Constructing Witness PRF and Offline Witness Encryption Without Multilinear Maps

Tapas Pal, Ratna Dutta
Department of Mathematics,
Indian Institute of Technology Kharagpur,
Kharagpur-721302, India
tapas.pal@iitkgp.ac.in,ratna@maths.iitkgp.ernet.in

Abstract

Witness pseudorandom functions (witness PRFs), introduced by Zhandry [Zha16], was defined for an NP language L and generate a pseudorandom value for any instance x. The same pseudorandom value can be obtained efficiently using a valid witness w for $x \in L$. Zhandry built a subset-sum encoding scheme from multilinear maps and then converted a relation circuit corresponding to an NP language L to a subset-sum instance to achieve a witness PRF for L. The main goal in developing witness PRF in [Zha16] is to avoid obfuscation from various constructions of cryptographic primitives. Reliance on cryptographic tools built from multilinear maps may be perilous as existing multilinear maps are still heavy tools to use and suffering from many non-trivial attacks.
In this work, we give constructions of the following cryptographic primitives without using multilinear maps and instantiating obfuscation from randomized encoding:

– We construct *witness PRFs* using a puncturable pseudorandom function and sub-exponentially secure randomized encoding scheme in common reference string (CRS) model. A sub-exponentially secure randomized encoding scheme in CRS model can be achieved from a sub-exponentially secure public key functional encryption scheme and learning with error assumptions with sub-exponential hardness.
– We turn our witness PRF into a *multi-relation witness PRF* where one can use the scheme with a class of relations related to an NP language.
– Furthermore, we construct an *offline witness encryption* scheme using any extractable witness PRF. The offline witness encryption scheme of Abusalah et al. [AFP16] was built from a plain public-key encryption, a statistical simulation-sound non-interactive zero knowledge (SSS-NIZK) proof system and obfuscation. In their scheme, a(n) SSS-NIZK proof is needed for the encryption whose efficiency depends on the underlying public key encryption. We replace SSS-NIZK by extractable witness PRF and construct an offline witness encryption scheme. More precisely, our scheme is based on a public-key encryption, a witness PRF and employs a sub-exponentially secure randomized encoding scheme in CRS model instantiating obfuscation. Our offline witness encryption can be turned into an *offline functional witness encryption* scheme where decryption releases a function of a message and witness as output.

Keywords: Witness PRF, Offline witness encryption, Randomized encoding.

1 Introduction

Witness PRF. *Witness pseudorandom function* (witness PRF) is a relatively new cryptographic primitive introduced by Zhandry [Zha16] to avoid obfuscation in various cryptographic applications like multiparty non-interactive key exchange without trusted setup, poly-many hardcore bits, re-usable witness encryption, Rudich secret sharing for monotone NP language and fully distributed broadcast encryption that do not need to hide a programme P completely. For instance, the Boneh-Zhandry [BZ17] protocol of multiparty non-interactive key exchange without a trusted setup needs a program P to be obfuscated where P computes a pseudorandom function using a secret key on its inputs. Zhandry has shown how obfuscation can be avoided by introducing witness PRF.

A witness PRF for an NP language L enables one to compute a pseudorandom function f on a statement x without a secret key if (s)he has a valid witness for $x \in L$ where $f(x)$ is indistinguishable from a random element if $x \notin L$. More specifically, witness PRF first generates a pair of keys $(\mathsf{fk}, \mathsf{ek})$ based on a relation circuit R corresponding to an NP language L where fk is the function secret key and ek is the function public key. A user with fk computes a pseudorandom value $\mathrm{F}(\mathsf{fk}, x)$ for any input x while a witness holder can obtain the same pseudorandom value using $\mathrm{Eval}(\mathsf{ek}, x, w)$ if w is a valid witness for $x \in L$.

Witness PRF is closely related to constrained PRFs and smooth projective hash functions (SPHFs). In constrained PRF, we can generate multiple keys for different circuits. In witness PRF, we generate only one key ek for a given relation circuit R in the setup phase. Existing constructions of SPHFs cannot handle arbitrary NP languages while witness PRFs support any NP language. There is another variant of witness PRF which is extractable in nature. In *extractable witness PRF*, existence of an adversary capable of distinguishing $\mathrm{F}(\mathsf{fk}, x)$ from a random value implies existence of a polynomial time extractor which produces a witness for the instance x if $x \in L$. There is another notion related to witness PRF which is flexible for a class of relations with a specified size bound and is called *multi-relation witness PRF*. Here we generate different keys for each relation and the security is based on the indistinguishability of $\mathrm{F}(\mathsf{fk}, x)$ from a random element if x has no valid witness relative to any of the queried relations.

Applications of witness PRF. There are many efficient tools that can be built from witness PRFs and extractable witness PRFs. We mention the schemes where Zhandry uses witness PRFs fending off obfuscation [Zha16]:

- Boneh and Zhandry [BZ17] designed the first multi-party key exchange (MIKE) protocol (for n-users with $n > 3$) using obfuscation that does not require a trusted party or setup. Obfuscation can be replaced with witness PRFs to obtain such an MIKE.
- Bellare et al. [BST14] constructed a hardcore function of arbitrary output size for any one-way function using differing inputs obfuscation (or extractable obfuscation). Witness PRFs suffices here.
- It has been seen that obfuscation implies *witness encryption* [GGSW13]. Zhandry introduced and constructed re-usable witness encryption from witness PRFs. Re-usable witness encryption has a special feature of producing very short ciphertexts with size proportional to the security parameter and independent of the size of the relation. This re-usable witness encryption can be used in the transformation of [GGSW13] to get an attribute-based encryption with similar short ciphertexts.
- Witness PRFs also replace obfuscation in secret sharing [KNY17] and fully distributed broadcast encryption scheme [Zha16].

Constructing witness PRF of [Zha16]. The only existing construction of witness PRF [Zha16] is based on subset-sum encoding scheme. A subset-sum encoding scheme corresponding to a (multi-)set of integers S is capable of encoding any integer secretly such that a public evaluation function on input a subset T of S can compute an encoding of the integer $t =$

$\sum_{i \in T} i$. In [Zha16], a new notion of subset-sum encoding is introduced from multilinear maps [GGH13, CLT13] and the security is based on the fact that if there does not exist a subset of S for which the target sum t is achieved then the encoding $\hat{t}$ is indistinguishable from a random element. The hardness of this problem gives rise to a new complexity assumption (based on multilinear maps) which was named as the *multilinear subset-sum Diffie-Hellman assumption*. Witness PRF is instantiated from the subset-sum encoding. Consequently, it is based on multilinear maps and the security depends upon *multilinear subset-sum Diffie-Hellman assumption*. We emphasize that the pseudorandom value of the witness PRF cannot be immediately computed for any instance x, rather a reduction procedure is followed where an instance x of an NP language L is converted into a subset-sum instance. The reduced subset-sum instance depends on L and the size of instance x.

All currently known constructions of multilinear maps [GGH13, CLT13] are only approximations to the ideal multilinear maps and the noise increases with the number of multiplications and pairing operations. The multilinearity level of witness PRF increases with the number of gates used in the relation circuit corresponding to the NP language. Existing witness PRF [Zha16] is approximate in the sense that the underlying subset-sum encoding is based on multilinear maps and thereby approximate and noisy. Furthermore, complications arise when the size of the relation circuit grows.

Zhandry showed the hardness of the *multilinear subset-sum Diffie-Hellman assumption* in the generic multilinear map model [Zha16]. The recent line of attacks on multilinear maps [CLT14a, CHL+14, BWZ14b, GHMS14b] breaks many useful assumptions and hence threats to the cryptosystems where security is based on complexity assumptions related to multilinear maps. Therefore, one may not want to rely on cryptographic tools that are instantiated from multilinear maps.

Witness encryption. *Witness encryption* (WE) was introduced by Garg et al. [GGSW13]. There are many applications of WE in secret sharing, identity based encryption, attribute-based encryption, asymmetric password based encryption, differing inputs obfuscation ([GGSW13, GKP+13b, BH15, BCP14]). In a plain public-key encryption (PKE) scheme, we encrypt data using a public key and decryption is possible if the corresponding secret key is known. WE enables us to encrypt a message with respect to an instance x of an NP language L. Only a witness holder can recover the original message from the ciphertext if he has a valid witness w for $x \in L$. *Functional witness encryption* was introduced by Boyle et al. [BCP14] where a decrypter can only learn a function of the message if a valid witness for the instance is known. The equivalence of functional WE and differing inputs obfuscation was also observed in [BCP14].

WE with an additional setup phase is called *offline witness encryption* (OWE) [AFP16]. In OWE, the heavy-duty part is done by a trusted third party in an offline phase making encryption more efficient than the existing WE constructions. The only OWE construction [AFP16] uses a standard public key encryption and a statistically simulation-sound non-interactive zero knowledge (SSS-NIZK) proof system to produce a ciphertext and an obfuscated circuit created in the setup phase is used for decryption.

Mostly, WE have been constructed using either multilinear maps or using obfuscation directly. Impracticality or computationally expensive nature of multilinear maps and obfuscation have made all these WE schemes unusable in practical devices.

Our contribution and technical overview. In this work, we construct a witness PRF without using multilinear maps and an offline witness encryption without SSS-NIZK. Our construction of witness PRFs is inspired by the puncturable programming technique introduced by Sahai and Waters [SW14]. They used indistinguishability obfuscation ($i\mathcal{O}$) with puncturable pseudorandom function to built many interesting cryptographic tools. The job of $i\mathcal{O}$ is to make a class of circuits $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ unintelligible in such a way that for any circuit $C \in \mathcal{C}_\lambda$ for some $\lambda \in \mathbb{N}$, we have $i\mathcal{O}(1^\lambda, C)(x) = C(x)$. The security demands that given two equivalent circuits (or Turing machines) $C_0, C_1 \in \mathcal{C}_\lambda$, one cannot distinguish between $i\mathcal{O}(1^\lambda, C_0)$ and $i\mathcal{O}(1^\lambda, C_1)$. Instead of using $i\mathcal{O}$ directly, we look into recent developments on $i\mathcal{O}$ from functional encryption [AJ15, BV15, AJ15, BNPW16, KNT17, LPST16b] which is a relatively weaker primitive. In our constructions of witness PRFs and offline witness encryption, we integrate the technique of getting $i\mathcal{O}$ from randomized encoding scheme in common reference string (CRS) model [LPST16b].

We built our witness PRFs by coupling a puncturable pseudorandom function (pPRF) and a sub-exponentially secure sub-linearly compact randomized encoding (RE) scheme in CRS model. The functional secret key fk is a pPRF key K and the functional evaluation key ek consists of $\overrightarrow{\mathrm{crs}} = \{\mathrm{crs}_i\}_{i=0}^n$ and a randomized encoding of an input less Turing machine $\Pi[\overrightarrow{pk}_1, E, \epsilon, \alpha]$ (defined in Remark 2, section 2) where the hardcoded elements are $\overrightarrow{pk}_1 = \{pk_j\}_{j=1}^n$, a circuit E, the null string ϵ and a randomly chosen bit-string α. The $n+1$ pairs of common reference string-encoding key (crs_i, pk_i) for $i = 0, 1, 2, \ldots, n$ are generated in the setup phase of RE. Here n denotes the total size of the instance-witness pair. The circuit E on taking input an instance-witness (x, w) pair, verifies that w is a valid witness for the instance x and then outputs the evaluation of pPRF with input x using the key K.

Our witness PRF computes the pseudorandom value corresponding to some x as the pPRF evaluation of x using the functional secret key fk. A valid witness holder can recover the same pseudorandom value using the fucntional evaluation key ek by computing $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}](x, w)$ where $\mathcal{G}$ is the special circuit (defined in Figure 6, section 2) that recursively computes the evaluation phase of RE to achieve $E(x, w)$.

A pPRF can be constructed from one-way functions [GGM86, BW13, BGI14] and a sub-exponentially secure sub-linear compact RE scheme in CRS model can be instantiated from a sub-exponentially secure weakly sub-linear compact public key functional encryption (PKFE) scheme and assuming sub-exponential hardness of learning with error (LWE) assumption [BNPW16, LPST16b]. We achieve the following result.

Theorem 1. (Informal) *Assuming* LWE *with sub-exponential hardness and the existence of sub-exponentially secure one-way functions, if there exists a weakly sub-linear compact public key functional encryption* (PKFE) *scheme with sub-exponential security, then there exists a secure witness* PRF *scheme.*

We note the following advantages of our scheme over the witness PRF scheme of Zhandry [Zha16]:

- Zhandry used subset-sum encoding to build witness PRF and subset-sum encoding is constructed from multilinear maps. Our scheme is established from a puncturable pseudorandom function and a randomized encoding scheme in CRS model which does not use multilinear maps. Therefore our scheme is more reliable in light of the recent attacks on multilinear maps. Security of our witness PRF is based on sub-exponentially secure one-way functions, sub-exponential hardness of LWE assumption and sub-exponentially secure PKFE. Note that PKFE are well studied and secure tools compared to existing multilinear maps which are noisy, approximation to ideal multilinear maps and vulnerable to many attacks.

- The security proof of the witness PRF of [Zha16] relies on a newly suggested *multilinear subset-sum Diffie-Hellman assumption* which is instance dependent non-standard assumption with hardness proved in the generic multilinear maps model. In contrast, we achieve security by showing indistinguishability between hybrid sequences. Our proof is instance independent and does not rely on any such non-standard assumptions.

- The multilinearity level increases at least linearly with the number of gates used in the relation circuit of the underlying NP language in the construction of [Zha16] and the efficiency of their scheme also decreases with the size of the relation circuit. On the other hand, our scheme uses a randomized encoding scheme that essentially executes decryption procedure of a PKFE scheme for $(n+1)$ times [LPST16b] where n denotes the total size of the instance and witness. Our construction provides more efficient witness PRF evaluation than the scheme of Zhandry.

As an application of witness PRF, we present an offline witness encryption (OWE) scheme encouraged by the construction of Abusalah et al.[AFP16]. They used a standard public key encryption (PKE) scheme to encrypt a message m together with an instance x of an NP language L twice with two different randomness to produce two different ciphertexts c_1 and c_2. Besides, they generate a(n) SSS-NIZK proof π of the statement that c_1, c_2 *encrypt the same message-instance pair*. The resulting ciphertext of their OWE is (x, c_1, c_2, π). They have instantiated the encryption of their OWE using an ElGamal encryption scheme for the PKE and established a(n) SSS-NIZK proof π of the statement that "*two ElGamal ciphertexts* c_1, c_2 *encrypt the same message*" via Gorth-Sahai proofs (GS-proofs) [GS08]. Here we note that GS-proofs are efficient non-interactive witness-indistinguishable proofs for some specific languages involving pairing product equations, multi-scaler multiplication equations or quadratic equations over some groups. The ElGamal ciphertexts can be represented in a way to get a set of pairing product equations that supports the above statement and a(n) SSS-NIZK proof can be ensured using the GS-proofs for those equations. Therefore, for practical use of the OWE scheme of [AFP16], we need to carefully choose the PKE scheme so that a(n) SSS-NIZK proof can be achieved through the GS-proofs. If some other PKE scheme is used for more efficiency or security then one may face problem in generating such a proof for the abovementioned statement as there exists no efficient SSS-NIZK proof system for any general relation (according to our knowledge).

We try to fix this limitation of the OWE scheme of [AFP16] by replacing SSS-NIZK with any extractable witness PRF which can efficiently handle any relation. Our OWE is built with a plain PKE scheme and a secure witness PRF for encryption. In the setup phase, we generate two pairs of secret-public keys $(\mathrm{SK}_1, \mathrm{PK}_1)$, $(\mathrm{SK}_2, \mathrm{PK}_2)$ from the key generation algorithm of the PKE and a pair of functional secret key, evaluation key $(\mathsf{fk}, \mathsf{ek})$ for the witness PRF which corresponds to the NP statement that *two given ciphertexts obtained from the PKE scheme encrypt the same message*. The public parameters for encryption consist of $\mathrm{PK}_1, \mathrm{PK}_2$ and ek. Our OWE encryption first computes two ciphertexts c_1, c_2 encrypting the same instance-message pair (x, m) under the two public keys $\mathrm{PK}_1, \mathrm{PK}_2$, then utilizing the evaluation key ek executes the evaluation process of witness PRF to produce a pseudorandom string y for the statement that c_1, c_2 encrypt the same message. The OWE ciphertext components for an NP language L are c_1, c_2, x and y.

For decryption, we need to compute a circuit C in the setup phase of OWE. The circuit C on input (c_1, c_2, x, y) and a witness w for $x \in L$ works as follows:

- Use the functional secret key fk for the statement that c_1, c_2 encrypt the same message to get a value y' and check if $y' = y$.

- Check whether w is a valid witness for the statement x.

- If both the checks pass, then decrypt c_1 using SK_1 and obtain (x', m').

– Output m' if $x = x'$.

As in the case of evaluation procedure of our witness PRF scheme, here also we pick the RE scheme in CRS model of [LPST16b] to get a randomized encoding of an input less Turing machine $\Pi[\overrightarrow{pk}_1, C, \epsilon, \alpha]$ and use the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha], \overrightarrow{\text{crs}}]$ that corresponds to the circuit C as the public parameter for decryption. The decryption algorithm of our OWE scheme computes the circuit $\mathcal{G}$ on an input (c, w) to recover the original message m if w is a valid witness for $x \in L$, where $c = (c_1, c_2, x, y)$. In the OWE scheme of [AFP16], they managed decryption with an obfuscated circuit and claimed that the decryption phase is less efficient than that of the reusable WE scheme of [Zha16]. We employ the randomized encoding technique of [LPST16b] to achieve an efficient decryption for our OWE scheme where the decryption time is polynomial in the size of the circuit C and the total size of ciphertext and witness.

To prove our OWE scheme is secure, we need to consider the extractable security of the underlying witness PRF scheme. Unfortunately, we do not know how to construct a polynomial time extractor which can extract a witness using a challenge statement x^*, a witness PRF value $\mathrm{F}(\mathsf{fk}, x^*)$ and some auxiliary information. Extractable security requires that an adversary cannot distinguish $\mathrm{F}(\mathsf{fk}, x^*)$ from a random value unless a valid witness to the statement x^* is known to him. In this application of witness PRF for OWE, we note that an adversary does not have a witness for the statement "c_1, c_2 *encrypt the same message*" as the randomness used at the time of encryption which is a part of the witness is computationally hidden in the ciphertexts of the PKE. In this context, we note that the witness PRF of [Zha16] was assumed to be extractable under the *extracting subset-sum Diffie-Hellman assumption*, rather fabricating an extractor for the witness PRF. Assuming our witness PRF is extractable we arrive at the following result.

Theorem 2. (Informal) *Assuming existence of sub-exponentially secure one-way functions, a secure public-key encryption* (PKE)*, an extractable witness* PRF *and a sub-exponential simulation secure randomized encoding* (RE) *scheme in* CRS *model for Turing machines, there exists a secure offline witness encryption* (OWE) *scheme.*

We can also transform our OWE into an offline functional witness encryption (OFWE) [AFP16] where a decryption outputs a function of message and witness instead of only message. The encryption algorithm of our OFWE takes a function f as an additional input and encrypts the pair (x, m') using the PKE scheme to produce two ciphertexts c_1, c_2 under two public keys $\mathrm{PK}_1, \mathrm{PK}_2$ where $m' = (f, m)$. For decryption, we use the same circuit C except that we get (x', m') by decrypting c_1 using SK_1 and then output $f(m, w)$ if $x = x'$ and w is a valid witness for $x \in L$. The security demands that an adversary should not distinguish between the encryptions of $(x, (f_0, m_0))$ and $(x, (f_1, m_1))$ if $f_0(m_0, w) = f_1(m_1, w)$ for all valid witness w for $x \in L$. Our OFWE is secure under the same assumptions described in Theorem 2.

We further convert our single relation witness PRF scheme into a *multi-relation witness PRF*. In multi-relation witness PRF (mwPRF), a size bound of the relation circuits supported by the scheme is pre-specified depending on which we generate a secret functional key fk that is used for every relation. We use a randomly chosen pPRF secret key for fk as in our single relation scheme. The generation of evaluation key ek_R corresponding to a relation R is similar to that in our single relation witness PRF except the fact that the circuit E is hardcoded with the relation R in the key generation phase. We achieve the following result.

Theorem 3. (Informal) *Assuming* LWE *with sub-exponential hardness and the existence of sub-exponentially secure one-way functions, if there exists a weakly sub-linear compact public key functional encryption* (PKFE) *scheme with sub-exponential security, then there exists a secure multi-relation witness* PRF *scheme.*

Our mwPRF inherits the same advantages as of our single relation witness PRF over the scheme of [Zha16]. Additionally, the generation of evaluation key of [Zha16] needs to compute a multilinear map with the multilinearity level equal to the size of the description of the corresponding relation R which makes evaluation key ek_R computationally more expensive than ours.

Related works. The concept of witness encryption was introduced by Garg et al. [GGSW13]. They gave two candidate constructions of witness encryption for NP-complete *exact-cover problem.* One is based on a multilinear group family and other uses a graded encoding system. The security of their schemes depends on the *decision multilinear no-exact-cover assumption* and the *decision graded encoding no-exact-cover assumption.*

In [GKP+13b], Goldwasser et al. have shown how to obtain extractable witness encryption scheme using the construction of [GGSW13]. They developed attribute-based encryption (ABE) schemes for any polynomial time Turing machines and Random Access Machines (RAM) employing an extractable witness encryption, a succinct argument of knowledge and an existentially unforgeable signature as the ingredients. Additionally, they have constructed a (single-key and succinct) functional encryption scheme coupling their ABE-scheme for Turing machines with a fully homomorphic encryption scheme.

Boyel, Chung and Pass [BCP14] initiated the study of extractability obfuscation. They constructed an extractability obfuscator for all non-uniform polynomial-time Turing machines. A new notion of functional witness encryption was introduced where decryption gives a function of a message when a valid witness for the instance is known to the recipient of the ciphertext. In this work, it is shown that functional witness encryption is, in fact, equivalent to extractability obfuscation.

Witness encryption with soundness security was introduced by Garg et al. [GGSW13]. Bellare and Hoang [BH15] found that the soundness security of witness encryption scheme does not suffice for the security of the applications in [GGSW13]. The gap can be filled by the new security notion called adaptive soundness security. In this security model, given a security parameter λ, an adversary produces an instance x, a challenge pair of messages (m_0, m_1) and a state St. After receiving a ciphertext of message m_b from the challenger where b is chosen randomly from $\{0,\ 1\}$, the adversary has to guess for b provided the given instance does not belong to the corresponding language. In this work [BH15], Bellare and Hoang established a way to achieve the adaptive soundness security of witness encryption from indistinguishability obfuscator.

In [GGHW17], Garg et al. came up with implausibility results on extractable witness encryption with auxiliary input and general-purpose $di\mathcal{O}$ (differing-inputs obfuscation) by assuming the existence of a *special-purpose obfuscation.* In particular, a specific circuit C^* with special auxiliary input aux^* cannot be obfuscated in a way that hides some specific information. However, the existence of such a special-purpose obfuscation is a falsifiable assumption which they did not able to show how to break for candidate obfuscation schemes.

Aritra and Hnada [AH14] constructed a witness encryption scheme based on multilinear maps. They took the problem of existence of Hamilton cycle in a huge graph. The security proof is based on a generic colored matrix model as defined in the work of candidate indistinguishability obfuscation [GGH+16].

One of the main limitations of [GGSW13] is that the candidate had no proof of security (other than essentially assuming the scheme is secure). In [GLW14], Gentry, Lewko and Waters introduced *positional witness encryption* which provides a proof reduction of a witness encryption scheme via a sequence of 2^n hybrid experiments where n is witness length of the NP-statement.

Liu, Kakvi and Warinschi [LKW15] constructed a witness encryption scheme based on multilinear maps where they achieved extractable security without obfuscation. In particular, they presented the scheme for a special subset-sum problem. To encrypt any instance of an NP language, they needed to reduce it from conjunctive normal form-satisfiability problem (CNF-SAT) to the special subset-sum problem.

Derler and Slamanig [DS] uses SPHFs to build a practical witness encryption scheme for algebraic languages defined over bilinear groups. Their witness encryption scheme is compatible with the statements used in Groth-Sahai proofs.

Organization. The rest of the paper is arranged as follows. In section 2, we provide definitions of some cryptographic tools that are related to our work. Next section 3 contains our construction of witness PRFs and the security requirements. We present our offline witness encryption scheme in section 4. In section 5, we show that a multi-relation witness PRF can be instantiated from the witness PRF developed in section 3. Finally, we conclude in section 6.

2 Preliminaries

We use the notations in Table 1 throughout this paper.

$a \leftarrow A$	a is an output of the procedure A.
$a \xleftarrow{\$} X$	a is chosen uniformly at random from set X.
negligible function	$\mu : \mathbb{N} \to \mathbb{R}$ is a negligible function if $\mu(n) \leq \frac{1}{p(n)}$ holds for every polynomial $p(\cdot)$ and all sufficiently large $n \in \mathbb{N}$.
$(\lambda_0, S(\cdot))$-indistinguishability	Two ensembles $\{X_\lambda\}$ and $\{Y_\lambda\}$ are $(\lambda_0, S(\cdot))$-indistinguishable means $\lvert\Pr[x \xleftarrow{\$} X_\lambda : \mathcal{D}(x) = 1] - \Pr[y \xleftarrow{\$} Y_\lambda : \mathcal{D}(y) = 1]\rvert \leq \frac{1}{S(\lambda)}$ for any security parameter $\lambda > \lambda_0$ and every $S(\lambda)$-size distinguisher $\mathcal{D}$, $S : \mathbb{N} \to \mathbb{N}$.
δ-sub-exponential indistinguishability	Two ensembles $\{X_\lambda\}$ and $\{Y_\lambda\}$ are δ-sub-exponential indistinguishable means $\lvert\Pr[x \xleftarrow{\$} X_\lambda : \mathcal{D}(x) = 1] - \Pr[y \xleftarrow{\$} Y_\lambda : \mathcal{D}(y) = 1]\rvert < \delta(\lambda)^{\Omega(1)}$, for any security parameter λ and every poly-size distinguisher $\mathcal{D}$, where $\delta(\lambda) < 2^{\lambda^\epsilon}$, $0 < \epsilon < 1$.
$\text{Expt}(1^\lambda, 0) \approx_\delta \text{Expt}(1^\lambda, 1)$	For any polynomial size distinguisher $\mathcal{D}$, the advantage $\Delta = \lvert\Pr[\mathcal{D}(\text{Expt}(1^\lambda, 0)) = 1] - \Pr[\mathcal{D}(\text{Expt}(1^\lambda, 1)) = 1]\rvert$ is bounded by δ.

Table 1: Notations

2.1 Pseudorandom Function

A finite set of functions $\{\mathsf{F}_s : \mathcal{X} \to \mathcal{Y}\}_s$ with a seed or key s is said to form a pseudorandom function family [GGM86] if F_s can be efficiently computed for given s and is computationally indistinguishable from a random function $\mathsf{R} : \mathcal{X} \to \mathcal{Y}$ given oracle access to R.

Definition 1. (Pseudorandom function). A *pseudorandom function*(PRF) is a function $\mathsf{F} : \{0,1\}^\lambda \times \mathcal{X} \to \mathcal{Y}$ with polynomial runtime satisfying

$$\lvert\Pr[\mathcal{A}^{\mathsf{F}(K,\,\cdot)}(1^\lambda) = 1 : K \xleftarrow{\$} \{0,1\}^\lambda] - \Pr[\mathcal{A}^{\mathsf{R}(\cdot)}(1^\lambda) = 1 : \mathsf{R} \xleftarrow{\$} \mathcal{U}]\rvert \leq \mu(\lambda)$$

for every probabilistic polynomial time (PPT) adversary $\mathcal{A}$, where $\mathcal{U}$ is the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$ and μ is a negligible function in λ. The pseudorandom function F is said to be δ-secure for some specific negligible function $\delta(\cdot)$ if the indistinguishability gap $\mu(\lambda)$ is less than $\delta(\lambda)^{\Omega(1)}$.

2.2 Puncturable Pseudorandom Function

Sahai and Waters [SW14] introduced a key-puncturing technique for pseudorandom functions that can be used to build many cryptographic primitives with the help of obfuscation. The punctured key of a puncturable pseudorandom function allows to evaluate PRF at all points except for the points in a (predefined) polynomial-size set.

Definition 2. (Puncturable pseudorandom function). A *puncturable pseudorandom function* (pPRF) consists of a tuple of algorithms pPRF = (Gen, Eval, Punc) over the domain $\mathcal{X}$ and range $\mathcal{Y}$ and is defined as follows:

- $K \leftarrow$pPRF.Gen(1^λ): It is a randomized algorithm run by a trusted authority which takes as input a security parameter λ and outputs a secret key $K \in \{0,1\}^\lambda$.
- $y \leftarrow$pPRF.Eval(K', x): It is a deterministic algorithm which on input a key K' and an element $x \in \mathcal{X}$, outputs the PRF value $y \in \mathcal{Y}$.
- $K\{\mathcal{S}\} \leftarrow$pPRF.Punc$(K, \mathcal{S})$: It is a deterministic algorithm that takes a secret key K and a polynomial-size set $\mathcal{S} \subset \mathcal{X}$ as input and outputs a punctured key $K\{\mathcal{S}\}$. If S contains a single element, say x, then we simply write $K\{\mathcal{S}\}$ as $K\{x\}$.

Correctness: (Functionality preserving under puncturing) For all polynomial-size subset $\mathcal{S}$ of $\mathcal{X}$, and for all $x \in \mathcal{X} \setminus \mathcal{S}$ we have that

$$\Pr[\text{pPRF.Eval}(K, x) = \text{pPRF.Eval}(K\{\mathcal{S}\}, x)] = 1.$$

Definition 3. (Pseudorandomness at punctured points). We say that a puncturable pseudorandom function pPRF = (Gen, Eval, Punc) preserves pseudorandomness at punctured points if

$$|\Pr[\mathcal{A}(K\{\mathcal{S}\}, \{\text{pPRF.Eval}(K, x)\}_{x \in \mathcal{S}}) = 1] - \Pr[\mathcal{A}(K\{\mathcal{S}\}, U^{|\mathcal{S}|}) = 1]| \leq \mu(\lambda)$$

for every PPT adversary $\mathcal{A}$ and any polynomial-size subset $\mathcal{S}$ of $\mathcal{X}$, where $K \leftarrow$ pPRF.Gen(1^λ), $K\{\mathcal{S}\} \leftarrow$ pPRF.Punc$(K, \mathcal{S})$, U denotes the uniform distribution over $\mathcal{Y}$ and μ is a negligible function in λ. The pPRF is said to be δ-secure for some specific negligible function $\delta(\cdot)$ if the above indistinguishability gap $\mu(\lambda)$ is less than $\delta(\lambda)^{\Omega(1)}$.

It has been observed by [BW13, BGI14] that puncturable PRFs can be constructed from one-way functions using the GGM tree-based construction of PRFs [GGM86] where the size of the punctured key grows polynomially with the number of elements in the set $\mathcal{S}$. The GGM construction [GGM86] of pseudorandom functions uses a cryptographically strong bit (CSB) generator or pseudorandom generator (PRG) which can be obtained from one-way functions. Moreover, if the one-way functions are assumed to be sub-exponentially hard then the PRG is also sub-exponentially secure. We describe these facts in the following theorems:

Theorem 4. [GGM86, Lev87] *Assuming the existence of sub-exponentially secure one-way functions, there exists an efficiently computable sub-exponentially secure pseudorandom generator for any desired poly-size input length.*

Theorem 5. [GGM86, BW13, BGI14] *Assuming the existence of one-way functions, there exists an efficiently computable puncturable pseudorandom function for any desired poly-size input length.*

1. The challenger runs PKE.Gen(1^λ) $\rightarrow$(SK, PK) and makes PK public.
2. The adversary $\mathcal{A}$ selects $m_0, m_1 \in \mathcal{M}$ such that $|m_0| = |m_1|$ and sends (m_0, m_1, st) to the challenger where st contains some auxiliary information.
3. Next, the challenger chooses a random bit $b \in \{0,1\}$, a randomness r and sends $c_b \leftarrow$PKE.Enc(PK, $m_b; r$) to $\mathcal{A}$.
4. The adversary $\mathcal{A}$ observes c_b and st and outputs a guess b' for b.

Figure 1: $\mathbf{Expt}_{\mathcal{A}}^{\mathrm{PKE}}(1^\lambda, b)$: The security game of a CPA-secure public-key encryption

2.3 Public-Key Encryption

Definition 4. (Public-key encryption). A *public-key encryption* scheme for a message space $\mathcal{M}$ is a tuple of PPT algorithms PKE = (Gen, Enc, Dec) with the following properties:

- (SK, PK) $\leftarrow$ PKE.Gen(1^λ): This is a randomized key generation algorithm which is run by a trusted third party with a security parameter λ as input. It outputs a public key PK and a secret key SK. A user who obtains a key pair (PK, SK) from a trusted party, keeps the secret key SK and publishes the public key PK.
- $c \leftarrow$ PKE.Enc(PK, m; r): The encrypter uses the public key PK to encrypt a message $m \in \mathcal{M}$ using a randomness r and produces a ciphertext c which is broadcasted over a public domain.
- PKE.Dec(SK, c) $\in \mathcal{M} \cup \{\perp\}$: The recipient of a ciphertext c runs this algorithm using the secret key SK and gets either a message $m \in \mathcal{M}$ or $\perp$ where $\perp$ indicates a failure of the algorithm.

Correctness: For every $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$, we have

$$\Pr[\text{PKE.Dec}(\text{SK}, c) = m : (\text{SK, PK}) \leftarrow \text{PKE.Gen}(1^\lambda),\ c \leftarrow \text{PKE.Enc}(\text{PK},\ m; r)] = 1$$

Definition 5. (Indistinguishability under chosen-plaintext attacks). We say that a public-key encryption scheme PKE = (Gen, Enc, Dec) is indistinguishable under chosen plaintext attacks (CPA) if

$$|\Pr[\text{Expt}_{\mathcal{A}}^{\text{PKE}}(1^\lambda, 0) = 1] - \Pr[\text{Expt}_{\mathcal{A}}^{\text{PKE}}(1^\lambda, 1) = 1]| \leq \mu(\lambda)$$

for any $\lambda \in \mathbb{N}$ and every PPT adversary $\mathcal{A}$ in the experiments $\text{Expt}_{\mathcal{A}}^{\text{PKE}}(1^\lambda, b)$ defined in Figure 1 where $b \in \{0,1\}$ and μ is a negligible function of λ. The PKE is said to be δ-selectively secure for some specific negligible function $\delta(\cdot)$ if the above indistinguishability gap $\mu(\lambda)$ is less than $\delta(\lambda)^{\Omega(1)}$.

2.4 Witness PRF

Informally, a witness PRF scheme [Zha16] produces a somewhat random value from a set with respect to an instance $x \in L$ for an NP language L and a user can recompute the value provided he has a witness w for $x \in L$.

Definition 6. (Witness PRF). A *witness PRF* (wPRF) for an NP language L with the witness relation $R : \chi \times \mathcal{W} \rightarrow \{0,1\}$ consists of three algorithms wPRF = (Gen, F, Eval) and works as follows:

1. The adversary $\mathcal{A}$ chooses a single challenge query on an instance $x^* \in \mathcal{X} \setminus L$ to the challenger.
2. The challenger generates $(\mathsf{fk}, \mathsf{ek}) \leftarrow w\text{PRF.Gen}(1^\lambda, R)$ and gives ek to the adversary $\mathcal{A}$.
3. The challenger computes $y_0 \leftarrow w\text{PRF.F}(\mathsf{fk}, x^*)$, selects $y_1 \xleftarrow{\$} \mathcal{Y}$ and sends y_b to $\mathcal{A}$ for a randomly chosen $b \in \{0,1\}$.
4. The adversary $\mathcal{A}$ makes polynomially many queries on instance $x_i \in \mathcal{X}$, $i = 1, 2, \ldots, u$, to which the challenger responses with $w\text{PRF.F}(\mathsf{fk}, x_i)$ if $x_i \neq x^*$; otherwise terminates the game.
5. Then $\mathcal{A}$ outputs a guess b' for b.

Figure 2: $\mathbf{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, b)$: The security game of a selectively-secure witness PRF

- $(\mathsf{fk}, \mathsf{ek}) \leftarrow w\text{PRF.Gen}(1^\lambda, R)$: A trusted authority [1] takes as input the security parameter λ and a relation circuit $R : \chi \times \mathcal{W} \rightarrow \{0,1\}$ and randomly generates a secret function key fk and a public evaluation key ek. A user receiving $(\mathsf{fk}, \mathsf{ek})$ through a secure channel, keeps fk as a secret key and publishes ek. We note that $R(x, w) = 1$ if and only if w is a valid witness for $x \in L$.
- $y \leftarrow w\text{PRF.F}(\mathsf{fk}, x)$: Using a function key fk and an input $x \in \chi$, the user runs this algorithm which deterministically outputs some $y \in \mathcal{Y}$.
- $w\text{PRF.Eval}(\mathsf{ek}, x, w) \in \mathcal{Y} \cup \{\perp\}$: An witness holder runs this algorithm using an evaluation key ek, an input $x \in \chi$ and a witness $w \in \mathcal{W}$ and deterministically recovers either $y \in \mathcal{Y}$ or $\perp$.

Correctness: For all $x \in \mathcal{X}$, $w \in \mathcal{W}$, we have that

$$w\text{PRF.Eval}(\mathsf{ek}, x, w) = \begin{cases} w\text{PRF.F}(\mathsf{fk}, x) & \text{if } R(x, w) = 1 \\ \perp & \text{if } R(x, w) = 0 \end{cases} \quad (1)$$

Definition 7. (Selectively secure witness PRF). We say that a witness PRF scheme $w\text{PRF} =$ (Gen, F, Eval) for an NP language L, a relation $R : \chi \times \mathcal{W} \rightarrow \{0,1\}$, a set $\mathcal{Y}$, is selectively secure if

$$\left|\Pr\left[\text{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, 0) = 1\right] - \Pr\left[\text{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, 1) = 1\right]\right| \leq \mu(\lambda)$$

for any $\lambda \in \mathbb{N}$ and every PPT adversary $\mathcal{A}$ in the experiments $\text{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, b)$ defined in Figure 2 where $b \in \{0,1\}$ and μ is a negligible function of λ. The wPRF is said to be δ-selectively secure for some specific negligible function $\delta(\cdot)$ if the above indistinguishability gap $\mu(\lambda)$ is smaller than $\delta(\lambda)^{\Omega(1)}$.

Definition 8. (Extractable witness PRFs). A witness PRF scheme $w\text{PRF} =$ (Gen, F, Eval) for an NP language L with relation R is said to be a secure *extractable witness PRF* with respect to an R-instance sampler $\mathcal{D}$ if there exists a polynomial $p(\cdot)$ such that

$$\left|\Pr\begin{bmatrix} \mathcal{A}^{w\text{PRF.F}(\mathsf{fk},\cdot)}(\mathsf{ek}, x^*, \mathsf{Aux}, y^*) = 1 : (\mathsf{fk}, \mathsf{ek}) \leftarrow w\text{PRF.Gen}(\lambda, R), \\ (x^*, \mathsf{Aux}) \xleftarrow{\$} \mathcal{D}^{w\text{PRF.F}(\mathsf{fk},\cdot)}(\mathsf{ek}), y^* \leftarrow w\text{PRF.F}(\mathsf{fk}, x^*) \end{bmatrix} - \right.$$

[1] We note that a user may itself run this algorithm to get the secret function key fk and the evaluation key ek which is made public.

$$\Pr\left[\begin{array}{c}\mathcal{A}^{w\text{PRF.F}(\mathsf{fk},\cdot)}(\mathsf{ek}, x^*, \mathsf{Aux}, y^*) = 1 : (\mathsf{fk}, \mathsf{ek}) \leftarrow w\text{PRF.Gen}(\lambda, R), \\ (x^*, \mathsf{Aux}) \xleftarrow{\$} \mathcal{D}^{w\text{PRF.F}(\mathsf{fk},\cdot)}(\mathsf{ek}), y^* \xleftarrow{\$} \mathcal{Y}\end{array}\right] \geq \frac{1}{2} + \frac{1}{p(\lambda)} \quad (2)$$

for every PPT adversary $\mathcal{A}$ and infinitely many λ, then there exists a PPT extractor $\mathcal{E}$ and a polynomial $q(\cdot)$ such that

$$\Pr\left[\begin{array}{c}w^* \xleftarrow{\$} \mathcal{E}(\mathsf{ek}, x^*, \mathsf{Aux}, y^*, \{x_i\}, r) : R(x^*, w^*) = 1, \{x_i\} \text{ are the } w\text{PRF.F quires} \\ \text{of } \mathcal{A} \text{ and } r \text{ is the random coin of } \mathcal{A}\end{array}\right] \geq \frac{1}{q(\lambda)} \quad (3)$$

for infinitely many λ.

2.5 Multi-Relation Witness PRF

The notion of multi-relation witness PRFs was introduced by Zhandry [Zha16] to work with multiple relations but with the same secret function key.

Definition 9. (Multi-Relation Witness PRFs). A multi-relation witness PRF scheme for for a set of relations $\mathcal{R} = \{R : |R| \leq s, R : \chi \times \mathcal{W} \to \{0, 1\}\}$ consists of three algorithms mwPRF=(Gen, F, Eval) and works as follows:

- $\mathsf{fk} \leftarrow mw\text{PRF.Gen}(\lambda, s)$: It is a randomized algorithm run by a user[2] which takes as input a security parameter λ and a bound s on the size of supported relations and produces a secret function key fk.
- $y \leftarrow mw\text{PRF.F}(\mathsf{fk}, x)$: It is a deterministic algorithm that takes as input a secret function key fk and an instance $x \in \mathcal{X}$, and outputs an element $y \in \mathcal{Y}$ for some set $\mathcal{Y}$.
- $\mathsf{ek}_R \leftarrow mw\text{PRF.KeyGen}(\mathsf{fk}, R)$: It is possibly a randomized algorithm run by a user having fk, which needs as input a secret function key fk and a relation circuit R, and produces a public evaluation key ek_R corresponding to the relation R.
- $mw\text{PRF.Eval}(\mathsf{ek}_R, x, w) \in \mathcal{Y} \cup \{\perp\}$: It is a deterministic algorithm run by a witness holder that on input an evaluation key ek_R, an instance x and a witness w, outputs an element $y \in \mathcal{Y}$ or $\perp$.

Correctness: For all $x \in \mathcal{X}$, $w \in \mathcal{W}$, we have that

$$mw\text{PRF.Eval}(\mathsf{ek}_R, x, w) = \begin{cases} mw\text{PRF.F}(\mathsf{fk}, x) & \text{if } R(x, w) = 1 \\ \perp & \text{if } R(x, w) = 0 \end{cases}$$

Definition 10. (Selectively secure multi-relation witness PRF). We say that a multi-relation witness PRF scheme mwPRF=(Gen, F, Eval) for a set of relations $\mathcal{R} = \{R : |R| \leq s, R : \chi \times \mathcal{W} \to \{0, 1\}\}$, a set $\mathcal{Y}$, is selectively secure if

$$\left|\Pr\left[\text{Expt}_{\mathcal{A}}^{mw\text{PRF}}(1^\lambda, 0) = 1\right] - \Pr\left[\text{Expt}_{\mathcal{A}}^{mw\text{PRF}}(1^\lambda, 1) = 1\right]\right| \leq \mu(\lambda)$$

for any $\lambda \in \mathbb{N}$ and every PPT adversary $\mathcal{A}$ in the experiments $\text{Expt}_{\mathcal{A}}^{mw\text{PRF}}(1^\lambda, b)$ defined in Figure 3 where $b \in \{0, 1\}$ and μ is a negligible function of λ. The mwPRF is said to be δ-selectively secure for some specific negligible function $\delta(\cdot)$ if the above indistinguishability gap $\mu(\lambda)$ is smaller than $\delta(\lambda)^{\Omega(1)}$.

We can similarly define extractable multi-relation witness PRFs as described in Definition 8 for single relation witness PRFs.

[2] We note that this algorithm can be processed by a trusted third party to generate the secret function key fk and in that case the key is sent to a user through a secure channel.

1. The adversary $\mathcal{A}$ chooses a single challenge query on an instance $x^* \in \mathcal{X}$ to the challenger.

2. The challenger generates $\mathsf{fk} \xleftarrow{\$} mw\text{PRF.Gen}(1^\lambda,\ s)$, and delivers s to the adversary $\mathcal{A}$.

3. The challenger computes $y_0 \leftarrow mw\text{PRF.F}(\mathsf{fk},\ x^*)$ and $y_1 \xleftarrow{\$} \mathcal{Y}$ and sends y_b to $\mathcal{A}$ for a randomly chosen $b \in \{0, 1\}$.

4. The adversary $\mathcal{A}$ makes polynomially many evaluation key quires $R_i \in \mathcal{R}$ for $i = 1, 2, \cdots l$, to which the challenger responses with $\mathsf{ek}_{R_i} \leftarrow mw\text{PRF.KeyGen}(\mathsf{fk}, R_i)$ if x^* has no witness corresponding to the relation R_i; otherwise stops the game.

5. Next, $\mathcal{A}$ makes polynomially many queries on instance $\{x_{i_j}\}_{j=1}^{u(i)} \in \mathcal{X}$, for $i = 1, 2, \cdots, l$, $u(i)$ is a polynomial in λ. The challenger responses with $mw\text{PRF.F}(\mathsf{fk},\ x_{i_j})$ if $x_{i_j} \neq x^*$ and x_{i_j} is an instance of interest corresponding to the relation R_i; otherwise the challenger terminates the game.

6. Finally, $\mathcal{A}$ outputs a guess b' for b.

Figure 3: $\mathbf{Expt}_{\mathcal{A}}^{mw\text{PRF}}(1^\lambda, b)$: The security game of a selectively-secure multi-relation witness PRF scheme

2.6 Witness Encryption

Witness encryption was introduced by Garg et al. [GGSW13] to encrypt a message with an NP statement and decryption is successful with a valid witness to the statement.

Definition 11. (Witness encryption). A *witness encryption* (WE) scheme for an NP language L with the witness relation $R : \chi \times \mathcal{W} \to \{0, 1\}$ consists of two algorithms WE = (Enc, Dec) satisfying the following:

- $c \leftarrow$WE.Enc$(1^\lambda, x, m)$: An encrypter takes as input a security parameter λ, an instance $x \in \chi$ and a message $m \in \mathcal{M}$ and outputs a ciphertext c.
- WE.Dec$(c, w) \in \mathcal{M} \cup \{\perp\}$: A witness holder takes a ciphertext c and a witness $w \in \mathcal{W}$ as input and outputs a message $m \in \mathcal{M}$ or $\perp$.

Correctness: For any $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$, $(x, w) \in \chi \times \mathcal{W}$ such that $x \in L$, $R(x, w) = 1$, we have that

$$\Pr\big[\text{WE.Dec}(c, w) = m : c \leftarrow \text{WE.Enc}(1^\lambda, x, m)\big] = 1.$$

Definition 12. (Soundness security of witness encryption). A tuple of algorithms WE = (Enc, Dec) is soundness secure witness encryption scheme for an NP language L and a relation $R : \chi \times \mathcal{W} \to \{0, 1\}$, if

$$|\Pr[\ \mathcal{A}(\text{WE.Enc}(1^\lambda, x, m_0))\text{=}1] - \Pr[\ \mathcal{A}(\text{WE.Enc}(1^\lambda, x, m_1))\text{=}1]| \leq \mu(\lambda)$$

for any $x \notin L$, for any PPT adversary $\mathcal{A}$ and messages $m_0, m_1 \in \mathcal{M}$ with $|m_0| = |m_1|$ where μ is a negligible function of λ. The WE scheme is said to be δ-soundness secure for some specific negligible function $\delta(\cdot)$ if the above indistinguishability gap $\mu(\lambda)$ is less than $\delta(\lambda)^{\Omega(1)}$.

2.7 Offline Witness Encryption

Witness encryption scheme with an offline phase [AFP16] reduces time of encryption by shifting the heavy-computing part into a setup algorithm. We note that setup is independent of the statement and message to be encrypted.

1. The adversary $\mathcal{A}$ chooses $x \in \mathcal{X} \setminus L$, $m_0, m_1 \in \mathcal{M}$ such that $|m_0| = |m_1|$ and sends (x, m_0, m_1, st) to the challenger where st is a state containing some auxiliary information.
2. The challenger generates $(pp_e, pp_d) \leftarrow$ OWE.Setup$(1^\lambda, R)$ and sends this to $\mathcal{A}$.
3. The challenger selects $b \in \{0, 1\}$ and sends $c_b \leftarrow$ OWE.Enc$(1^\lambda, x, m_b, pp_e)$ to $\mathcal{A}$.
4. The adversary $\mathcal{A}$ outputs a bit b' for b by observing (st, c_b, pp_e, pp_d).

Figure 4: $\mathbf{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, b)$: The security game of a selectively-secure offline witness encryption

Definition 13. (Offline witness encryption). An *offline witness encryption* (OWE) scheme for an NP language L with witness relation $R : \chi \times \mathcal{W} \to \{0, 1\}$ is a tuple of algorithms OWE = (Setup, Enc, Dec) with the following requirements:

- $(pp_e, pp_d) \leftarrow$OWE.Setup$(1^\lambda, R)$: This algorithm is run by a trusted third party which takes as input a security parameter λ and publishes a public parameter pp_e for encryption and a public parameter pp_d for decryption.
- $c \leftarrow$OWE.Enc$(1^\lambda, x, m, pp_e)$: The encryption algorithm takes as input the security parameter λ, an instance $x \in \chi$, a message $m \in \mathcal{M}$ and encryption parameter pp_e. It computes a ciphertext c and broadcasts it over a public channel.
- OWE.Dec$(c, w, pp_d) \in \mathcal{M} \cup \{\perp\}$: A witness holder on receiving a ciphertext c runs this algorithm using a witness w and decryption parameter pp_d to recover either $m \in \mathcal{M}$ or $\perp$.

Correctness: For any $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$, $(x, w) \in \chi \times \mathcal{W}$ such that $x \in L$, $R(x, w) = 1$, we have that

$$\Pr\begin{bmatrix} \text{OWE.Dec}(c, w, pp_d) = m : (pp_e, pp_d) \leftarrow \text{OWE.Setup}(1^\lambda, R), \\ c \leftarrow \text{OWE.Enc}(1^\lambda, x, m, pp_e) \end{bmatrix} = 1. \tag{4}$$

Definition 14. (Selectively secure offline witness encryption). We say that an offline witness encryption OWE = (Setup, Enc, Dec) for an NP language L and a relation $R : \chi \times \mathcal{W} \to \{0, 1\}$, is selectively secure if

$$\left|\Pr\left[\mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, 0) = 1\right] - \Pr\left[\mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, 1) = 1\right]\right| \leq \mu(\lambda)$$

for any $\lambda \in \mathbb{N}$ and every PPT adversary $\mathcal{A}$ in the experiments $\mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, b)$ defined in Figure 4 where $b \in \{0, 1\}$ and μ is a negligible function of λ. The OWE is said to be δ-selectively secure for some specific negligible function $\delta(\cdot)$ if the above indistinguishability gap $\mu(\lambda)$ is less than $\delta(\lambda)^{\Omega(1)}$.

2.8 Offline Functional Witness Encryption

The notion of functional witness encryption scheme was given by Boyel et al. [BGI14] who established the equivalence between extractable obfuscation and functional witness encryption with extractable security. Abusalah et al. [AFP16] introduced functional witness encryption with a setup algorithm and named it as offline functional witness encryption.

Definition 15. (Offline functional witness encryption). An *offline functional witness encryption* (OFWE) scheme for an NP language L with witness relation $R : \chi \times \mathcal{W} \to \{0, 1\}$ and a class of functions $\{f_\lambda\}_{\lambda \in \mathbb{N}}$ is a tuple of algorithms OFWE = (Setup, Enc, Dec) with the following requirement:

1. The adversary $\mathcal{A}$ chooses $x \in \mathcal{X}$, $(f_0, m_0), (f_1, m_1) \in f_\lambda \times \mathcal{M}$ such that $f_0(m_0, w) = f_1(m_1, w)$ for all w satisfying $R(x, w) = 1$ and $|(f_0, m_0)| = |(f_1, m_1)|$ and sends (x, m_0, m_1, st) to the challenger where st is a state containing some auxiliary information.

2. The challenger generates $(pp_e, pp_d) \leftarrow$ OFWE.Setup$(1^\lambda, R)$ and sends this to $\mathcal{A}$.

3. The challenger selects $b \in \{0, 1\}$ and sends $c_b \leftarrow$ OFWE.Enc$(1^\lambda, x, (f_b, m_b), pp_e)$ to $\mathcal{A}$.

4. The adversary $\mathcal{A}$ guesses a bit b' for b by observing (st, c_b, pp_e, pp_d).

Figure 5: $\mathbf{Expt}_{\mathcal{A}}^{\text{OFWE}}(1^\lambda, b)$: The security game of a selectively-secure offline functional witness encryption

- $(pp_e, pp_d) \leftarrow$OFWE.Setup$(1^\lambda, R)$: A trusted third party runs this algorithm taking input a security parameter λ and publishes a public parameter pp_e for encryption and a public parameter pp_d for decryption.
- $c \leftarrow$OFWE.Enc$(1^\lambda, x, (f, m), pp_e)$: The encryption algorithm takes as input a security parameter λ, an instance $x \in \chi$, a function $f \in f_\lambda$, a message $m \in \mathcal{M}$ and encryption parameter pp_e. It outputs a ciphertext c over a public channel. The domain of the function class is $\mathcal{M} \times \mathcal{W}$.
- OFWE.Dec$(c, w, pp_d) \in \mathcal{M} \cup \{\perp\}$: A witness holder on receiving a ciphertext c runs this algorithm using a witness w and decryption parameter pp_d and recovers either $f(m, w)$ or $\perp$.

Correctness: For any $\lambda \in \mathbb{N}$, $f \in f_\lambda$, $m \in \mathcal{M}$, $(x, w) \in \chi \times \mathcal{W}$ such that $x \in L$, $R(x, w) = 1$, we have that

$$\Pr\begin{bmatrix}\text{OFWE.Dec}(c, w, pp_d) = f(m, w) : (pp_e, pp_d) \leftarrow \text{OFWE.Setup}(1^\lambda, R), \\ c \leftarrow \text{OFWE.Enc}(1^\lambda, x, (f, m), pp_e)\end{bmatrix} = 1.$$

Definition 16. (Selectively secure offline functional witness encryption). We say that an offline functional witness encryption OFWE = (Setup, Enc, Dec) for an NP language L with relation $R : \chi \times \mathcal{W} \to \{0, 1\}$ and a function class $\{f_\lambda\}_{\lambda \in \mathbb{N}}$, is selectively secure if

$$\left|\Pr\left[\text{Expt}_{\mathcal{A}}^{\text{OFWE}}(1^\lambda, 0) = 1\right] - \Pr\left[\text{Expt}_{\mathcal{A}}^{\text{OFWE}}(1^\lambda, 1) = 1\right]\right| \leq \mu(\lambda)$$

for any $\lambda \in \mathbb{N}$ and every PPT adversary $\mathcal{A}$ in the experiments $\text{Expt}_{\mathcal{A}}^{\text{OFWE}}(1^\lambda, b)$ defined in Figure 5 where $b \in \{0, 1\}$ and μ is a negligible function of λ. The OFWE is said to be δ-selectively secure for some specific negligible function $\delta(\cdot)$ if the above indistinguishability gap $\mu(\lambda)$ is less than $\delta(\lambda)^{\Omega(1)}$.

2.9 Randomized Encoding Scheme in CRS Model

Randomized encoding was introduced by Ishai and Kushilevitz [IK00] to encode a complex deterministic function Π along with an input x through an encoding algorithm whose output distribution $\widehat{\Pi}(x)$ can efficiently compute $\Pi(x)$ and reveals no information beyond $\Pi(x)$. Recently, Lin et al. [LPST16b] studied randomized encoding scheme in both plain model and common reference string (CRS) model with compactness and sub-linear compactness of the size of encodings.

Definition 17. (Randomized encoding schemes in CRS model). A randomized encoding scheme RE = (Setup, Enc, Eval) in CRS model for a class of Turing machines $\{\mathcal{M}_\lambda\}$ where Setup and Enc are randomized algorithms and Eval is a deterministic algorithm, performs as follows:

- (crs, pk)$\leftarrow$ RE.Setup($1^\lambda, 1^m, 1^n, 1^T, 1^l$): A trusted third party takes as input a security parameter λ, a machine size bound m, input length bound n, time bound T and output length l. It outputs a common reference string crs and a public key pk.
- $\widehat{\Pi}_x \leftarrow$RE.Enc($pk, \Pi, x$): The encoding algorithm uses a public key pk, a Turing machine $\Pi \in \mathcal{M}_\lambda$ together with an input x and outputs an encoding $\widehat{\Pi}_x$.
- $y \leftarrow$RE.Eval($\widehat{\Pi}_x$, crs): An evaluator makes use of an encoding $\widehat{\Pi}_x$ and a common reference string crs and outputs some y.

Correctness: For any $\lambda \in \mathbb{N}$, $m(\lambda), n(\lambda), T(\lambda), l(\lambda) \in \mathbb{N}$, Turing machine $\Pi \in \mathcal{M}_\lambda$ and input x with $|\Pi| \leq m$, $|x| \leq n$ and $|\Pi^T(x)| \leq l$, we have that

$$\Pr\left[\begin{matrix}\text{RE.Eval}(\widehat{\Pi}_x, \text{crs}) = \Pi^T(x) : (\text{crs}, pk) \leftarrow \text{RE.Setup}(1^\lambda, 1^m, 1^n, 1^T, 1^l) \\ \widehat{\Pi}_x \leftarrow \text{RE.Enc}(pk, \Pi, x)\end{matrix}\right] = 1$$

Here $\Pi^T(x)$ denotes the output of the Turing machine Π on input x when run in at most T steps.

Definition 18. (($\lambda_0, S(\cdot)$)-simulation security of randomized encoding in CRS model). We say that a randomized encoding scheme RE for a class of Turing machines $\{\mathcal{M}_\lambda\}$ in CRS model is $(\lambda_0, S(\cdot))$-simulation secure if there exists a PPT algorithm Sim and a constant c such that for every $\{\Pi, x, m, n, l, T\}$ where $\Pi \in \mathcal{M}_\lambda$ and $|\Pi|, |x|, m, n, l, T \leq B(\lambda)$ for some polynomial B, the ensembles

$$\left\{(\text{crs}, pk, \widehat{\Pi}_x) : (\text{crs}, pk) \leftarrow \text{RE.Setup}(1^\lambda, 1^m, 1^n, 1^T, 1^l), \widehat{\Pi}_x \leftarrow \text{RE.Enc}(pk, \Pi, x)\right\} \text{ and}$$

$$\left\{(\text{crs}, pk, \widehat{\Pi}_x) : (\text{crs}, pk, \widehat{\Pi}_x) \leftarrow \mathsf{Sim}(1^\lambda, \Pi^T(x), 1^{|\Pi|}, 1^{|x|}, 1^m, 1^n, 1^T, 1^l)\right\}$$

are $(\lambda_0, S'(\lambda))$-indistinguishable (see Table 1), with $S'(\lambda) = S(\lambda) - B(\lambda)^c$ for all $\lambda \in \mathbb{N}$. The RE is said to be δ-simulation secure for some specific negligible function $\delta(\cdot)$ if $S'(\lambda)$ is greater than $\delta(\lambda)^{\Omega(1)}$. Also, we say that RE is δ-sub-exponential simulation secure if $\delta(\lambda) < 2^{\lambda^\epsilon}$, $0 < \epsilon < 1$.

Definition 19. (Compactness randomized encoding for Turing machines). A $(\lambda_0, S(\cdot))$-simulation secure randomized encoding scheme is said to be compact if

$$\mathsf{Time}_{\text{RE.Enc}}(1^\lambda, \Pi, x, T) = \text{poly}(\lambda, |\Pi|, |x|, \log T) \text{ and}$$

$$\mathsf{Time}_{\text{RE.Eval}}(\widehat{\Pi}_x, \text{crs}) = \text{poly}(\lambda, |\Pi|, |x|, T)$$

for every security parameter λ, Turing machine Π, input x, time-bound T and every encoding $\widehat{\Pi}_x \leftarrow$ RE.Enc(pk, Π, x) where (crs, pk) $\leftarrow$ RE.Setup($1^\lambda, \cdot$). Here $\mathsf{Time}_X(\cdot)$ denotes the time-bound of the algorithm X with a specified class of inputs.

Definition 20. (Succinct randomized encoding for Turing machines). A $(\lambda_0, S(\cdot))$-simulation secure randomized encoding scheme is said to be succinct for a class of Turing machines $\{M_\lambda\}$ if the efficiency requirement for RE.Enc is defined as

$$\mathsf{Time}_{\text{RE.Enc}}(1^\lambda, \Pi, x, T) = l \cdot \text{poly}(\lambda, |\Pi|, |x|, \log T).$$

The notations are the same as in Definition 19

Definition 21. (Sub-linear compactness of randomized encoding for Turing machines). A $(\lambda_0, S(\cdot))$-simulation secure randomized encoding scheme is said to be sub-linearly compact for a class of Turing machines $\{M_\lambda\}$ if the efficiency requirement for RE.Enc is defined as

$$\mathsf{Time}_{\text{RE.Enc}}(1^\lambda, \Pi, x, T) \le \text{poly}(\lambda, |\Pi|, |x|) \cdot T^{1-\epsilon}$$

for some $\epsilon \in (0,1)$. The notations are the same as in Definition 19

Randomized encoding schemes in CRS model can be constructed from a public key functional encryption (PKFE) scheme and a pseudorandom generator [LPST16b]. The compactness (respectively, sub-linear compactness) of RE in CRS model depends on the compactness (respectively, sub-linear compactness) of the underlying PKFE. In [BNPW16], a weakly sub-linear compact PKFE for $\mathsf{P/poly}$ (i.e. for polynomial size circuits) is constructed using plain public key encryption and strong exponentially-efficient indistinguishability obfuscation (SXIO). They also instantiated SXIO from sub-exponentially secure secret key functional encryption (SKFE) schemes. The existence of a sub-linearly compact randomized encoding scheme for Turing machines follows from the two theorems stated below.

Theorem 6. [BNPW16] *Assuming a plain public-key encryption* (PKE) *and strong exponentially-efficient indistinguishability obfuscation* (SXIO) *with a small enough constant compressing factor, there exists a weakly sub-linear compact public key functional encryption* (PKFE) *scheme (for functions with long output).*

Theorem 7. [LPST16b] *Assuming hardness of Learning With Error (respectively, sub-exponential hardness), if there exists a selectively secure weakly sub-linear compact public key functional encryption* (PKFE) *scheme for* $\mathsf{P/poly}$ *(respectively, with sub-exponential hardness), then there exists a sub-linearly compact randomized encoding* (RE) *scheme for Turing machines in* CRS *model with (respectively, sub-exponential) simulation security.*

Remark 1. The existence of sub-linearly compact RE scheme for Turing machines is almost directly followed from a succinct RE scheme and a weakly sub-linear compact RE scheme for Turing machines [LPST16b]. The succinctness of a RE scheme for Turing machines depends on the succinctness of PKFE for $\mathsf{P/poly}$. Using only the sub-exponential hardness of LWE, there exists a succinct PKFE with sub-exponential security for NC^1 [GKP$^+$13a]. Also, there exist transformations [GKP$^+$13a, ABSV14] from symmetric key encryption with decryption circuit in NC^1 together with succinct PKFE for NC^1 to a succinct PKFE for $\mathsf{P/poly}$. We note that if the symmetric key encryption and the succinct PKFE for NC^1 are both sub-exponentially secure then the resulting succinct PKFE for $\mathsf{P/poly}$ is also sub-exponentially secure.

A sub-exponentially secure weakly sub-linear compact PKFE is achieved in [BNPW16] using SXIO and a public-key encryption scheme with sub-exponential security. Indistinguishability obfuscation ($i\mathcal{O}$) is a technique to make a class of circuits $\{\mathcal{C}_\lambda\}$ unintelligible in the sense that for any circuit $C \in \mathcal{C}_\lambda$, C and $i\mathcal{O}(1^\lambda, C)$ have the same output on all possible inputs and $i\mathcal{O}(1^\lambda, C_0)$ is indistinguishable from $i\mathcal{O}(1^\lambda, C_1)$ for any two circuits C_0, $C_1 \in \mathcal{C}_\lambda$ such that $C_0(x) = C_1(x)$, where λ is the security parameter. We want the size of $i\mathcal{O}(1^\lambda, C)$ bounded by $\text{poly}(\lambda, |C|)$. An SXIO has the same functionality as an indistinguishability obfuscator with nontrivial efficiency. The running time of SXIO on input $(1^\lambda, C)$ is at most $2^{n\gamma} \cdot \text{poly}(\lambda, |C|)$ where the circuit $C \in \mathcal{C}_\lambda$ takes an input of length n and γ (< 1) is the compressing factor. To get such an SXIO with an arbitrary compressing factor, it is required to have a multi-input SKFE [BKS15] which supports an unbounded polynomial number of functional keys. Also, 1-input single-key SKFE suffices to get an SXIO but with a restriction on compressing factor γ satisfying $\frac{1}{2} \le \gamma \le 1$ and such an SXIO can be used to build a sub-exponentially secure weakly sub-linear compact PKFE [BNPW16].

Remark 2. In [LPST16b], an $i\mathcal{O}$ is instantiated from a sub-exponentially secure and sub-linearly compact RE scheme in CRS model and a sub-exponentially secure pseudorandom generator (PRG). They followed the technique of GGM construction [GGM86] of building a PRF from a PRG using a tree. To get an $i\mathcal{O}$, the PRG in the GGM construction is replaced with a sub-exponentially secure sub-linear compact RE in CRS model. Let $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be a circuit class with maximum size S, input size n, output size l and the running time bound T. The obfuscation procedure for a circuit $C \in \mathcal{C}_\lambda$ works as follows:

- We generate $(\text{crs}_i, pk_i) \leftarrow \text{RE.Setup}(1^\lambda, 1^S, 1^n, 1^T, 1^l)$, for $i \in \{0, 1, \ldots, n\}$, where crs_i is a common reference string and pk_i is an encoding key. Let $\overrightarrow{\text{crs}} = \{\text{crs}_i\}_{i=0}^n$, $\overrightarrow{pk}_i = \{pk_j\}_{j=i}^n$.

- We construct an input less Turing machine $\Pi[\overrightarrow{pk}_{i+1}, C, z, \alpha^i_{z_i}]$ where hardcoded entities are $\overrightarrow{pk}_{i+1}, C, z = z_1z_2\ldots z_i \in \{0,1\}^i$ and a string $\alpha^i_{z_i} \in \{0,1\}^{2p(\lambda,i)}$ (p being a polynomial depending on $\lambda, i)$[3] for all $i \in \{0, 1, \ldots, n-1\}$. When $i = 0$, z is the null string ϵ and $\alpha^i_{z_i}$ is a random string $\alpha \xleftarrow{\$} \{0,1\}^{2p(\lambda,0)}$. The Turing machine $\Pi[\overrightarrow{pk}_1, C, \epsilon, \alpha]$ computes randomized encodings of $\Pi[\overrightarrow{pk}_2, C, 0, \alpha^1_0]$ and $\Pi[\overrightarrow{pk}_2, C, 1, \alpha^1_1]$ where $(\alpha^1_0, \alpha^1_1) \leftarrow \text{PRG}(\alpha)$ with $|\alpha^1_0| = |\alpha^1_1| = 2p(\lambda, 1)$, PRG being a sub-exponentially secure pseudorandom generator. To be more specific, the Turing machine $\Pi[\overrightarrow{pk}_1, C, \epsilon, \alpha]$ first generates $(\alpha^1_0, \alpha^1_1) \leftarrow \text{PRG}(\alpha)$ and uses the randomness α^1_0 to compute encoding $\widetilde{\Pi}[\overrightarrow{pk}_2, C, 0, \alpha^1_0] \leftarrow\text{RE.Enc}(pk_1, \Pi[\overrightarrow{pk}_2, C, 0, \alpha^1_0], \epsilon)$ and the randomness α^1_1 to compute the encoding $\widetilde{\Pi}[\overrightarrow{pk}_2, C, 1, \alpha^1_1] \leftarrow\text{RE.Enc}(pk_1, \Pi[\overrightarrow{pk}_2, C, 1, \alpha^1_1], \epsilon)$. More generally, the Turing machine $\Pi[\overrightarrow{pk}_{i+1}, C, z, \alpha^i_{z_i}]$ computes randomized encodings $\widetilde{\Pi}[\overrightarrow{pk}_{i+2}, C, z0, \alpha^{i+1}_0] \leftarrow \text{RE.Enc}(pk_{i+1}, \Pi[\overrightarrow{pk}_{i+2}, C, z0, \alpha^{i+1}_0], \epsilon)$ and $\widetilde{\Pi}[\overrightarrow{pk}_{i+2}, C, z1, \alpha^{i+1}_1] \leftarrow \text{RE.Enc}(pk_{i+1}, \Pi[\overrightarrow{pk}_{i+2}, C, z1, \alpha^{i+1}_1], \epsilon)$, where $(\alpha^{i+1}_0, \alpha^{i+1}_1) \leftarrow \text{PRG}(\alpha^i_{z_i})$ for $i \in \{1, 2, \ldots, n-1\}$. When $i = n$, the machine $\Pi[\overrightarrow{pk}_{i+1}, C, z, \alpha^i_{z_i}]$ outputs $C(z)$. We denote the class of all such Turing machines associated with the class of circuits $\{\mathcal{C}_\lambda\}$ as $\{\mathcal{M}_\lambda\}$.

- We compute an encoding $\widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha] \leftarrow \text{RE.Enc}(pk_0, \Pi[\overrightarrow{pk}_1, C, \epsilon, \alpha], \epsilon)$. Next, we construct the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha], \overrightarrow{\text{crs}}]$ as described in Figure 6 which takes input an n bit string $z = z_1z_2\cdots z_n$. For each $i \in \{0, 1, \ldots, n-1\}$, the circuit recursively computes $\text{RE.Eval}(\widetilde{\Pi}[\overrightarrow{pk}_{i+1}, C, z_1z_2\cdots z_i, \alpha^i_{z_i}], \text{crs}_i)$ which by correctness of RE, is equal to the output of the Turing machine $\Pi[\overrightarrow{pk}_{i+1}, C, z_1z_2\cdots z_i, \alpha^i_{z_i}]$ i.e. two randomized encodings $\widetilde{\Pi}[\overrightarrow{pk}_{i+2}, C, z_1z_2\cdots z_i0, \alpha^{i+1}_0]$ and $\widetilde{\Pi}[\overrightarrow{pk}_{i+2}, C, z_1z_2\cdots z_i1, \alpha^{i+1}_1]$ (as in line 3 of Figure 6). Finally, the circuit returns $\text{RE.Eval}(\widetilde{\Pi}[\overrightarrow{pk}_{n+1}, C, z, \alpha^n_{z_n}], \text{crs}_n)$ which actually is equal to $C(z)$. The obfuscation of the circuit C is $i\mathcal{O}(1^\lambda, C) = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha], \overrightarrow{\text{crs}}]$.

- To evaluate the circuit C for an input z, we compute $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha], \overrightarrow{\text{crs}}](z)$.

Lin et al. [LPST16b] proved that for any pair of functionally equivalent circuits C_0, $C_1 \in \mathcal{C}_\lambda$, the joint distribution $(\widetilde{\Pi}[\overrightarrow{pk}_1, C_0, \epsilon, \alpha], \overrightarrow{\text{crs}})$ is indistinguishable from $(\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\text{crs}})$. In particular, they have shown using the method of induction that for any label $i \in \{0, 1, \ldots, n\}$, $z \in \{0,1\}^i$ the joint distributions $(\widetilde{\Pi}[\overrightarrow{pk}_{i+1}, C_0, z, \alpha^i_{z_i}], \overrightarrow{\text{crs}}_i, \overrightarrow{pk}_i)$ and $(\widetilde{\Pi}[\overrightarrow{pk}_{i+1}, C_1, z, \alpha^i_{z_i}], \overrightarrow{\text{crs}}_i, \overrightarrow{pk}_i)$ are indistinguishable. The indistinguishability was achieved by the simulation security of the RE scheme.

Theorem 8. [LPST16b] *Assuming the existence of sub-exponentially secure one-way functions, if there exists a sublinearly compact randomized encoding scheme in the CRS model with sub-exponential simulation security, then there exists an bounded-input indistinguishability obfuscator for Turning machines.*

[3] For every $\lambda \in \mathbb{N}, i \leq 2^\lambda$, $p(\lambda, i) = p(\lambda, i-1) + (2d\lambda)^{1/\epsilon}$ and $p(\lambda, -1) = \lambda$ where ϵ is a constant associated with the sub-exponential security of PRG, $d > 0$ is any constant strictly greater than c and the constant c represents the security loss associated with the indistinguishability security of RE (section 4, [LPST16b]).

Hardwired: $\widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha]$, $\overrightarrow{\text{crs}}$
Input: an input $z = (z_1 z_2 \ldots z_n)$

1. $\widetilde{\Pi} \leftarrow \widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha]$, $i \leftarrow 0$
2. **while** $i < n$ **do**
3. $\quad (\widetilde{\Pi}[\overrightarrow{pk}_{i+2}, C, z_1 z_2 \cdots z_i 0, \alpha_0^{i+1}], \widetilde{\Pi}[\overrightarrow{pk}_{i+2}, C, z_1 z_2 \cdots z_i 1, \alpha_1^{i+1}]) \leftarrow \text{RE.Eval}(\widetilde{\Pi}, \text{crs}_i)$
4. $\quad \widetilde{\Pi} \leftarrow \widetilde{\Pi}[\overrightarrow{pk}_{i+2}, C, z_1 z_2 \cdots z_i z_{i+1}, \alpha_{z_{i+1}}^{i+1}]$
5. **end do**
6. **return** RE.Eval$(\widetilde{\Pi}, \text{crs}_i)$

Figure 6: **The Special Circuit** $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha], \overrightarrow{\text{crs}}]$

Hardwired: a pPRF key K.
Input: an instance $x \in \mathcal{X} = \{0,1\}^k$ and a witness $w \in \mathcal{W} = \{0,1\}^{n-k}$.
Padding: the circuit is padded to size pad = pad(s, n, λ), determined in the analysis.

1. **if** $R(x, w) = 1$ **then**
2. $\quad y \leftarrow$ pPRF.Eval(K, x).
3. **else** $y \leftarrow \perp$
4. **end if**
5. **return** y

Figure 7: **Evaluation Circuit** $E = \mathsf{EC}[K]$

We stress that RE.Enc$(pk_0, \Pi[\overrightarrow{pk}_1, C, \epsilon, \alpha], \epsilon)$ is actually a ciphertext obtained from the encryption algorithm of underlying PKFE that uses $(\Pi[\overrightarrow{pk}_1, C, \epsilon, \alpha], \epsilon, 0^{\lambda+1})$ as the plaintext. The size of the special circuit $\mathcal{G}$ is bounded by poly$(\lambda, |C|, T)$ and runtime of $\mathcal{G}$ on input z is bounded by poly$(\lambda, |z|, |C|, T)$. We will use the notation $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C, \epsilon, \alpha], \overrightarrow{\text{crs}}]$ for obfuscating a circuit C using a randomized encoding scheme in CRS model.

3 Our Witness PRF

Construction 1. We describe our construction of witness PRF (wPRF) that uses a puncturable pseudorandom function pPRF = (Gen, Eval, Punc) with domain $\mathcal{X} = \{0,1\}^k$ and range $\mathcal{Y}$ and a randomized encoding scheme RE = (Setup, Enc, Eval) which is a bounded input sub-linearly compact randomized encoding scheme in CRS model. Our scheme wPRF = (Gen, F, Eval) for an NP language L with relation circuit $R : \mathcal{X} \times \mathcal{W} \rightarrow \{0,1\}$, $\mathcal{X} = \{0,1\}^k$, $\mathcal{W} = \{0,1\}^{n-k}$ and $|R| \leq s$, is given by the following algorithms.

- $(\mathsf{fk}, \mathsf{ek}) \leftarrow w$PRF.Gen$(1^\lambda, R)$: A trusted third party generates a secret function key fk and a public evaluation key ek for a relation R by executing the following steps where λ is a security parameter.
  - Choose a pPRF key $K \leftarrow$ pPRF.Gen(1^λ) where $K \in \{0,1\}^\lambda$.
  - Construct the circuit $E = \mathsf{EC}[K] \in \{E_\lambda\}$ as defined in Figure 7. Let the circuit E be of size S with input size n, output size l and T is the runtime bound of the circuit.
  - Generate $(\text{crs}_i, pk_i) \leftarrow$RE.Setup$(1^\lambda, 1^S, 1^n, 1^T, 1^l)$ for $i \in \{0, 1, \ldots, n\}$ where crs_i is a common reference string and pk_i is an encoding key. We define $\overrightarrow{\text{crs}} = \{\text{crs}\}_{i=0}^n$ and

$\overrightarrow{\mathrm{pk}}_i = \{\mathrm{pk}_j\}_{j=i}^n$.

– Compute the randomized encoding $\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha] \leftarrow \mathrm{RE.Enc}(pk_0, \Pi[\overrightarrow{pk}_1, E, \epsilon, \alpha], \epsilon)$ where ϵ is a null string, α is a random binary string and $\Pi[\overrightarrow{pk}_1, E, \epsilon, \alpha]$ is a Turing machine defined in Remark 2.
– Build the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ as described in Figure 6.
– Set $\mathsf{fk} = K$, $\mathsf{ek} = \mathcal{G}\ [\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ and output $(\mathsf{fk}, \mathsf{ek})$. The secret function key fk is sent to a user over a secure channel and the evaluation key ek is made public.

- $y \leftarrow w\mathrm{PRF.F}(\mathsf{fk}, x)$: This algorithm is run by the user who has a secret function key fk and outputs a wPRF value $y \leftarrow \mathrm{pPRF.Eval}(K, x) \in \mathcal{Y}$ for an instance $x \in \mathcal{X}$ using the secret function key fk as a pPRF key K.

- $w\mathrm{PRF.Eval}(\mathsf{ek}, x, w)$: An evaluator takes a witness $w \in \mathcal{W}$ for $x \in L$ and uses the public evaluation key $\mathsf{ek} = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ to get back the wPRF value as $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}](z)$ where $z = (x, w) \in \{0,1\}^n$.

Correctness. The output of wPRF.F for an instance x is a pPRF evaluation $y \leftarrow \mathrm{pPRF.Eval}(K, x) \in \mathcal{Y}$ on x using the secret key $K \in \{0,1\}^\lambda$. On the other hand, for wPRF.Eval an witness-holder computes $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]\ (z)$ where $z = (x, w)$. By the correctness of randomized encoding scheme as discussed in Remark 2, we have $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}](z) = E(x, w)$. The circuit E (Figure 7) on input x, w, first checks whether $R(x, w) = 1$ holds. If this condition is satisfied, then it outputs $y \leftarrow \mathrm{pPRF.Eval}(K, x) \in \mathcal{Y}$ using the hardcoded key K. Therefore a valid witness-holder of $x \in L$ can recompute the wPRF value $y \in \mathcal{Y}$ associated with x using the witness w and the evaluation key $\mathsf{ek} = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$. Note that, if w is not valid witness for $x \in L$ then the output of $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}](z) = E(x, w)$ is the distinguished symbol $\perp$. Therefore, our witness PRF follows the correctness property stated in Equation 1, Definition 6.

Padding Parameter. The proof of security relies on the indistinguishability of randomized encodings of the machines $\Pi[\overrightarrow{pk}_1, E, \epsilon, \alpha]$ and $\Pi[\overrightarrow{pk}_1, E^*, \epsilon, \alpha]$(where E and E^* are defined in Figure 7 and 8). For this we set $\mathrm{pad} = \max(|E|, |E^*|)$. The circuits E and E^* compute the relation circuit R on an input (x, w) of size n and evaluate a puncturable PRF over the domain $\mathcal{X} = \{0,1\}^k$ of size 2^k using a hardwired element which is a simple pPRF key for E or a punctured pPRF key for E^*. Thus $\mathrm{pad} \leq \mathrm{poly}(\lambda, s, k)$ where s is the size of the relation circuit R.

Efficiency. In this analysis, we discuss the size of wPRF.F and wPRF.Eval. The size of $\mathcal{X}$ is 2^k and wPRF.F includes a PRF evaluation over the domain $\mathcal{X}$. Therefore, size of wPRF.F is bounded by $\mathrm{poly}(\lambda, k)$. We note that, wPRF.Eval only runs the circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ over an input of size n. The running time of $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ is $\mathrm{poly}(\lambda, n, |E|, T) = \mathrm{poly}(\lambda, n, k, s, T)$ and the size of $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ is $\mathrm{poly}(\lambda, |E|, T) = \mathrm{poly}(\lambda, k, s, T)$. In particular, the running time and size of wPRF.Eval are respectively $\mathrm{poly}(\lambda, n, k, s, T)$ and $\mathrm{poly}(\lambda, k, s, T)$. Here we note that the runtime T of the circuit E is bounded by the runtime of the relation R and the runtime of a pPRF evaluation. So, $T \leq T_R+$ $\mathrm{poly}(\lambda, k)$ where T_R is the runtime of the relation circuit R on input (x, w) of size n. Hence, the runtime of wPRF.Eval is bounded by $\mathrm{poly}(\lambda, n, k, s, T_R)$ and size of wPRF.Eval is bounded by $\mathrm{poly}(\lambda, k, s, T_R)$.

Theorem 9. *Assuming* LWE *with sub-exponential hardness and the existence of δ-sub-exponentially secure one-way functions, if there exists a weakly sub-linear compact public key functional encryption scheme for* $\mathsf{P/poly}$ *with δ-sub-exponential security, then there exists a δ-secure witness* PRF *scheme.*

Hardwired: a punctured key $K\{x^*\}$.
Input: an instance $x \in \mathcal{X}$ and a witness $w \in \mathcal{W}$

1. **if** $R(x, w) = 1$ **then**
2. **if** $x = x^*$ **then**
3. **return** y^*
4. **else return** pPRF.Eval($K\{x^*\}, x$)
5. **end if**
6. **return** $\perp$

Figure 8: **Evaluation Circuit** $E^* = \mathsf{EC}[K\{x^*\}]$

Proof. The existence of sub-exponentially hard LWE and δ-sub-exponentially secure weakly sub-linear compact public key functional encryption scheme for $\mathsf{P/poly}$ imply a δ-sub-exponential simulation secure (Definition 18) randomized encoding scheme for Turing machines in CRS model (Theorem 7). We get a δ-secure puncturable PRF (Definition 3) from δ-secure one-way functions (Theorem 5). We need sub-exponentially secure one-way functions for getting a sub-exponentially secure pseudorandom generator (Theorem 4) which is required for constructing the special circuit $\mathcal{G}$ in Figure 6. The theorem follows directly from the following claim.

Claim 1. *Assume existence of δ-sub-exponentially secure one-way functions. Our construction 1 of* wPRF = (Gen, F, Eval) *is δ-selectively secure witness* PRF *if the* pPRF *integrated in our scheme is δ-secure puncturable* PRF *and the* RE *scheme employed in our scheme is a bounded input sub-linearly compact randomized encoding scheme in* CRS *model with δ-sub-exponential simulation security for the class of Turing machines $\{\mathcal{M}_\lambda\}$ associated with the circuit class $\{E_\lambda\}$.*

Proof. We prove this by showing that for any non-uniform PPT adversary $\mathcal{A}$, the distinguishing advantage between the two experiments $\mathrm{Expt}_{\mathcal{A}}^{w\mathrm{PRF}}(1^\lambda, 0)$ and $\mathrm{Expt}_{\mathcal{A}}^{w\mathrm{PRF}}(1^\lambda, 1)$ given in Figure 2 is negligible. We consider the following hybrid games.

Hybd$_0$ This is the standard security experiment $\mathrm{Expt}_{\mathcal{A}}^{w\mathrm{PRF}}(1^\lambda, 0)$ described in Figure 9.

Hybd$_1$ In this hybrid game we change $K \leftarrow \mathrm{pPRF.Gen}(1^\lambda)$ into a punctured key $K\{x^*\} \leftarrow \mathrm{pPRF.Punc}(K, x^*)$ and $\mathsf{ek} = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E^*, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ instead of $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ where $E^* = \mathsf{EC}[K\{x^*\}]$ is the circuit as defined in Figure 8 and $y^* \leftarrow \mathrm{pPRF.Eval}(K, x^*) \in \mathcal{Y}$. We note that the functionality and running time of both the circuits E and E^* are the same. Also, the size of the two machines $\Pi[\overrightarrow{pk}_1, E, \epsilon, \alpha]$ and $\Pi[\overrightarrow{pk}_1, E^*, \epsilon, \alpha]$ is the same due to padding. Therefore, the joint distribution $(\widetilde{\Pi}[\overrightarrow{pk}_{i+1}, E, z, \alpha^i_{z_i}], \overrightarrow{\mathrm{crs}}_i, \overrightarrow{pk}_i)$ is indistinguishable from $(\widetilde{\Pi}[\overrightarrow{pk}_{i+1}, E^*, z, \alpha^i_{z_i}], \overrightarrow{\mathrm{crs}}_i, \overrightarrow{pk}_i)$ for every label $i \in \{0, 1, \ldots, n\}$ and $z \in \{0,1\}^i$ (as discussed in Remark 2). Hence by simulation security of the RE scheme, we have $\mathrm{Hybd}_0 \approx_\delta \mathrm{Hybd}_1$ (notation explained in Table 1), i.e., these two hybrid games are computationally indistinguishable.

Hybd$_2$ This hybrid game is the same as previous one except that here we take y^* as a uniformly random element from $\mathcal{Y}$ instead of setting $y^* \leftarrow \mathrm{pPRF.Eval}(K, x^*) \in \mathcal{Y}$. From the pseudorandomness at punctured points (Definition 3) of the pPRF we have,

$$\begin{aligned}\mu(\lambda) &\geq |\Pr[\mathcal{A}(K\{x^*\}, \mathrm{pPRF.Eval}(K, x^*)) = 1] - \Pr[\mathcal{A}(K\{x^*\}, U) = 1]| \\ &\geq |\Pr[\mathrm{Hybd}_1(\lambda) = 1] - \Pr[\mathrm{Hybd}_2(\lambda) = 1]|\end{aligned}$$

1. The adversary $\mathcal{A}$ submits a challenge statement $x^* \in \mathcal{X} \setminus L$.
2. The challenger generates $(\mathsf{fk}, \mathsf{ek}) \leftarrow w\text{PRF.Gen}(1^\lambda, R)$ as follows and sends ek to $\mathcal{A}$:
   2.1 Chose $K \leftarrow \text{pPRF.Gen}(1^\lambda)$ and set $\mathsf{fk} = K$
   2.2 Construct the circuit $E = \mathsf{EC}[K]$ as defined in Figure 7
   2.3 Generate $(\text{crs}_i, pk_i) \leftarrow$RE.Setup$(1^\lambda, 1^S, 1^n, 1^T, 1^l)$ for $i \in \{0, 1, \ldots, n\}$ where S, n, T, l are the same as in Construction 1 and set $\overrightarrow{\text{crs}} = \{\text{crs}\}_{i=0}^n$ and $\overrightarrow{\text{pk}_i} = \{\text{pk}_j\}_{j=i}^n$
   2.4 Build the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\text{crs}}]$ as described in Figure 6 where $\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha] \leftarrow$ RE.Enc $(pk_0, \Pi[\overrightarrow{pk}_1, E, \epsilon, \alpha], \epsilon)$ and $\Pi[\overrightarrow{pk}_1, E, \epsilon, \alpha]$ is a Turing machine defined in Remark 2.
   2.5 Set $\mathsf{ek} = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\text{crs}}]$
3. The challenger computes $y^* \leftarrow w\text{PRF.F}(\mathsf{fk}, x^*) \in \mathcal{Y}$ and sends it to $\mathcal{A}$.
4. The adversary $\mathcal{A}$ can make polynomial number of quires for wPRF.F on some $x \in \mathcal{X} \setminus \{x^*\}$ to the challenger and receives $w\text{PRF.F}(\mathsf{fk}, x)$.
5. The adversary $\mathcal{A}$ outputs a bit b'.

Figure 9: Hybd_0 associated with our wPRF

for infinitely many λ and a negligible function μ where U denotes uniform distribution over the domain $\mathcal{Y}$ of pPRF.Eval. Since the pPRF is δ-secure, we have $\mu(\lambda) \leq \delta(\lambda)^{\omega(1)}$. Thus it holds that $\text{Hybd}_1 \approx_\delta \text{Hybd}_2$.

Hybd_3 In this hybrid game, again we consider $\mathsf{ek} = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E, \epsilon, \alpha], \overrightarrow{\text{crs}}]$ corresponding to the circuit $E = \mathsf{EC}[K]$ as in the original experiment Hybd_0. Everything else is the same as in Hybd_2. Following the similar argument as in Hybd_1, we have $\text{Hybd}_2 \approx_\delta \text{Hybd}_3$.

Note that Hybd_3 is actually the regular experiment $\text{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, 1)$. Hence, by the above sequence of hybrid arguments, $\text{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, 0)$ is indistinguishable from $\text{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, 1)$ and we write $\text{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, 0) \approx_\delta \text{Expt}_{\mathcal{A}}^{w\text{PRF}}(1^\lambda, 1)$.
This completes the proof of Theorem 9. □

4 Our Offline Witness Encryption

Construction 2. We now construct an offline witness encryption scheme OWE = (Setup, Enc, Dec) for any NP language L with relation circuit $R : \mathcal{X} \times \mathcal{W} \to \{0, 1\}$ based on an extractable witness PRF (wPRF) and a randomized encoding (RE) scheme in CRS model. The main ingredients and notations used in this proposed construction are the following:

(i) A public-key encryption PKE = (Gen, Enc, Dec) with CPA-security.

(ii) An extractable witness PRF wPRF = (Gen, F, Eval) for the NP language $L' = \{(c_1, c_2, \text{PK}_1, \text{PK}_2) : \exists\ (x, m, r_1, r_2)$ such that $c_i = \text{PKE.Enc}(\text{PK}_i, (x, m); r_i)$ for $i = 1, 2\}$ with the relation $R' : \chi' \times \mathcal{W}' \to \{0, 1\}$. Therefore, $R'((c_1, c_2, \text{PK}_1, \text{PK}_2), (x, m, r_1, r_2)) = 1$ if c_1 and c_2 are both encryptions of the same message (x, m) using public keys PK_1, PK_2 and randomness r_1, r_2 respectively; otherwise 0. Here we assume that message, ciphertext of the PKE and the wPRF value can be represented as bit-strings.

(iii) A sub-linearly compact bounded input randomized encoding scheme RE = (Setup, Enc, Eval) in CRS model with δ-sub-exponential simulation security for Turing machines.
We describe below our offline witness encryption scheme OWE = (Setup, Enc, Dec) for an NP language L with relation circuit $R : \mathcal{X} \times \mathcal{W} \to \{0, 1\}$.

- $(pp_e, pp_d) \leftarrow$ OWE.Setup$(1^\lambda, R)$: This is run by a trusted authority to generate public parameters for both encryption and decryption where R is a relation circuit and λ is a security parameter. It works as follows:
  - Obtain two pairs of PKE keys $(\mathrm{SK}_1, \mathrm{PK}_1) \leftarrow$ PKE.Gen(1^λ) and $(\mathrm{SK}_2, \mathrm{PK}_2) \leftarrow$ PKE.Gen(1^λ).
  - Generate $(\mathsf{fk}, \mathsf{ek}) \leftarrow w$PRF.Gen$(1^\lambda, R')$ for the relation circuit R' defined in (ii).
  - Construct the circuit $C_1 = \mathsf{MOC}[\mathrm{SK}_1, \mathsf{fk}] \in \{\mathcal{C}_\lambda\}$ as defined in Figure 10. Let S be the size of the circuit C_1 with input size n, output size l and T is the runtime bound of the circuit on an input of size n.
  - Generate $(\mathrm{crs}_i, pk_i) \leftarrow$ RE.Setup$(1^\lambda, 1^S, 1^n, 1^T, 1^l)$ for $i \in \{0, 1, \ldots, n\}$ where crs_i is a common reference string and pk_i is an encoding key. Set $\overrightarrow{\mathrm{crs}} = \{\mathrm{crs}\}_{i=0}^n$ and $\overrightarrow{\mathrm{pk}}_i = \{\mathrm{pk}_j\}_{j=i}^n$.
  - Compute the randomized encoding $\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha] \leftarrow$ RE.Enc$(pk_0, \Pi[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \epsilon)$ where ϵ is a null string, α is a random binary string and $\Pi[\overrightarrow{pk}_1, C_1, \epsilon, \alpha]$ is a Turing machine defined in Remark 2.
  - Construct the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ as described in Figure 6.
  - Set $pp_e = (\mathrm{PK}_1, \mathrm{PK}_2, \mathsf{ek})$, $pp_d = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$.
  - Output (pp_e, pp_d).
- $c \leftarrow$ OWE.Enc$(1^\lambda, x, m, pp_e)$: An encrypter encrypts a message $m \in \mathcal{M}$ with respect to an NP statement $x \in \mathcal{X}$ using the public parameters for encryption pp_e and produces a ciphertext as follows:
  - Choose two random strings $r_1, r_2 \xleftarrow{\$} \{0,1\}^{l_{\mathrm{PKE}}(\lambda)}$ where l_{PKE} is a polynomial in λ.
  - Compute two ciphertexts $c_i = \mathrm{PKE.Enc}(\mathrm{PK}_i, (x, m); r_i)$ for $i = 1, 2$.
  - Generate a wPRF evaluation of the statement $(c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2)$ with witness (x, m, r_1, r_2) as $y \leftarrow w\mathrm{PRF.Eval}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m, r_1, r_2))$.
  - Output $c = (c_1, c_2, x, y)$ as ciphertext.
- OWE.Dec(c, w, pp_d): On receiving a ciphertext c, a receiver who has a witness w for $x \in L$, runs this algorithm using public parameter $pp_d = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ for decryption to learn the message by outputting $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}](z)$ where $z = (c, w)$.

Correctness. The ciphertext c has four components where first two components c_1, c_2 are the encryptions of the same message (x, m), the third one is a statement $x \in L$ and the last component y is a wPRF evaluation of the statement $(c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2)$ with witness (x, m, r_1, r_2). Therefore, we pass the check at line 2 of the circuit C_1 (Figure 10) as the correctness of wPRF scheme (Equation 1, Definition 6) implies $w\mathrm{PRF.Eval}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m, r_1, r_2)) = w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2))$. We recover $(x, m) \leftarrow \mathrm{PKE.Dec}(\mathrm{SK}_1, c_1)$ in line 3 of the circuit C_1 assuming the exactness of the PKE scheme. If $w \in \mathcal{W}$ is a valid witness for the statement $x \in L$, then $R(x, w) = 1$ and the circuit C_1 returns the message $m \in \mathcal{M}$. Finally, by the correctness of RE as described in Remark 2, we have $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}](z) = C_1(z) = m$ where $z = (c, w)$. Hence Equation 4 of Definition 13 holds for this construction which establishes the correctness of our OWE scheme.

Efficiency. The encryption algorithm OWE.Enc computes two public-key encryption on a message of size $(|x| + |m|)$ and one wPRF evaluation of an input of the form $(c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2)$ with

Hardwired: a PKE secret key SK_j, a wPRF function key fk.
Input: a ciphertext c and a witness $w \in \mathcal{W}$

1. Parse $c = (c_1, c_2, x, y)$
2. **if** $(w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2)) = y)$ **then**
3. $\quad(\hat{x}, \hat{m}) \leftarrow \mathrm{PKE.Dec}(\mathrm{SK}_j, c_j)$
4. $\quad$**if** $((\hat{x} = x) \bigwedge (R(\hat{x}, w) = 1))$ **then**
5. $\qquad$**return** $\hat{m}$
6. $\quad$**end if**
7. **end if**
8. **return** $\perp$

Figure 10: **Message Output Circuit** $C_j = \mathsf{MOC}[\mathrm{SK}_j, \mathsf{fk}]$, $j = 1, 2$

size-bound $\mathrm{poly}(\lambda, |x| + |m|)$ using a witness of the form (x, m, r_1, r_2) with size-bound $(|x| + |m|$ $+ 2.\mathrm{poly}(\lambda))$. Therefore, time of encryption is bounded by the time of PKE.Enc and evaluation time of wPRF and we have that $\mathsf{Time}_{\mathrm{OWE.Enc}} \leq 2.\ \mathrm{poly}(\lambda, |x|+|m|) + \mathrm{poly}(\lambda, \mathrm{poly}(\lambda, |x|+|m|)+$ $(|x| + |m| + 2.\mathrm{poly}(\lambda)), s_{R'}, T_{R'})$ where $s_{R'}$ and $T_{R'}$ are the size and time-bound for the relation R' defined as $R'((c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m, r_1, r_2)) = 1$ if and only if $\mathrm{PKE.Enc}(\mathrm{PK}_i, (x, m); r_i) = c_i$ for $i = 1, 2$. Thus R' verifies that two given ciphertexts c_1, c_2 are encryptions of the same message (x, m) with randomness r_1, r_2 respectively under respective public keys $\mathrm{PK}_1, \mathrm{PK}_2$. Hence, $s_{R'}$, $T_{R'}$ both can be bounded by $\mathrm{poly}(\lambda, |x| + |m|)$. Therefore, OWE.Enc has the time-bound $\mathsf{Time}_{\mathrm{OWE.Enc}} \leq \mathrm{poly}(\lambda, |x| + |m|)$.

Theorem 10. *Assuming the existence of sub-exponentially secure one-way functions, our construction 2 of* OWE = (Setup, Enc, Dec) *is δ-selectively secure offline witness encryption if the underlying* PKE *utilized in our* OWE *is a δ-selectively secure public-key encryption (Definition 4), the* wPRF *integrated in our* OWE *is an extractable witness* PRF *(Definition 8) and the* RE *employed in our* OWE *is a bounded input δ-sub-exponential simulation secure (Definition 18) sub-linear compact randomized encoding scheme (Definition 21) in* CRS *model for the class of Turing machines $\{\mathcal{M}_\lambda\}$ associated with the class of circuits $\{\mathcal{C}_\lambda\}$.*

Proof. We show that the distinguishing advantage between two experiments $\mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, 0)$ and $\mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, 1)$ (see Figure 4) for any non-uniform PPT adversary $\mathcal{A}$ is negligible by defining the following sequence of hybrid games and thereby, prove the indistinguishability between them. Let the challenge messages be m_0 and m_1.

Hybd$_0$ The first game is the standard selective security experiment $\mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, 0)$ where the adversary $\mathcal{A}$ is given the ciphertext corresponding to the message m_0. We describe it in Figure 11.

Hybd$_1$ In this hybrid game, we pick y uniformly at random from $\mathcal{Y}$ instead of setting $y \leftarrow w\mathrm{PRF.Eval}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m_0, r_1, r_2))$. All other contents of the ciphertext c remain the same as in the previous game. Since the random values r_1 and r_2 which are the part of witnesses corresponding to the ciphertexts c_1 and c_2 respectively are not known to the adversary, by the extractable nature of wPRF (Definition 8) and the CPA-security of the underlying PKE scheme (Definition 5) these two hybrid games are computationally indistinguishable. We prove this in the following claim.

Claim 2. *Assuming the* PKE *is a δ-selectively secure public-key encryption and the* wPRF

1. The adversary chooses $(x, m_0, m_1, st) \leftarrow \mathcal{A}(1^\lambda)$ and sends it to the challenger. Here $x \notin L$, $|m_0| = |m_1|$ and st contains some auxiliary information.
2. The challenger generates public parameters $(pp_e, pp_d) \leftarrow$ OWE.Setup$(1^\lambda, R)$ as follows and sends it to $\mathcal{A}$:
   - 2.1 Generate $(\mathrm{SK}_i, \mathrm{PK}_i) \leftarrow$ PKE.Gen(1^λ) for $i = 1, 2$ and $(\mathsf{fk}, \mathsf{ek}) \leftarrow w$PRF.Gen$(1^\lambda, R')$ where relation R' is the same as in Construction 2
   - 2.2 Set $pp_e = (\mathrm{PK}_1, \mathrm{PK}_2, \mathsf{ek})$
   - 2.3 Construct the message output circuit $C_1 = \mathsf{MOC}[\mathrm{SK}_1, \mathsf{fk}]$ (see Figure 10)
   - 2.4 Generate $(\mathrm{crs}_i, pk_i) \leftarrow$ RE.Setup$(1^\lambda, 1^S, 1^n, 1^T, 1^l)$ for $i \in \{0, 1, \ldots, n\}$ where S, n, T, l are the same as in Construction 2 and set $\overrightarrow{\mathrm{crs}} = \{\mathrm{crs}\}_{i=0}^n$ and $\overrightarrow{\mathrm{pk}_i} = \{\mathrm{pk}_j\}_{j=i}^n$
   - 2.5 Construct the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ as described in Figure 6, where $\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha] \leftarrow$ RE.Enc$(pk_0, \Pi[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \epsilon)$ and $\Pi[\overrightarrow{pk}_1, C_1, \epsilon, \alpha]$ is a Turing machine defined in Remark 2.
   - 2.6 Set $pp_d = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$
3. The challenger produces the ciphertext $c \leftarrow$ OWE.Enc$(1^\lambda, x, m_0, pp_e)$ as follows and submits it to $\mathcal{A}$:
   - 3.1 Chose $r_1, r_2 \xleftarrow{\$} \{0,1\}^{l_{\mathrm{PKE}}(\lambda)}$
   - 3.2 Compute $c_1 \leftarrow$PKE.Enc$(\mathrm{PK}_1, (x, m_0); r_1)$, $c_2 \leftarrow$PKE.Enc$(\mathrm{PK}_2, (x, m_0); r_2)$
   - 3.3 Evaluate $y \leftarrow w$PRF.Eval$(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m_0, r_1, r_2))$
   - 3.4 Set $c \leftarrow (c_1, c_2, x, y)$
4. The adversary observing (st, c, pp_e, pp_d), outputs a bit $b' \leftarrow \mathcal{A}(st, c, pp_e, pp_d)$.

Figure 11: $\mathrm{Hybd}_0(\lambda)$ associated with our OWE

is extractable, Hybd_0 *and* Hybd_1 *are δ-indistinguishable.*

Proof. Let us prove this by contradiction. Suppose, the OWE-adversary $\mathcal{A}$ can distinguish between Hybd_0 and Hybd_1. Hence, there exist a polynomial $p(\cdot)$ such that for infinitely many λ,

$$|\Pr[\mathrm{Hybd}_0(\lambda) = 1] - \Pr[\mathrm{Hybd}_1(\lambda) = 1]| \geq \tfrac{1}{2} + \tfrac{1}{p(\lambda)}.$$

We note that in Hybd_0, y is computed as $w\mathrm{PRF.Eval}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m_0, r_1, r_2))$ which is in fact equal to $w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2)) \in \mathcal{Y}$ (by the correctness of wPRF) and in Hybd_1, y is chosen uniformly from $\mathcal{Y}$. Therefore the distinguishing advantage of $\mathcal{A}$ is the same advantage as in Equation 2 (Definition 8) and hence (from Equation 3) the extractable security of wPRF implies that there exists a PPT extractor $\mathcal{E}$ and a polynomial $q(\cdot)$ such that

$$|\Pr[w' \leftarrow \mathcal{E}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), \mathsf{Aux}, y, \{x_i\}, r) : R'((c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), w') = 1]| \geq \tfrac{1}{q(\lambda)}$$

where $\{x_i\}$ are the same wPRF queries made by $\mathcal{A}$ and r is the random coin used by $\mathcal{A}$. We can construct a PKE-adversary $\mathcal{B}$ against the CPA security (Definition 5) of PKE scheme for the key PK_2 as described in Figure 12.

If $c'_b \leftarrow$ PKE.Enc$(\mathrm{PK}_2, (x, m_0); r_2)$, then $y = w\mathrm{PRF.Eval}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m_0, r_1, r_2)) = w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2))$ and hence the PKE-adversary $\mathcal{B}$ simulates Hybd_0. If $c'_b \leftarrow$ PKE.Enc$(\mathrm{PK}_2, (x, m_1); r_2)$, then $w\mathrm{PRF.Eval}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m_0, r_1, r_2)) \neq w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2))$ with high probability and $y = w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2))$ acts like a random y from $\mathcal{Y}$ which implies that $\mathcal{B}$ simulates Hybd_1.

By hypothesis, the adversary $\mathcal{A}$ can distinguish between the hybrids Hybd_0 and Hybd_1, and hence there exists an extractor $\mathcal{E}$ that, on input $\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), \mathsf{Aux}, y, \{x_i\}, r$, is able to find a witness $w' = (x, m_b, r_1, r_2)$, where Aux contains $st_{\mathcal{A}}$, the wPRF queries $\{x_i\}$ and r indicates the random coin used by $\mathcal{A}$. Therefore, the OWE-adversary $\mathcal{A}$ can

1. The PKE-challenger runs PKE.Gen$(1^\lambda) \to (\mathrm{SK}_2, \mathrm{PK}_2)$ and makes Pk_2 public.
2. The PKE-adversary $\mathcal{B}$ submits the challenge messages m'_0, m'_1 to the PKE-challenger as follows with some auxiliary information st and $|m'_0| = |m'_1|$:
   - 2.1 Invoke OWE-adversary $\mathcal{A}$ to obtain $(x, m_0, m_1, st_{\mathcal{A}}) \leftarrow \mathcal{A}(1^\lambda)$ where $x \notin L$ and $|m_0| = |m_1|$
   - 2.2 Genereate $(\mathrm{SK}_1, \mathrm{PK}_1) \leftarrow \mathrm{PKE.Gen}(1^\lambda)$
   - 2.3 Compute $(\mathsf{fk}, \mathsf{ek}) \leftarrow w\mathrm{PRF.Gen}(1^\lambda, R')$ for the relation R' defined in Construction 2
   - 2.4 Choose $r_1 \xleftarrow{\$} \{0,1\}^{l_{\mathrm{PKE}}(\lambda)}$
   - 2.5 Compute $c_1 = \mathrm{PKE.Enc}(\mathrm{PK}_1, (x, m_0); r_1)$
   - 2.6 Set $m'_0 = (x, m_0)$, $m'_1 = (x, m_1)$ and $st = (\mathrm{SK}_1, \mathrm{PK}_1, \mathsf{fk}, \mathsf{ek}, c_1, r_1, x, m_0, m_1, st_{\mathcal{A}})$
3. The PKE-challenger chooses a random bit $b \in \{0,1\}$ and sends the ciphertext $c'_b \leftarrow \mathrm{PKE.Enc}(\mathrm{PK}_2, m'_b = (x, m_b); r_2)$ to $\mathcal{B}$, where $r_2 \xleftarrow{\$} \{0,1\}^{l_{\mathrm{PKE}}(\lambda)}$.
4. The PKE-adversary $\mathcal{B}$ simulates $\mathcal{A}$ to output a guess for b by observing (st, c'_b) as foloows:
   - 4.1 Set $c_2 = c'_b$
   - 4.2 Compute $y = w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2))$
   - 4.3 Construct the message output circuit $C_1 = \mathsf{MOC}[\mathrm{SK}_1, \mathsf{fk}]$ (see Figure 10)
   - 4.4 Generate $(\mathrm{crs}_i, pk_i) \leftarrow \mathrm{RE.Setup}(1^\lambda, 1^S, 1^n, 1^T, 1^l)$ for $i \in \{0, 1, \ldots, n\}$ where S, n, T, l are the same as in Construction 2 and set $\overrightarrow{\mathrm{crs}} = \{\mathrm{crs}\}_{i=0}^n$ and $\overrightarrow{\mathrm{pk}_i} = \{\mathrm{pk}_j\}_{j=i}^n$
   - 4.5 Construct the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ as described in Figure 6, where $\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha] \leftarrow \mathrm{RE.Enc}(pk_0, \Pi[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \epsilon)$ and $\Pi[\overrightarrow{pk}_1, C_1, \epsilon, \alpha]$ is a Turing machine defined in Remark 2.
   - 4.6 Set $c = (c_1, c_2, x, y)$, $pp_e = (\mathrm{PK}_1, \mathrm{PK}_2, \mathsf{ek})$, $pp_d = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ and send $(st_{\mathcal{A}}, c, pp_e, pp_d)$ to the OWE-adversary $\mathcal{A}$
   - 4.7 Output a guess $b' \leftarrow \mathcal{A}(st_{\mathcal{A}}, c, pp_e, pp_d)$ for b

Figure 12: The PKE-adversary $\mathcal{B}$ simulating Hybd_1

learn the bit b chosen by the PKE-challenger while executing the CPA security game and sends it to the PKE-adversary $\mathcal{B}$ which helps in breaking the CPA security of the PKE scheme for the key PK_2. Consequently, we have for infinitely many λ,

$$\tfrac{1}{q(\lambda)} \leq |\Pr[w' \leftarrow \mathcal{E}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), \mathsf{Aux}, y, \{x_i\}, r) : R'((c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), w') = 1]|$$
$$= |\Pr[\mathrm{Expt}_{\mathcal{B}}^{\mathrm{PKE}}(1^\lambda, 0) = 1] - \Pr[\mathrm{Expt}_{\mathcal{B}}^{\mathrm{PKE}}(1^\lambda, 1) = 1]|$$

But the CPA security of the underlying PKE scheme implies that this can happen only with a negligible probability. Hence we reach a contradiction to the fact that the success probability of the extractor must be non-negligible. Therefore, we have $\mathrm{Hybd}_0 \approx_\delta \mathrm{Hybd}_1$. This completes the proof of Claim 2.

Hybd_2 This hybrid game is exactly same as Hybd_1 except that we set $c_2 \leftarrow \mathrm{PKE.Enc}(\mathrm{PK}_2, (x, m_1); r_2)$ instead of $c_2 \leftarrow \mathrm{PKE.Enc}(\mathrm{PK}_2, (x, m_0); r_2)$. These two hybrids are computationally indistinguishable by the CPA security of the underlying PKE scheme for the key PK_2 as shown in Claim 3.

Claim 3. *Assuming the* PKE *is a δ-selectively secure public-key encryption,* Hybd_1 *and* Hybd_2 *are δ-indistinguishable.*

Proof. We prove this by contradiction. Let us assume that there exists a polynomial $p(\cdot)$ such that for infinitely many λ

$$|\Pr[\mathrm{Hybd}_1(\lambda) = 1] - \Pr[\mathrm{Hybd}_2(\lambda) = 1]| \geq \tfrac{1}{2} + \tfrac{1}{p(\lambda)}.$$

We construct a PPT adversary $\mathcal{B}$ against the CPA security (Definition 5) of the PKE scheme for the key PK_2 as described in Figure 12 with only change in line 4.2 where we choose a uniformly random y from $\mathcal{Y}$ instead of computing $y = w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2))$.

Note that, in both Hybd_1 and Hybd_2 the value of $w\mathrm{PRF.Eval}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1.\mathrm{PK}_2), \cdot)$ is set as a randomly chosen y from $\mathcal{Y}$. Consequently, $\mathcal{B}$ simulates Hybd_1 if $c'_b \leftarrow \mathrm{PKE.Enc}(\mathrm{PK}_2, (x, m_0); r_2)$, and Hybd_2 if $c'_b \leftarrow \mathrm{PKE.Enc}(\mathrm{PK}_2, (x, m_1); r_2)$. Therefore, the distinguishing advantage of the OWE-adversary $\mathcal{A}$ between Hybd_1 and Hybd_2 is the same as the advantage of the PKE-adversary $\mathcal{B}$ in the CPA security game for the key PK_2. More formally, there exists a polynomial $q(\cdot)$ such that for infinitely many λ, we have

$$\begin{aligned}\tfrac{1}{q(\lambda)} &\leq |\Pr[\mathrm{Hybd}_1(\lambda) = 1] - \Pr[\mathrm{Hybd}_2(\lambda) = 1]| \\ &= |\Pr[\mathrm{Expt}_{\mathcal{B}}^{\mathrm{PKE}}(1^\lambda, 0) = 1] - \Pr[\mathrm{Expt}_{\mathcal{B}}^{\mathrm{PKE}}(1^\lambda, 1) = 1]|\end{aligned}$$

This shows that we arrive at a contradiction to the CPA security of the PKE scheme. Hence it holds that $\mathrm{Hybd}_1 \approx_\delta \mathrm{Hybd}_2$. This completes the proof of Claim 3.

Hybd_3 This hybrid game is the same as the previous game except that we take pp_d as the circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_2, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ instead of setting $pp_d \leftarrow \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$. We show that the adversary $\mathcal{A}$'s distinguishing advantage between Hybd_2 and Hybd_3 is negligible in the following claim.

Claim 4. *Assuming the* RE *is a δ-sub-exponential simulation secure sub-linear compact randomized encoding scheme in* CRS *model for the class of Turing machines $\{\mathcal{M}_\lambda\}$ associated with the class of circuits $\{\mathcal{C}_\lambda\}$,* Hybd_2 *and* Hybd_3 *are δ-indistinguishable.*

Proof. We need to show that the joint distributions $(\widetilde{\Pi}[\overrightarrow{pk}_{i+1}, C_1, z, \alpha^i_{z_i}], \overrightarrow{\mathrm{crs}}_i, \overrightarrow{pk}_i)$ and $(\widetilde{\Pi}[\overrightarrow{pk}_{i+1}, C_2, z, \alpha^i_{z_i}], \overrightarrow{\mathrm{crs}}_i, \overrightarrow{pk}_i)$ for every label $i \in \{0, 1, \ldots, n\}$ and $z \in \{0,1\}^i$, are indistinguishable. It will imply that the two hybrids Hybd_2, Hybd_3 are indistinguishable. If the functionality, runtime and size of two circuits C_1 and C_2 are the same then the above indistinguishability follows from the underlying simulation security of RE scheme in CRS model according to the discussion in Remark 2.

We define an RE-adversary $\mathcal{B}$ against the indistinguishability secure RE scheme in Figure 13. We note RE is δ-indistinguishability secure implies that, if the two ensembles $\{\Pi_1(x_1), |\Pi_1|, |x_1|, T_1 : (\Pi_1, x_1, T_1) \xleftarrow{\$} X_{1,\lambda}\}$ and $\{\Pi_2(x_2), |\Pi_2|, |x_2|, T_2 : (\Pi_2, x_2, T_2) \xleftarrow{\$} X_{2,\lambda}\}$ are δ-indistinguishable then the two distributions $\{\mathrm{RE.Enc}(pk, \Pi_1, x_1)\colon (\Pi_1, x_1, T_1) \xleftarrow{\$} X_{1,\lambda}\}$ and $\{\mathrm{RE.Enc}(pk, \Pi_2, x_2)\colon (\Pi_2, x_2, T_2) \xleftarrow{\$} X_{2,\lambda}\}$ are also δ-indistinguishable, where $\Pi_j \in \mathcal{M}_\lambda$ and T_j denotes the runtime of Π_j on input x_j for $j = 1, 2$.

Therefor, if $pp_d = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ then $\mathcal{B}$ simulates Hybd_2 and if $pp_d = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_2, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ then $\mathcal{B}$ simulates Hybd_3. Now we show the functional equivalence of the circuits C_1 and C_2. Let (c, w) be any arbitrary input to the circuits C_j, $j = 1, 2$ where $c = (c_1, c_2, x, y)$.

Case 1. $(x = \bar{x}, c_1 = \bar{c}_1$ and $c_2 = \bar{c}_2)$: Since $\bar{x} \notin L$, we have $R(x, w) = 0$ in line 4 of C_j (Figure 10), thus C_1 and C_2 both output $\perp$.

Case 2. $(x \neq \bar{x}, c_1 = \bar{c}_1$ and $c_2 = \bar{c}_2)$: Correctness of PKE scheme implies $\mathrm{PKE.Dec}(\mathrm{SK}_j, c_j) = (\bar{x}, \bar{m}_j)$ in line 3 of C_j (Figure 10) and both the circuits returns $\perp$ as $x \neq \bar{x}$ in line 4.

Case 3. $(c_1 \neq \bar{c}_1$ or $c_2 \neq \bar{c}_2)$: If c_1 and c_2 are encryptions of the same message then we have $\mathrm{PKE.Dec}(\mathrm{SK}_1, c_1) = \mathrm{PKE.Dec}(\mathrm{SK}_2, c_2)$. Therefore, the behavior of both circuits

1. The OWE-adversary chooses $(\bar{x}, \bar{m}_0, \bar{m}_1, st) \leftarrow \mathcal{A}(1^\lambda)$ and sends it to the RE-adversary $\mathcal{B}$, where $\bar{x} \notin L$, $|\bar{m}_0| = |\bar{m}_1|$ and st contains some auxiliary information.
2. The RE-adversary $\mathcal{B}$ generates public parameters (pp_e, pp_d) as follows and sends it to $\mathcal{A}$:
   2.1 Generate $(\mathrm{SK}_i, \mathrm{PK}_i) \leftarrow$ PKE.Gen(1^λ) for $i = 1, 2$ and $(\mathsf{fk}, \mathsf{ek}) \leftarrow w$PRF.Gen$(1^\lambda, R')$ where relation R' is the same as in Construction 2
   2.2 Set $pp_e = (\mathrm{PK}_1, \mathrm{PK}_2, \mathsf{ek})$
   2.3 Construct the message output circuits $C_j = \mathsf{MOC}[\mathrm{SK}_j, \mathsf{fk}]$ for $j = 1, 2$ (see Figure 10)
   2.4 Generate $(\mathrm{crs}_i, pk_i) \leftarrow$ RE.Setup$(1^\lambda, 1^S, 1^n, 1^T, 1^l)$ for $i \in \{0, 1, \dots, n\}$ where S, n, T, l are the same as in Construction 2 and set $\overrightarrow{\mathrm{crs}} = \{\mathrm{crs}\}_{i=0}^n$ and $\overrightarrow{\mathrm{pk}_i} = \{\mathrm{pk}_i\}_{j=i}^n$
   2.5 Submit the circuits C_j to the RE-challenger for $j = 1, 2$
   2.6 The RE-challenger pick a random $j \xleftarrow{\$} \{1, 2\}$ and sends $\widetilde{\Pi}[\overrightarrow{pk}_1, C_j, \epsilon, \alpha] \leftarrow$ RE.Enc$(pk_0, \Pi[\overrightarrow{pk}_1, C_j, \epsilon, \alpha], \epsilon)$ to $\mathcal{B}$ where $\Pi[\overrightarrow{pk}_1, C_j, \epsilon, \alpha]$ is a Turing machine defined in Remark 2.
   2.7 Construct the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_j, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ as described in Figure 6.
   2.8 Set $pp_d = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$
3. The RE-adversary $\mathcal{B}$ produces a OWE-ciphertext $\bar{c}$ as follows and submits it to the OWE-adversary $\mathcal{A}$:
   3.1 Chose $r_1, r_2 \xleftarrow{\$} \{0, 1\}^{l_{\mathrm{PKE}}(\lambda)}$
   3.2 Compute $\bar{c}_1 \leftarrow$PKE.Enc$(\mathrm{PK}_1, (\bar{x}, \bar{m}_0); r_1)$, $\bar{c}_2 \leftarrow$PKE.Enc$(\mathrm{PK}_2, (\bar{x}, \bar{m}_1); r_2)$
   3.3 Choose $\bar{y} \xleftarrow{\$} \mathcal{Y}$
   3.4 Set $\bar{c} = (\bar{c}_1, \bar{c}_2, \bar{x}, \bar{y})$and send it to the OWE-adversary $\mathcal{A}$
4. Output $b' \leftarrow \mathcal{A}(st, \bar{c}, pp_e, pp_d)$.

Figure 13: The RE-adversary $\mathcal{B}$ simulating Hybd$_3$

C_1 and C_2 are the same as they differ only in line 3. If the decryptions of c_1 and c_2 are not equal then $(c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2) \notin L'$ and by the correctness of wPRF scheme we have $y \neq w$PRF.F$(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2))$. Hence, the circuits C_1 and C_2 return $\perp$ due to line 2 (Figure 10).

This shows that C_1 and C_2 are functionally equivalent. Also, we note that size and time bound for both the circuits are the same. Hence, we have Hybd$_2 \approx_\delta$ Hybd$_3$. This completes the proof of Claim 4.

Hybd$_4$ The only difference of this hybrid from Hybd$_3$ is that we compute $c_1 \leftarrow$PKE.Enc$(\mathrm{PK}_1, (x, m_1); r_1)$ instead of $c_1 \leftarrow$PKE.Enc$(\mathrm{PK}_1, (x, m_0); r_1)$. Therefore, these two hybrids Hybd$_3$ and Hybd$_4$ are computationally indistinguishable by the CPA security of the underlying PKE scheme for the key PK_1 as stated in the following claim.

Claim 5. *Assuming the* PKE *is a δ-selectively secure public-key encryption,* Hybd$_3$ *and* Hybd$_4$ *are δ-indistinguishable.*

The proof is analogous to that of Claim 3.

Hybd$_5$ In this hybrid game we take pp_d as the circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_1, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ instead of $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, C_2, \epsilon, \alpha], \overrightarrow{\mathrm{crs}}]$ as in the standard scheme. Therefore, by the underlying simulation secure RE scheme we have Hybd$_4$*and*Hybd$_5$ are computationally indistinguishable as stated in the following claim.

Claim 6. *Assuming the* RE *is a δ-sub-exponential simulation secure sub-linear compact randomized encoding scheme in* CRS *model for the class of Turing machines $\{\mathcal{M}_\lambda\}$ associated with the class of circuits $\{\mathcal{C}_\lambda\}$,* Hybd$_4$ *and* Hybd$_5$ *are δ-indistinguishable.*

The proof is similar to that of Claim 4.

Hardwired: a PKE secret key SK_j, a wPRF function key fk.
Input: a ciphertext c and a witness $w \in \mathcal{W}$

1. Parse $c = (c_1, c_2, x, y)$
2. **if** $(w\mathrm{PRF.F}(\mathsf{fk}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2)) = y)$ **then**
3. $\quad(\hat{x}, (\hat{f}, \hat{m})) \leftarrow \mathrm{PKE.Dec}(\mathrm{SK}_j, c_j)$
4. $\quad$**if** $((\hat{x} = x) \bigwedge (R(\hat{x}, w) = 1))$ **then**
5. $\quad\quad$**return** $\hat{f}(\hat{m}, w)$
6. $\quad$**end if**
7. **end if**
8. **return** $\perp$

Figure 14: **Modified Message Output Circuit** $F_j = \mathsf{MMOC}[\mathrm{SK}_j, \mathsf{fk}]$, $j = 1, 2$

Hybd$_6$ Here we again reset $y \leftarrow w\mathrm{PRF.Eval}(\mathsf{ek}, (c_1, c_2, \mathrm{PK}_1, \mathrm{PK}_2), (x, m_1, r_1, r_2))$ instead of randomly chosen $y \in \mathcal{Y}$. Again, by a similar argument as in the Hybd_1, we have that the distinguishing advantage of the adversary $\mathcal{A}$ between the hybrid games Hybd_5 and Hybd_6 is negligible. We state this in the following claim.

Claim 7. *Assuming the* PKE *is a δ-selectively secure public-key encryption and the* wPRF *is extractable,* Hybd_5 *and* Hybd_6 *are δ-indistinguishable.*

The proof is analogous to that of Claim 2.

Observe that Hybd_6 is the experiment $\mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, 1)$. The indistinguishability between the above hybrid games implies that $\mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, 0) \approx_\delta \mathrm{Expt}_{\mathcal{A}}^{\mathrm{OWE}}(1^\lambda, 1)$ and the distinguishing advantage for the adversary $\mathcal{A}$ is strictly less than $\mu(\lambda)$, μ is a negligible function of λ. This completes the proof. □

Remark 3. We convert our OWE scheme into an offline functional witness encryption (OFWE) scheme for a class of functions $\{f_\lambda\}_{\lambda\in\mathbb{N}}$. The encryption algorithm of OFWE is the same as our OWE except that it takes an additional input a function $f \in f_\lambda$ and then encrypts the pair of the function f and a message m with the statement x using the PKE encryption to produce ciphertexts $c_i \leftarrow \mathrm{PKE.Enc}(\mathrm{PK}_i, (x, (f, m)); r_i)$ for $i = 1, 2$. In line 3 of the circuit C_j (Figure 10), we will have $\mathrm{PKE.Dec}(\mathrm{SK}_j, c_j) = (\hat{x}, (\hat{f}, \hat{m}))$ and in line 5 it will return $\hat{f}(\hat{m}, w)$ instead of $\hat{m}$ (see circuit F_j in Figure 14). Rest of the algorithms of OFWE.Setup and OFWE.Dec will be the same as that of our OWE scheme. The time of encryption of the OWEF is bounded by $\mathrm{poly}(\lambda, |x|+|m|+|f|)$ where $|x|, |m|, |f|$ are the size of x, m, f respectively. The correctness and the security of the OFWE depend on the same assumptions as in the case of our OWE.

5 Our Multi-Relation witness PRF

Construction 3. Let pPRF = (Gen, Eval, Punc) be a puncturable pseudorandom function with domain $\{0,1\}^k$, range $\mathcal{Y}$ and RE = (Setup, Enc, Eval) be a bounded input δ-sub-exponential simulation secure sub-linear compact randomized encoding scheme in CRS model for Turing machines. Our mwPRF = (Gen, F, KeyGen, Eval) for an NP language L with a set of relations $\mathcal{R} = \{R : |R| \leq s,\ R : \chi \times \mathcal{W} \to \{0,1\}, \mathcal{X} = \{0,1\}^k$ and $\mathcal{W} = \{0,1\}^{n-k}\}$, is given by the following algorithms.

Hardwired: a pPRF key K and a relation R.
Input: an instance $x \in \mathcal{X} = \{0,1\}^k$ and a witness $w \in \mathcal{W} = \{0,1\}^{n-k}$.
Padding: the circuit is padded to size pad $= \text{pad}_{E_R}(s, d, n, \lambda)$, determined in the analysis.

1. **if** $(R(x, w) = 1)$ **then**
2. $\quad y \leftarrow$ pPRF.Eval(K, x)
3. **else** $y \leftarrow \perp$
4. **end if**
5. **return** y

Figure 15: **Evaluation Circuit** $E_R = \mathsf{EC}[K, R]$

- $\mathsf{fk} \leftarrow mw\text{PRF.Gen}(1^\lambda, s)$: This is run by a user with input a security parameter λ and a bound s on the size of the relations in $\mathcal{R}$.
  - Choose a pPRF key $K \leftarrow \text{pPRF.Gen}(1^\lambda)$, $K \in \{0,1\}^\lambda$.
  - Set and output $\mathsf{fk} = K$ as the secret function key. The user keeps fk as secret.
- $y \leftarrow mw\text{PRF.F}(\mathsf{fk}, x)$: This algorithm generates a PRF value $y \leftarrow \text{pPRF.Eval}(K, x)$ by taking input a secret function key $\mathsf{fk} = K$ and an instance $x \in \mathcal{X}$. The value $y \in \mathcal{Y}$ is treated as the mwPRF value corresponding to the statement $x \in \mathcal{X}$.
- $\mathsf{ek}_R \leftarrow mw\text{PRF.KeyGen}(\mathsf{fk}, R)$: This algorithm is used to compute a evaluation key for any given relation $R \in \mathcal{R}$. It works as follows on input a secret function key fk and a relation R:
  - Construct the circuit $E_R \in \{E_\lambda\}$ as described in Figure 15[4]. Let the circuit E_R be of size S with input size n, output size l and runtime bound T.
  - Generate $(\text{crs}_i, pk_i) \leftarrow \text{RE.Setup}(1^\lambda, 1^S, 1^n, 1^T, 1^l)$ for $i \in \{0, 1, \ldots, n\}$ where crs_i is a common reference string and pk_i is an encoding key. Set $\overrightarrow{\text{crs}} = \{\text{crs}\}_{i=0}^n$ and $\overrightarrow{\text{pk}}_i = \{\text{pk}_j\}_{j=i}^n$.
  - Compute the randomized encoding $\widetilde{\Pi}[\overrightarrow{pk}_1, E_R, \epsilon, \alpha] \leftarrow \text{RE.Enc}(pk_0, \Pi[\overrightarrow{pk}_1, E_R, \epsilon, \alpha], \epsilon)$ where ϵ is a null string, α is a random binary string and $\Pi[\overrightarrow{pk}_1, E_R, \epsilon, \alpha]$ is a Turing machine defined in Remark 2.
  - Build the special circuit $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E_R, \epsilon, \alpha], \overrightarrow{\text{crs}}]$ as described in Figure 6 and output $\mathsf{ek}_R = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E_R, \epsilon, \alpha], \overrightarrow{\text{crs}}]$.
- $mw\text{PRF.Eval}(\mathsf{ek}_R, x, w)$: An entity having a witness $w \in \mathcal{W}$ corresponding to an instance $x \in \mathcal{X}$, runs this algorithm using an evaluation key $\mathsf{ek}_R = \mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E_R, \epsilon, \alpha], \overrightarrow{\text{crs}}]$ and outputs $\mathcal{G}[\widetilde{\Pi}[\overrightarrow{pk}_1, E_R, \epsilon, \alpha], \overrightarrow{\text{crs}}](z)$ where $z = (x, w)$.

Correctness. We note that our $mw\text{PRF.F}(\mathsf{fk}, x)$ is a pPRF evaluation on $x \in \mathcal{X}$ and one can only use mwPRF.Eval with an evaluation key ek_R if he has a valid witness w for x such that $R(x, w) = 1$ as the circuit E_R is hardcoded with the relation circuit R. The correctness of this scheme follows from a similar argument discussed in the correctness of Construction 1.

Padding Parameter. We take $\text{pad}_{E_R}(s, d, n, \lambda) \leq \text{poly}(\lambda, k, s)$ due to a similar argument as in the case of our single relation witness PRF in Construction 1.

[4]The only difference from the circuit E (Figure 7) is that, E_R is now hardcoded with a relation R that can vary with evaluation key ek_R.

Efficiency. The efficiency of our multi-relation witness PRF is the same as that of our single relation witness PRF in Construction 1.

Theorem 11. *Assuming* LWE *with sub-exponential hardness and the existence of δ-sub-exponentially secure one-way functions, if there exists a weakly sub-linear compact public key functional encryption scheme for* $\mathsf{P/poly}$ *with δ-sub-exponential security, then there exists a δ-secure multi-relation witness* PRF *scheme.*

We skip the proof of this theorem as it is similar to the proof of Theorem 9.

6 Conclusion

We constructed a witness PRF from a pPRF and a sub-exponentially secure sub-linear compact RE scheme in CRS model. More precisely, a sub-exponentially secure sub-linear compact RE scheme can be obtained assuming sub-exponential hardness of LWE and existence of sub-exponentially secure one-way functions, sub-exponentially secure weakly sub-linear compact PKFE [LPST16b]. The existing construction of witness PRF [Zha16] is based on multi-linear maps and the security is proved in (unreliable) generic multi-linear group model. We also showed that our single relation witness PRF can be immediately converted into a multi-relation witness PRF where the security depends upon the same assumptions as in the case of our single relation witness PRF.

We use any extractable witness PRF to build an offline witness encryption (OWE) from a public-key encryption and a sub-exponentially secure sub-linear compact RE scheme in CRS model. Our OWE is inspired by the existing construction of OWE [AFP16]. We converted our OWE into a offline functional witness encryption with the same modification shown in [AFP16]. However, we unable to produce a polynomial time extractor for our witness PRF. It will be interesting to build such extractor so that extractable witness PRFs can be integrated in various cryptographic constructions.

Our witness PRF and OWE are both works with a statement-witness pair whose lengths are assumed to be bounded. As discussed in Remark 2, we have used the circuit $\mathcal{G}[\cdot]$ as an obfuscator for bounded input circuits (or Turing machines). Lin et al. [LPST16b] showed that compact RE for certain "*special purpose*" distributions can be used to obfuscate unbounded input Turing machines. Therefore we can use such compact RE for certain "*special purpose*" distributions to get witness PRF or OWE for unbounded length statement-witness pair. We note that compact RE for general distributions does not exist in the plain model [LPST16b]. It would be desirable to construct witness PRFs or OWEs that support unbounded length statement-witness pair.

References

[ABSV14] Prabhanjan Ananth, Zvika Brakerski, Gil Segev, and Vinod Vaikuntanathan. The trojan method in functional encryption: From selective to adaptive security, generically. *IACR Cryptology ePrint Archive*, 2014:917, 2014.

[AFP16] Hamza Abusalah, Georg Fuchsbauer, and Krzysztof Pietrzak. Offline witness encryption. In *International Conference on Applied Cryptography and Network Security*, pages 285–303. Springer, 2016.

[AH14] Seiko Arita and Sari Handa. Two applications of multilinear maps: group key exchange and witness encryption. In *Proceedings of the 2nd ACM workshop on ASIA public-key cryptography*, pages 13–22. ACM, 2014.

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In *Annual Cryptology Conference*, pages 308–326. Springer, 2015.

[AJN+16] Prabhanjan Ananth, Aayush Jain, Moni Naor, Amit Sahai, and Eylon Yogev. Universal constructions and robust combiners for indistinguishability obfuscation and witness encryption. In *Annual Cryptology Conference*, pages 491–520. Springer, 2016.

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability (aka differing-inputs) obfuscation. TCC, 2014.

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. Cryptology ePrint Archive, Report 2001/069, 2001.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *International Workshop on Public Key Cryptography*, pages 501–519. Springer, 2014.

[BH15] Mihir Bellare and Viet Tung Hoang. Adaptive witness encryption and asymmetric password-based cryptography. In *IACR International Workshop on Public Key Cryptography*, pages 308–331. Springer, 2015.

[BKS15] Zvika Brakerski, Ilan Komargodski, and Gil Segev. Multi-input functional encryption in the private-key setting: Stronger security from weaker assumptions. Cryptology ePrint Archive, Report 2015/158, 2015. https://eprint.iacr.org/2015/158.

[BNPW16] Nir Bitansky, Ryo Nishimaki, Alain Passelègue, and Daniel Wichs. From cryptomania to obfustopia through secret-key functional encryption. In *Theory of Cryptography Conference*, pages 391–418. Springer, 2016.

[BS03] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324(1):71–90, 2003.

[BST14] Mihir Bellare, Igors Stepanovs, and Stefano Tessaro. Poly-many hardcore bits for any one-way function and a framework for differing-inputs obfuscation. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 102–121. Springer, 2014.

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 171–190. IEEE, 2015.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 280–300. Springer, 2013.

[BWZ14a] Dan Boneh, David J Wu, and Joe Zimmerman. Immunizing multilinear maps against zeroizing attacks. *IACR Cryptology ePrint Archive*, 2014:930, 2014.

[BWZ14b] Dan Boneh, David J. Wu, and Joe Zimmerman. Immunizing multilinear maps against zeroizing attacks. Cryptology ePrint Archive, Report 2014/930, 2014. `https://eprint.iacr.org/2014/930`.

[BZ17] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. *Algorithmica*, 79(4):1233–1285, 2017.

[CHL+14] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehle. Cryptanalysis on the multilinear map over the integers and its related problems. Cryptology ePrint Archive, Report 2014/906, 2014. `https://eprint.iacr.org/2014/906`.

[CHL+15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 3–12. Springer, 2015.

[CLT13] Jean-Sébastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *Advances in Cryptology–CRYPTO 2013*, pages 476–493. Springer, 2013.

[CLT14a] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. *IACR Cryptology ePrint Archive*, 2014:975, 2014.

[CLT14b] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/975, 2014. `https://eprint.iacr.org/2014/975`.

[DS] David Derler and Daniel Slamanig. Practical witness encryption for algebraic languages or how to encrypt under groth-sahai proofs.

[GGH13] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–17. Springer, 2013.

[GGH$^+$16] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM Journal on Computing*, 45(3):882–929, 2016.

[GGHW17] Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. *Algorithmica*, 79(4):1353–1373, 2017.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33:792–807, 1986.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 467–476. ACM, 2013.

[GHMS14a] Craig Gentry, Shai Halevi, Hemanta K Maji, and Amit Sahai. Zeroizing without zeroes: Cryptanalyzing multilinear maps without encodings of zero. *IACR Cryptology ePrint Archive*, 2014:929, 2014.

[GHMS14b] Craig Gentry, Shai Halevi, Hemanta K. Maji, and Amit Sahai. Zeroizing without zeroes: Cryptanalyzing multilinear maps without encodings of zero. Cryptology ePrint Archive, Report 2014/929, 2014. `https://eprint.iacr.org/2014/929`.

[GKP$^+$13a] Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 555–564. ACM, 2013.

[GKP$^+$13b] Shafi Goldwasser, Yael Tauman Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. How to run turing machines on encrypted data. In *Advances in Cryptology–CRYPTO 2013*, pages 536–553. Springer, 2013.

[GLW14] Craig Gentry, Allison Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In *International Cryptology Conference*, pages 426–443. Springer, 2014.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 415–432. Springer, 2008.

[IK00] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 294–304. IEEE, 2000.

[KNT17] Fuyuki Kitagawa, Ryo Nishimaki, and Keisuke Tanaka. Indistinguishability obfuscation for all circuits from secret-key functional encryption. *IACR Cryptology ePrint Archive*, 2017:361, 2017.

[KNY17] Ilan Komargodski, Moni Naor, and Eylon Yogev. Secret-sharing for np. *Journal of Cryptology*, 30(2):444–469, 2017.

[Lev87] Leonid A Levin. One way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, 1987.

[LKW15] Jia Liu, Saqib A Kakvi, and Bogdan Warinschi. Extractable witness encryption and timed-release encryption from bitcoin. Technical report, Cryptology ePrint Archive, Report 2015/482, 2015. http://eprint. iacr. org/2015/482, 2015.

[LPST16a] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation with non-trivial efficiency. In *IACR International Workshop on Public Key Cryptography*, pages 447–462. Springer, 2016.

[LPST16b] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Output-compressing randomized encodings and applications. In *Theory of Cryptography Conference*, pages 96–124. Springer, 2016.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 475–484. ACM, 2014.

[Zha16] Mark Zhandry. How to avoid obfuscation using witness prfs. In *Theory of Cryptography Conference*, pages 421–448. Springer, 2016.