of size λ^k such that $\mathsf{EQRT}(M_0[\mathsf{pub}], M_1[\mathsf{pub}]) = 1$, $n \le m$. Assume that

$$\forall x \in \{0,1\}^n\ M_0[\mathsf{pub}](x) = M_1[\mathsf{pub}](x).$$

We need to prove that

$$\mathsf{iO}(1^\lambda, M_0[\mathsf{pub}], 1^n) \approx_{t,\varepsilon} \mathsf{iO}(1^\lambda, M_1[\mathsf{pub}], 1^n)$$

for $t = \lambda^k$ and $\varepsilon = \lambda^{-k}$. Let ρ, τ be functions and $d \in \mathbb{N}$ be a constant to be determined later, we will prove that

$$\mathsf{iO}_{\rho,\tau}(1^{\lambda+n^d}, M_0[\mathsf{pub}], 1^n) \approx_{t,\varepsilon} \mathsf{iO}_{\rho,\tau}(1^{\lambda+n^d}, M_1[\mathsf{pub}], 1^n).$$

Let $x_1, \ldots, x_{2^{n+1}-1}$ be all strings of length at most n. For every $i \in \{0, 1, \ldots, 2^{n+1} - 1\}$, let $N_i[\mathsf{pub}]$ be the machine

$$N_i[\mathsf{pub}](x_j) = \begin{cases} M_1[\mathsf{pub}](x_j) & j \le i, \\ M_0[\mathsf{pub}](x_j) & j > i, \\ \bot & \text{otherwise.} \end{cases}$$

In particular, we define N_0 as M_0 and $N_{2^{n+1}-1}$ as M_1. The description of N_i is $\mathsf{poly}(|M_0| + |M_1| + n)$ as we need to additionally hardcode $i \le 2^n$ in the program.

By a hybrid argument (see [HJLM26, Lemma 4.15]), it suffices to prove that for every $0 \le i < 2^{n+1} - 1$,

$$\mathsf{iO}_{\rho,\tau}(1^{\lambda+n^d}, N_i[\mathsf{pub}], 1^n) \approx_{t,\varepsilon\cdot 2^{-n}} \mathsf{iO}_{\rho,\tau}(1^{\lambda+n^d}, N_{i+1}[\mathsf{pub}], 1^n), \tag{B.1}$$

where $t = \lambda^k$ and $\varepsilon = \lambda^{-k}$.

We first prove that there is a Turing machine N_Π of size at most $O(1)$ that, given $N_i[\mathsf{pub}]$, $N_{i+1}[\mathsf{pub}]$, $1^{n'}$ for any $n' \le n$, outputs an Extended Frege proof of $[\![N_i[\mathsf{pub}](x) = N_{i+1}[\mathsf{pub}]]\!]_{n'}$ in time $n'^{O(1)}$. The Extended Frege proof consideres whether x is a string with ID $i+1$, or $\le i$, or $> i+1$.

- In the first case, $N_i[\mathsf{pub}](x) = M_0[\mathsf{pub}](x_i)$ and $N_{i+1}[\mathsf{pub}](x) = M_1[\mathsf{pub}](x_i)$. The Extended Frege proof verifies that $M_0[\mathsf{pub}](x_i) = M_1[\mathsf{pub}](x_i)$, which is a true statement with no input variable, and is thus provable in Extended Frege with polynomial size.
- In the last two cases, either both machines output $M_0[\mathsf{pub}](x)$ or both machines output $M_1[\mathsf{pub}](x)$, and the equality is provable in polynomial size.

It can be verified that the proof above is highly uniform, and thus can be generated by a machine N_Π of $\rho := O(1)$ size in time $T(n) := \mathsf{poly}(n)$.

Then, by the security of $\mathsf{iO}_{\rho,\tau}$ as an EF-iO scheme, namely the axiom $\mathsf{ExpEFIOSecure}[\delta, \lambda_0]$, we know that for sufficiently large λ_0',

$$\mathsf{iO}_{\rho,\tau}(1^{\lambda+n^d}, N_i[\mathsf{pub}], 1^n) \approx_{t',\varepsilon'} \mathsf{iO}_{\rho,\tau}(1^{\lambda+n^d}, N_{i+1}[\mathsf{pub}], 1^n),$$

where $t' = 2^{(\lambda+n^d)^\delta} \ge t$ and $\varepsilon' = 2^{-(\lambda+n^d)^\delta}$. Suppose we set $d \ge 2/\delta$ and λ_0' is sufficiently large, we will have $\varepsilon' \le \varepsilon \cdot 2^{-n}$. This will prove Equation (B.1) and thus complete the proof. □

B.3 Correctness of the Indistinguishability Game

Lemma 10.18. *Let $G_0, G_1, \mathcal{U}, \mathcal{U}_{\mathsf{pad}}, T$ be functions that define indistinguishability security, $t(\cdot), \ell(\cdot) : \mathsf{Log} \to \mathsf{Log}$ be PV functions, and $\mathcal{C}$ be the challenger of its indistinguishability game, $\gamma = 1/2$. Let $S(\lambda), \mathcal{E}(\lambda), N(\lambda) = \lambda^{\Omega(1)}$ be formalizable size bounds. Then for every function $\lambda_0(\cdot) : \mathbb{N} \to \mathbb{N}$, there exists a function $\lambda_0'(\cdot) : \mathbb{N} \to \mathbb{N}$ such that*

$$\begin{aligned}&\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”} + (G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T, t, \ell)\text{-}\mathsf{IndSecure}[S, \mathcal{E}, N, \lambda_0]\\ &\quad \vdash (\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)\text{-}\mathsf{GSecure}[S, \mathcal{E}, N, \lambda_0'].\end{aligned}$$

We will use the following lemma that is implicit in a recent work [CLOW26]. This lemma is an axiom in the theory APX_1, and its provability in APC_1 follows immediately from [CLOW26, Theorem 5.9].

Lemma B.2. *$\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$ proves the following. For every $n \in \mathsf{Log}$ and circuit $C : \{0,1\}^n \to \{0,1\}$, let C_0, C_1 be circuits that fix the same input bit of C to be 0 and 1, respectively, then for every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,*

$$\left|\Pr_\delta[C] - \frac{\Pr_\delta[C_0] + \Pr_\delta[C_1]}{2}\right| \leq 2\delta + \beta.$$

Proof of Lemma 10.18. We argue in $\mathsf{APC}_1 + \text{“}\mathsf{prBPP} = \mathsf{prP}\text{”}$. Suppose that

$$(G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T, t, \ell)\text{-}\mathsf{IndSecure}[S, \mathcal{E}, N, \lambda_0]$$

holds, our goal is to prove that for every $k \in \mathbb{N}$, there is a constant λ_0' such that for every $\lambda, n, \tau, s, \varepsilon^{-1} \in \mathsf{Log}$ satisfying that $\lambda > \lambda_0'$, $n \leq N(\lambda)$, $s \leq S(\lambda) \cdot \lambda^k$, $\varepsilon^{-1} \leq \mathcal{E}(\lambda)$, $\tau \geq T(n)$,

$$(\mathcal{C}, \mathcal{U}_{\mathsf{pad}}, T, \gamma, t, \ell)\text{-}\mathsf{GSecure}^{s,\varepsilon}_{n,\tau}$$

holds, where $\mathcal{C}$ is the challenger in indistinguishability game and $\gamma = 1/2$. Fix $\lambda, n, \tau, s, \varepsilon^{-1} \in \mathsf{Log}$. More specifically, we need to prove that for every strings $x \in \{0,1\}^n$ and w such that $\mathcal{U}_{\mathsf{pad}}(x, 1^\tau, w) = 1$, every circuit A of size at most s, and every $\delta^{-1}, \beta^{-1} \in \mathsf{Log}$,

$$\Pr_\delta[D] \leq \gamma + \varepsilon + \delta + \beta, \tag{B.2}$$

where $D(r) := \mathcal{C}(1^\tau, x, w, A; r)$.

Let $\eta^{-1} \in \mathsf{Log}$ be a parameter to be determined later. Let $r_1, r_2 \in \mathsf{Log}$ be the input and output length of $G_0(1^\tau, x, w, \cdot)$. Note that

$$(G_0, G_1, \mathcal{U}_{\mathsf{pad}}, T, t, \ell)\text{-}\mathsf{IndSecure}[S, \mathcal{E}, N, \lambda_0],$$

it follows that $G_0(1^\tau, x, w, \mathcal{U}_{r_1}) \approx_{\tau,\varepsilon} G_1(1^\tau, x, w, \mathcal{U}_{r_1})$. By the definition of indistinguishability, we know that

$$|\Pr_\eta[G_0(1^\tau, x, w, \cdot) = 1] - \Pr_\eta[G_1(1^\tau, x, w, \cdot) = 1]| \leq 3\eta + \varepsilon.$$

By [HJLM26, Proposition 7.3], we know that

$$|\Pr_\eta[G_0(1^\tau, x, w, \cdot) = 1] + \Pr_\eta[G_0(1^\tau, x, w, \cdot) = 1] - 1| \leq 3\eta.$$

By combining these two inequalities, we have

$$\Pr_\eta[G_0(1^\tau, x, w, \cdot) = 0] + \Pr_\eta[G_1(1^\tau, x, w, \cdot) = 1] \leq 1 + 6\eta + \varepsilon. \tag{B.3}$$

Note that $D(r) = \mathcal{C}(1^\tau, x, w, A; r)$, where the random seed r consists of $b \in \{0,1\}$ and a random seed $\mathsf{sd} \in \{0,1\}^{r_1}$ for G_0 or G_1. Let D_b be the circuit the first part is fixed to be $b \in \{0,1\}$. By Lemma B.2, it follows that

$$\left|\Pr_\eta[D] - \frac{\Pr_\eta[D_0] + \Pr_\eta[D_1]}{2}\right| \leq 3\eta. \tag{B.4}$$

Note that D_b is functionally equivalent to "$G_b(1^\tau, x, w, \cdot) = b$" for $b \in \{0,1\}$, and thus by [HJLM26, Proposition 7.2], we know that for every $b \in \{0,1\}$,

$$|\Pr_\eta[D_b] - \Pr_\eta[G_b(1^\tau, x, w, \cdot) = b]| \leq 3\eta. \tag{B.5}$$

By combining Equation (B.3), (B.4), (B.5), we can derive that

$$\begin{aligned}\Pr_\eta[D] &\leq \frac{(\Pr_\eta[G_0(1^\tau, x, w, \cdot) = 0] + \Pr_\eta[G_1(1^\tau, x, w, \cdot) = 1])}{2} + 6\eta \\ &\leq \frac{1}{2} + \varepsilon + 9\eta.\end{aligned}$$

Equation (B.2) then follows by [HJLM26, Proposition 7.1] when we set $\eta := \beta/20$. □

B.4 EF-Equivalence Proof for Lemma 6.16

In the proof of Lemma 6.16, we only provide a high level agreement on the EF-equivalence between $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$. Here, we fill in the formal argument.

Lemma B.3 (Missing piece of Lemma 6.16)**.** *Let* $\mathsf{pub} = (f_q, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}})$*, where* $Z_{\mathsf{ek}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 0; \sigma_{\mathsf{ek}})$*,* $Z_{\mathsf{ch}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, \mathsf{seed}; \sigma_{\mathsf{ch}})$ *and* $Z_{\mathsf{td}} \leftarrow \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, (0, 0^{L\cdot\ell_{\mathsf{ct}}(\lambda)}, 0^{L\cdot\ell_{\mathsf{ct}}(\lambda)}); \sigma_{\mathsf{td}})$*. Let* $M[\mathsf{pub}]$ *be the machine defined in Construction 2, which takes in* n *ciphertexts of form* $(c_{i.0}, c_{i,1}, \pi_i)$*, verifies the NIWI proof* π_i *of* R_{ct} *for each ciphertexts, decrypts* $\mathbf{x}_i$ *from* $c_{i,0}$ *using* sk_0*, and outputs* $f_q(\mathbf{x})$*. Let* $M'[\mathsf{pub}]$ *be the machines that differs with* $M[\mathsf{pub}]$ *on the decryption stage, which decrypts* $\mathbf{x}_i$ *from* $c_{i,1}$ *using* sk_1*.*

Assuming NIWI *admits a* PV *proof of extractability,* Com *admits a* PV *proof of extractability, and* PKE *admits a* PV *proof of correctness, then* $M[\mathsf{pub}]$ *and* $M'[\mathsf{pub}]$ *are* (ρ, τ)*-*EF*-equivalent for* $\rho = \ell \cdot \mathsf{poly}(\lambda)$ *and* τ *being a sufficiently large polynomial.*

Proof. Let D_{b,sk_b} be the PV function that takes in $\vec{\mathsf{ct}} = (\ldots, \mathsf{ct}_i = (c_{i,0}, c_{i,1}, \pi_i), \ldots)$ and outputs $\vec{\mathbf{x}}_b = (\ldots, \mathbf{x}_{i,b} = \mathsf{PKE.Dec}(\mathsf{sk}_b, c_{i,b}), \ldots)$ for $b \in \{0,1\}$. By definition of M and M', for all $\vec{\mathsf{ct}}$, $f_q(D_{(0,\mathsf{sk}_0)}(\vec{\mathsf{ct}})) = f_q(D_{(1,\mathsf{sk}_1)}(\vec{\mathsf{ct}})) \vdash_{\mathsf{PV}} (M[\mathsf{pub}](\vec{\mathsf{ct}}) = M'[\mathsf{pub}](\vec{\mathsf{ct}}))$. Next, let $\boldsymbol{V}_{\mathsf{seed},\mathsf{ch}}$ be the Turing machine that outputs the challenge part ch of $\mathsf{U}_{\mathsf{TM}}(\mathsf{seed}, 1^t)$, and let $\boldsymbol{V}_{\pi_{\mathsf{EF}}}$ be the Turing machine (of size $O(\ell)$) that outputs the EF-proof π_{EF} of the admissibility of ch. For a challenge $\mathsf{ch} = \{\{f_q\}_{q\in[Q]}, \mathcal{I}, \{\mathbf{x}^{(0)}_{i,q}, \mathbf{x}^{(1)}_{i,q}\}_{i\in[n]\setminus\mathcal{I}, q\in[Q]}\}$, define the PV function of membership $\mathsf{Val}_{\mathsf{seed}}(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$ that outputs 1 if $\mathbf{x}^{(b)} = (\ldots, \mathbf{x}^{(b)}_i, \ldots)$ satisfies $(\mathbf{x}^{(0)}_i, \mathbf{x}^{(1)}_i) = (\mathbf{x}^{(0)}_{i,q_i}, \mathbf{x}^{(1)}_{i,q_i})$ for some $q_i \in [Q]$ for all $i \in [n] \setminus \mathcal{I}$ and $\mathbf{x}^0_i = \mathbf{x}^1_i$ for all $i \in \mathcal{I}$, and outputs 0 otherwise. Using $\boldsymbol{V}_{\pi_{\mathsf{EF}}}$, there exists a uniform-EF proof of size $O(\ell)$ for the statement

$$\mathsf{Val}_{\mathsf{seed}}(D_{(0,\mathsf{sk}_0)}(\vec{\mathsf{ct}}), D_{(1,\mathsf{sk}_1)}(\vec{\mathsf{ct}})) = 1 \to (M[\mathsf{pub}](\vec{\mathsf{ct}}) = M'[\mathsf{pub}](\vec{\mathsf{ct}})). \tag{B.6}$$

It remains to argue that there exists a uniform-EF proof for the statement

$$\mathsf{Val}_{\mathsf{seed}}(D_{(0,\mathsf{sk}_0)}(\vec{\mathsf{ct}}), D_{(1,\mathsf{sk}_1)}(\vec{\mathsf{ct}})) = 0 \to (M[\mathsf{pub}](\vec{\mathsf{ct}}) = M'[\mathsf{pub}](\vec{\mathsf{ct}})).$$

Toward this, it suffices to show that the following sentence is provable in PV:

$$\forall(\mathsf{param} = (\mathsf{pub}, \mathsf{sk}_0, \mathsf{sk}_1, \mathsf{seed}), \mathsf{aux} = (\mathsf{td}_{\mathsf{ek}}, \mathsf{td}_{\mathsf{ch}}, \mathsf{td}_{\mathsf{COM}}, \sigma_{\mathsf{ek}}, \sigma_{\mathsf{ch}}, \sigma_{\mathsf{td}}), \vec{\mathsf{ct}}), \tag{B.7}$$
$$R(\mathsf{param}, \mathsf{aux}) = 1 \vdash_{\mathsf{PV}} \left(\mathsf{Val}_{\mathsf{seed}}(D_{(0,\mathsf{sk}_0)}(\vec{\mathsf{ct}}), D_{(1,\mathsf{sk}_1)}(\vec{\mathsf{ct}})) = 0 \to (M[\mathsf{pub}](\vec{\mathsf{ct}}) = M'[\mathsf{pub}](\vec{\mathsf{ct}}))\right),$$

where $\mathsf{td}_{\mathsf{ek}}$, $\mathsf{td}_{\mathsf{ch}}$ represents the extraction trapdoor for the NIWI crs $\mathsf{crs}_{\mathsf{ek}}$, $\mathsf{crs}_{\mathsf{ch}}$ respectively, $\mathsf{td}_{\mathsf{COM}}$ represents the extraction trapdoor for the commitment scheme, and R is the relation describing the honest generation of the elements in pub and aux. In particular, R outputs 1 if and only if all of the following consistency checks hold:

- Parse $\mathsf{pub} = (f_q, \mathsf{crs}_{\mathsf{ek}}, \mathsf{crs}_{\mathsf{ct}}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{crs}_{\mathsf{com}}, Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}})$.
- Check whether $(\mathsf{crs}_{\mathsf{ek}}, \mathsf{td}_{\mathsf{ek}})$, $(\mathsf{crs}_{\mathsf{ch}}, \mathsf{td}_{\mathsf{ch}})$, and $(\mathsf{crs}_{\mathsf{com}}, \mathsf{td}_{\mathsf{COM}})$ are consistent NIWI and commitment trapdoors, respectively. It is valid to assume these checks are efficiently computable, as one can always set the random coins for generating the crs (or a seed for pseudorandomness) to be the trapdoor itself.
- Check whether $(\mathsf{pk}_b, \mathsf{sk}_b)$ for $b \in \{0, 1\}$ are valid PKE key pairs.
- Check that the following commitments are honestly generated:
  - $\mathsf{Com}(\mathsf{crs}_{\mathsf{COM}}, 0; \sigma_{\mathsf{ek}}) = Z_{\mathsf{ek}}$.
  - $\mathsf{Com}(\mathsf{crs}_{\mathsf{COM}}, \mathsf{seed}; \sigma_{\mathsf{ch}}) = Z_{\mathsf{ch}}$.
  - $\mathsf{Com}(\mathsf{crs}_{\mathsf{COM}}, (0, 0^{L\cdot\ell_{\mathsf{ct}}(\lambda)}, 0^{L\cdot\ell_{\mathsf{ct}}(\lambda)}); \sigma_{\mathsf{td}}) = Z_{\mathsf{td}}$.

Let $\mathsf{Ver}_{M,\mathsf{param}}(\vec{\mathsf{ct}})$ be the PV function describing the NIWI verification step for relation R_{ct} in the machine M and M'. By the PV proof of extractability for NIWI, it holds that

$$\forall(\mathsf{param}, \mathsf{aux}, \vec{\mathsf{ct}}, i),\ \mathsf{NIWI.Ext}(\mathsf{td}_{\mathsf{ct}}, \pi_i) = (\mathbf{x}_0, r_0, \mathbf{x}_1, r_1, \pi^*_{\mathsf{ek}}, \sigma^*_{\mathsf{ch}}, \sigma^*_{\mathsf{td}}),$$
$$(\mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, (i, c_{i,0}, c_{i,1}); \sigma^*_{\mathsf{td}}) = Z_{\mathsf{td}})$$
$$R(\mathsf{param}, \mathsf{aux}) = 1 \vdash_{\mathsf{PV}} \mathsf{Ver}_{M,\mathsf{param}}(\vec{\mathsf{ct}}) = 1 \to \vee \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_0, \mathbf{x}_0; r_0) = c_{i,0} \wedge \mathsf{Enc}(\mathsf{pk}_1, \mathbf{x}_1; r_1) = c_{i,1} \\ \wedge \begin{pmatrix} (\mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, \mathsf{seed}; \sigma^*_{\mathsf{ch}}) = Z_{\mathsf{ch}} \wedge \mathsf{Val}_{\mathsf{seed}}(i, \mathbf{x}_0, \mathbf{x}_1)) \\ \vee \mathsf{Ver}(\mathsf{crs}_{\mathsf{ek}}, (i, Z_{\mathsf{ek}}, Z_{\mathsf{ct}}), \pi^*_{\mathsf{ek}}) \end{pmatrix} \end{pmatrix}$$

where $\mathsf{Val}_{\mathsf{seed}}(i, \mathbf{x}_0, \mathbf{x}_1)$ is the PV function for membership defined similarly as previous, but restricted to the i-th index. Combined with the PV proof of extractability for Com and NIWI for R_{ek}, we have that

$$\forall(\mathsf{param}, \mathsf{aux}, \vec{\mathsf{ct}}, i),\ \mathsf{NIWI.Ext}(\mathsf{td}_{\mathsf{ct}}, \pi_i) = (\mathbf{x}_0, r_0, \mathbf{x}_1, r_1, \pi^*_{\mathsf{ek}}, \sigma^*_{\mathsf{ch}}, \sigma^*_{\mathsf{td}}),$$
$$(\mathsf{Ext}(\mathsf{td}_{\mathsf{com}}, Z_{\mathsf{td}}) = (i, c_{i,0}, c_{i,1}))$$
$$R(\mathsf{param}, \mathsf{aux}) = 1 \vdash_{\mathsf{PV}} \mathsf{Ver}_{M,\mathsf{param}}(\vec{\mathsf{ct}}) = 1 \to \vee \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_0, \mathbf{x}_0; r_0) = c_{i,0} \wedge \mathsf{Enc}(\mathsf{pk}_1, \mathbf{x}_1; r_1) = c_{i,1} \\ \wedge \begin{pmatrix} (\mathsf{Ext}(\mathsf{td}_{\mathsf{com}}, Z_{\mathsf{ch}}) = \mathsf{seed} \wedge \mathsf{Val}_{\mathsf{seed}}(i, \mathbf{x}_0, \mathbf{x}_1)) \\ \vee \mathsf{Ext}(\mathsf{td}_{\mathsf{com}}, Z_{\mathsf{ek}}) = 1 \end{pmatrix} \end{pmatrix}$$

Following the definition of R which checks the generation of $(Z_{\mathsf{ek}}, Z_{\mathsf{ch}}, Z_{\mathsf{td}})$, along with the PV proof of extractability of Com and correctness of PKE, we have that

$$\forall(\mathsf{param}, \mathsf{aux}, \vec{\mathsf{ct}}, i),$$
$$R(\mathsf{param}, \mathsf{aux}) = 1 \vdash_{\mathsf{PV}} \mathsf{Ver}_{M,\mathsf{param}}(\vec{\mathsf{ct}}) = 1 \to \begin{pmatrix} \mathsf{Ext}(\mathsf{td}_{\mathsf{com}}, Z_{\mathsf{ch}}) = \mathsf{seed} \\ \wedge \mathsf{Val}_{\mathsf{seed}}(i, \mathsf{Dec}(\mathsf{sk}_0, c_{i,0}), \mathsf{Dec}(\mathsf{sk}_1, c_{i,1})) = 1 \end{pmatrix},$$

which implies that

$$\forall(\mathsf{param}, \mathsf{aux}, \vec{\mathsf{ct}}),\ R(\mathsf{param}, \mathsf{aux}) = 1 \vdash_{\mathsf{PV}} \mathsf{Val}_{\mathsf{seed}}(D_{(0,\mathsf{sk}_0)}(\vec{\mathsf{ct}}), D_{(1,\mathsf{sk}_1)}(\vec{\mathsf{ct}})) = 0 \to \mathsf{Ver}_{M,\mathsf{param}}(\vec{\mathsf{ct}}) = 0$$

Finally, M and M' both aborts and output $\perp$ if $\mathsf{Ver}_{M,\mathsf{param}}(\vec{\mathsf{ct}}) = 0$, in which case we also have $M[\mathsf{pub}](\vec{\mathsf{ct}}) = M'[\mathsf{pub}](\vec{\mathsf{ct}})$. This gives the desired proof for Equation (B.7), implying a (ρ', τ') uniform-EF proof of the right hand side statement on honest parameters of size $\rho' = |\mathsf{aux}| + O(1) = \ell \cdot \mathsf{poly}(\lambda)$ and τ' a sufficiently large polynomial. Combined with the $(O(\ell), \tau'')$ uniform-EF proof for Equation (B.6), we have a (ρ, τ) uniform-EF proof for the equivalence between $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$. □

Remark 16 (Formalization in PV_1)**.** By a careful inspection of the proof, it can be observed that theorem is proved by first writing a PV proof and then translate it into a uniform EF proof via propositional translation. Therefore, by Theorem 3.1, the (ρ, τ)-EF-equivalence of $M[\mathsf{pub}]$ and $M'[\mathsf{pub}]$ can be proved in PV_1.

C Justifications and Cryptanalysis of Proof Complexity Generators

It can be observed that Assumption 3, as well as its counterpart against weak bounded theories, are closely related to strong circuit lower bounds circuits in several aspects.

Attacks from Circuit Lower Bounds. Suppose that the generator P outputs a circuit C_n such that every string in the range of C_n can be computed by a small circuit (called *local PCGs* for simplicity), the security assumption violates circuit lower bounds that are generally considered plausible. Formally:

Proposition C.1. *Suppose that there is a constant $\varepsilon \in (0,1)$ such that for every $y \in \mathsf{Range}(C_m)$, y is the prefix of the truth-table of a circuit $D : \{0,1\}^n \to \{0,1\}$ of size m^ε, where the input length of D is $n = O(\log m)$. Then Assumption 3 for the generator P is false if $\mathsf{E} \nsubseteq \text{i.o.-}\mathsf{SIZE}[2^{\varepsilon n}]$.*

Proof. Suppose that $\mathsf{E} \nsubseteq \text{i.o.-}\mathsf{SIZE}[2^{\varepsilon n}]$. Let $L \in \mathsf{E} \setminus \mathsf{SIZE}[2^{\varepsilon n}]$ and M be the exponential-time deterministic machines for L. Let n_0 be the largest input length on which $L \cap \{0,1\}^n$ does not have $2^{\varepsilon n}$ sized circuits. For any $m = 2^n$, consider the following polynomial-time deterministic algorithm $\mathcal{A}(1^m, y)$: It accepts if $m > 2^{n_0}$ and for every $x \in \{0,1\}^n$, $M(x) = y_x$, i.e., y is the truth-table of $L \cap \{0,1\}^n$.

It is clear that for large m, the algorithm will accept $y \in \{0,1\}^m$ if and only if it is the truth-table of $L \cap \{0,1\}^n$. Denote the string to be y_m. By definition, y_m is not the truth-table of circuits of size $2^{\varepsilon n}$, and thus $y, \notin \mathsf{Range}(C_m)$ by the assumption. Subsequently, for every sufficient m, $\mathcal{A}(1^m, y)$ will accept a string not in the range of C_m and reject any other strings. This violates Assumption 3 for the generator P. □

Note that $\mathsf{E} \nsubseteq \mathsf{SIZE}[2^{\varepsilon n}]$ is generally considered plausible, and is the main evidence for general derandomization [NW94, IW97]. Moreover, it is easy to see that the proof above relativizes in the sense that if we assume $\mathsf{E} \nsubseteq \mathsf{SIZE}^{\mathcal{O}}[2^{\varepsilon n}]$ for any oracle $\mathcal{O}$, Assumption 3 is insecure if for every $y \in \mathsf{Range}(C_m)$, y is the prefix of the truth-table of an $\mathcal{O}$-oracle circuit D of size m^ε. If one believes the strong circuit lower bound,

$$\mathsf{E} \nsubseteq \mathsf{SIZE}^{\mathsf{PH}}[2^{\varepsilon n}],$$

the circuit C_m cannot always output strings that admit $2^{\varepsilon n}$-size PH-oracle circuits.

An example of a local PCG is the truth-table generator: given a circuit C of description size $s(n) < 2^n$, it outputs the truth-table of C. For a notion called "$\vee$-hardness" that is stronger than PCG, this generator is indeed universal [Kra24].

Attack using Cryptographic Axioms. The above attack can be used to refute local PCGs against feasible cryptography for certain parameter regimes. This is because cryptographic axioms generally APC_1-provably imply hardness of SAT (as SAT is NP-hard). As $\mathsf{SAT} \in \mathsf{E}$, we might be able to prove hardness for E in the bounded theory and subsequently break the PCG following the proof of Proposition C.1.

Below we demonstrate an example. If a theory proves subexponential hardness of SAT, it could also prove the existence of hard truth-tables.

Proposition C.2. *Let $\mathcal{T}$ be an extension of PV_1, and suppose that $\mathcal{T}$ proves "$\mathsf{SAT} \notin \text{i.o.-}\mathsf{SIZE}[2^{n^\varepsilon}]$", formalized as the following $\forall\Sigma^b_2(\mathsf{PV})$ sentence:*

> *for every $n \in \mathsf{Log}$, $n > n_0$, and any circuit $C : \{0,1\}^n \to \{0,1\}$ of size at most 2^{n^ε}, there exists a satisfiable formula $\phi \in \{0,1\}^n$ such that either $C(\phi) = 1$ but ϕ is unsatisfiable, or $C(\phi) = 0$ but ϕ is satisfiable;*

where n_0, ε are constants. Then there exists a function $t(n) = 2^{(\log n)^{\Omega(1)}}$ such that $\mathcal{T} \vdash (\text{i.o.})y\ \forall C$ of size at most $t(|y|)$, y is not a prefix of the truth-table of C.

Proof Sketch. Let $\psi(y)$ be the formula that y is a hard truth-table. To prove $\mathcal{T} \vdash (\text{i.o.})y\ \psi(y)$, we define the polynomial-time algorithm $V(y, w)$ that ignores w and accepts iff the truth-table of SAT on input length $m := \lfloor \log_2 |y| \rfloor$ is a prefix of y. This is a polynomial-time algorithm as $\mathsf{SAT} \in \mathsf{DTIME}[2^{O(n)}]$. It then suffices to show the completeness and provable soundness properties:

- (*Completeness*). It is clear that for sufficiently large $n \in \mathbb{N}$, there exists $y \in \{0,1\}^n$ such that $V(y, w)$ accepts for any w, as there must be a string whose prefix is the truth-table of SAT.
- (*Provable Soundness*). Assume that V is formalized straightforwardly. Then $\mathcal{T}$ proves that for every y, w such that $V(y, w) = 1$, letting $m = \lfloor \log_2 |y| \rfloor$, $y_{\le 2^m}$ is the truth-table of SAT on input length m. As $\mathcal{T}$ proves $\mathsf{SAT} \notin \text{i.o.-}\mathsf{SIZE}[2^{n^\varepsilon}]$, we can further show that $\mathcal{T}$ proves there is no circuit of size 2^{m^ε} whose truth-table is $y_{\le 2^m}$. The proposition then follows by taking $t(n) = 2^{(\log n)^{\Omega(1)}}$ so $t(|y|) \le 2^{m^\varepsilon}$.

This completes the proof. □

Note that the parameter regimes that we use prevent this attack in two different ways. We do not require the PCG to be local. More importantly, when we use a subexponential assumption as an axiom in $\mathcal{T}$, we choose the seed length of the PCG to be $o(m)$ (rather than $m^{o(1)}$); in this way, even if we fix the PCG to be the truth-table generator, the attack still cannot apply as the hardness assumption is not strong enough to rule out $2^{o(n)}$-sized circuits computing SAT.

Adversary with Short Advice. As mentioned in Section 4.1, the assumption is trivially false if the adversary is non-uniform, as it can simply hard-code a sequence of strings outside of the range of C_m. It is worth mentioning that under strong circuit lower bounds, Section 4.1 is insecure if the adversary is allowed to take advice of $O(\log m)$ bits. Formally:

Proposition C.3. *Assuming $\mathsf{E} \not\subseteq \mathsf{SIZE}^{\mathsf{SAT}}[2^{\varepsilon n}]$ for a constant $\varepsilon \in (0,1)$, Assumption 3 is false against an deterministic polynomial-time adversary that takes an advice string α_m of length $O(\log m)$ that only depends on m.*

This is indeed a simple corollary of the following hitting set construction due to Klivans and van Melkebeek [KvM02] (following [NW94, IW97]).

Theorem C.4 ([KvM02]). *Suppose that* $\mathsf{E} \not\subseteq \mathsf{SIZE}^{\mathsf{SAT}}[2^{\varepsilon n}]$ *for a constant* $\varepsilon \in (0,1)$*, there is a polynomial-time algorithm* G *such that the following holds. Let* $L \in \mathsf{P}^{\mathsf{NP}}$ *be a language such that for every input length* m*,* $\Pr[x \in L] \geq 0.1$ *for uniformly random* $x \in \{0,1\}^m$*. Then for infinitely many* $m \in \mathbb{N}$*,* $G(1^m)$ *outputs a set of* m*-bit strings that contain at least one string* $y \in L \cap \{0,1\}^m$*.*

Proof of Proposition C.3. Fix any proof complexity generator P with seed length $\ell(m) < m$, and let $C_m = P(1^m)$. Let L_P be the following language: $y \in L_P$ if and only if $y \notin \mathsf{Range}(C_m)$. It is clear that $L_P \in \mathsf{P}^{\mathsf{NP}}$. Therefore, under the assumption that $\mathsf{E} \not\subseteq \mathsf{SIZE}^{\mathsf{SAT}}[2^{\varepsilon n}]$, there is a polynomial-time algorithm G and a sequence $\{m_i\}_{i\in\mathbb{N}}$ of input lengths such that $G(1^{m_i}) \cap L_P \neq \varnothing$ (see Theorem C.4).

In such case, we can define the advice string on the input length m_i as the index to the m-bit string $y \in G(1^{m_i})$ such that $y \in L$. Then the following algorithm M_G breaks the proof complexity generator: On the input (m, y), it takes the advice string α_m as described above, runs $G(1^m)$, and accepts if and only if the α_m-th string given by $G(1^m)$ is exactly y. This is because on the input lengths $\{m_i\}_{i\in\mathbb{N}}$, M_G (given correct advice) will accept and only accept one string that is in L_P (or equivalently, not in the range of P). □