New Constructions of Hinting PRGs, OWFs with Encryption, and more

Rishab Goyal[*] Satyanarayana Vusirikala[†] Brent Waters[‡]

Abstract

Over the last few years there has been a surge of new cryptographic results, including laconic oblivious transfer [CDG+17, DGI+19], (anonymous/ hierarchical) identity-based encryption [BLSV17], trapdoor functions [GH18, GGH19], chosen-ciphertext security transformations [KW19, KMT19], designated-verifier zero knowledge proofs [LQR+19, QRW19, KNYY19], due to a beautiful framework recently introduced in the works of Cho et al. [CDG+17], and Döttling and Garg [DG17a]. The primitive of one-way function with encryption (OWFE) [GH18, GGH19] and its relatives (chameleon encryption, one-time signatures with encryption, hinting PRGs, trapdoor hash encryption, batch encryption) [DG17a, DG17b, BLSV17, KW19, DGI+19] have been a centerpiece in all these results.

While there exist multiple realizations of OWFE (and its relatives) from a variety of assumptions such as CDH, Factoring, and LWE, all such constructions fall under the same general "missing block" framework [CDG+17, DG17a]. Although this framework has been instrumental in opening up a new pathway towards various cryptographic functionalities via the abstraction of OWFE (and its relatives), it has been accompanied with undesirable inefficiencies that has inhibited a much wider adoption in many practical scenarios. Motivated by the surging importance of the OWFE abstraction (and its relatives), a natural question to ask is whether the existing approaches can be diversified to not only obtain more constructions from different assumptions, but also in developing newer frameworks. We believe answering this question will eventually lead to important and previously unexplored performance trade-offs in the overarching applications of this novel cryptographic paradigm.

In this work, we propose a new *accumulation-style* framework for building a new class of OWFE as well as hinting PRG constructions with a special focus on achieving shorter ciphertext size and shorter public parameter size (respectively). Such performance improvements parlay into shorter parameters in their corresponding applications. Briefly, we explore the following performance trade-offs — (1) for OWFE, our constructions outperform in terms of ciphertext size as well as encryption time, but this comes at the cost of larger evaluation and setup times, (2) for hinting PRGs, our constructions provide a rather dramatic trade-off between evaluation time versus parameter size, with our construction leading to significantly shorter public parameter size. We also provide concrete performance measurements for our constructions and compare them with existing approaches. We believe highlighting such trade-offs will lead to a wider adoption of these abstractions in a practical sense.

1 Introduction

A major goal in cryptography is to study cryptographic primitives that could be used for securely implementing useful functionalities as well as lead to interesting applications. Significant effort in cryptographic research is geared towards diversifying existing frameworks and constructions for realizing such primitives with the goal of improving efficiency as well as obtaining more constructions from a wider set of well-studied assumptions. Over the last few years there has been a surge of new constructions [DG17a, DG17b, GS17, BLSV17, GH18, GS18, GOS18, DGHM18, GHMR18, KW19, GGH19, KMT19, LQR+19, QRW19, KNYY19, DGI+19,

[*]University of Texas at Austin. Email: rgoyal@cs.utexas.edu. Supported by IBM PhD Fellowship.

[†]University of Texas at Austin. Email: satya@cs.utexas.edu.

[‡]University of Texas at Austin and NTT Research. Email: bwaters@cs.utexas.edu. Supported by NSF CNS-1908611, CNS-1414082, DARPA SafeWare, Packard Foundation Fellowship, and Simons Investigator Award.

AMPR19, AMP19, GHM+19] due to a beautiful framework recently introduced in the works of Cho et al. [CDG+17], and Döttling and Garg [DG17a]. This new wave of cryptographic results, including laconic oblivious transfer [CDG+17, DGI+19], (anonymous/ hierarchical) identity-based encryption [BLSV17], trapdoor functions [GH18, GGH19], chosen-ciphertext security transformations [KW19, KMT19], designated-verifier zero knowledge proofs [LQR+19, QRW19, KNYY19], registration-based encryption [GHMR18, GHM+19] has been propelled by the primitive of one-way function with encryption (OWFE) [GH18, GGH19] and its relatives (chameleon encryption, one-time signatures with encryption, hinting PRGs, trapdoor hash encryption, batch encryption) [DG17a, DG17b, BLSV17, KW19, DGI+19] .

A one-way function with encryption scheme extends the fundamental notion of one-way functions in the following way. The scheme is associated with a family of one-way functions $\mathcal{F} = \{f_{\mathsf{pp}}\}_{\mathsf{pp}}$ where during setup one samples public parameters pp that fixes the underlying one-way function $f = f_{\mathsf{pp}}$. The special feature of an OWFE scheme, which makes them such a useful primitive, is the possibility of performing encryption and decryption without sampling additional keys. Formally, in an OWFE scheme, the encryption procedure is abstracted out into two components — algorithms E_1, E_2 which work as follows. Both E_1 and E_2 share the same random coins ρ, and take as inputs a value y (that lies in the image space of f), an index-bit pair (i, b), and parameters pp. Algorithm E_1 is used to compute the "ciphertext" ct, whereas E_2 computes the encrypted KEM key k. The decryption algorithm D on inputs a ciphertext ct, pre-image string x, and parameters pp, outputs a decrypted KEM key k'. For correctness it is important that if the string x is such that $y = f_{\mathsf{pp}}(x)$ and $x_i = b$, then the KEM keys should match, i.e. $k' = k$. While for security, other than unpredictability of the OWF f, it is required that the ciphertext does not leak the KEM key trivially. That is, given an input x, parameters pp, and a ciphertext ct, the associated KEM key k must be indistinguishable from random as long as the encryption is performed for some value $y = f_{\mathsf{pp}}(x)$ and any index-bit pair of the form $(i, 1 - x_i)$. There are various other properties which have been studied for OWFE schemes such as recyclability and smoothness, since they lead to more applications. For the purposes of this introduction we only focus on the above simpler properties, and more details are provided later.[1]

Intuitively, an OWFE scheme is simply a one-way function f equipped with matching encryption-decryption procedures such that encryption allows to encrypt messages with respect to an OWF output string y and a pre-image bit (i, b), while decryption requires a pre-image x such that $f(x) = y$ and $x_i = b$. An alternate and possibly more intuitive way to look at OWFE could be through the lens of (non-interactive) 2-party key-exchange instead of encryption. Consider an asymmetric setting in which the parties in a key-exchange are "typed", i.e. Type I and II parties where key-exchange is only possible between parties of different types. It turns out an OWFE inherently gives such a key-exchange protocol in which key-exchange leads to static/permanent keys for Type I parties, and the Type II parties can *conditionally* recover the shared keys. Concretely, a Type I party sets its public key as a "ciphertext" ct which is generated using the E_1 algorithm for some value y and pre-image bit (i, b). The static private key for any such Type I user simply corresponds to the associated KEM key (computable using E_2). And, a Type II user can recover the same private key if it possesses a pre-image $x \in f^{-1}(y)$ such that $x_i = b$. At a high level, looking through the lens of key-exchange, encryption algorithm E_2 and decryption algorithm D can be interpreted as the "shared key" generation algorithms for Type I and II parties, respectively.

The "Missing Block" Framework. While there exist multiple realizations of OWFE (and its relatives) from a variety of assumptions such as CDH, Factoring, and LWE, all such constructions fall under the same general "missing block" framework [CDG+17, DG17a]. To illustrate the aforementioned framework we sketch the CDH-based OWFE construction provided by Garg and Hajiabadi [GH18]. The public parameters consists of $2n$ randomly sampled group generators $\{g_{i,b}\}_{(i,b)\in[n]\times\{0,1\}}$, where n is the input length of the OWF. The function output is computed by performing subset-product on the public parameters, where the subset selection is done as per the input bits. Concretely, on an input $x \in \{0,1\}^n$, the output is $f(x) = \prod_i g_{i,x_i}$. The ciphertext structurally looks like the public parameters, that is it also consists of

[1] Briefly, the recyclability property says that the E_1 algorithm does not depend on the value y, while smoothness property says that given the output $y = f(x)$ of the OWF, the corresponding pre-image x is unpredictable as long as x is drawn from a distribution with sufficient min-entropy.

$2n$ group elements $\{c_{i,b}\}_{i,b}$. Here to encrypt to pre-image bit (i^*, b^*) under randomness ρ, the encryption algorithm E_1 simply sets $c_{i,b} = g_{i,b}^{\rho}$ for all $(i, b) \neq (i^*, 1 - b^*)$, with the $(i^*, 1 - b^*)^{th}$ term not being set (i.e., $c_{i^*,1-b^*} = \perp$). Pictorially, this can be represented as follows (where $i^* = 2$ and $b^* = 0$):

$$\mathsf{pp} = \begin{array}{|c|c|c|c|c|}\hline g_{1,0} & g_{2,0} & g_{3,0} & \cdots & g_{n,0} \\ \hline g_{1,1} & g_{2,1} & g_{3,1} & \cdots & g_{n,1} \\ \hline \end{array} \xrightarrow[E_1(\mathsf{pp},(2,0);\rho)]{\text{Encryption}} \mathsf{ct} = \begin{array}{|c|c|c|c|c|}\hline g_{1,0}^{\rho} & g_{2,0}^{\rho} & g_{3,0}^{\rho} & \cdots & g_{n,0}^{\rho} \\ \hline g_{1,1}^{\rho} & \times & g_{3,1}^{\rho} & \cdots & g_{n,1}^{\rho} \\ \hline \end{array}$$

The KEM key is simply computed by the encryptor as $\mathsf{HC}(y^{\rho})$, where y is the output of the OWF and HC corresponds to the hardcore predicate. The decryptor on the other does not know the randomness ρ, thus given the ciphertext ct and a valid pre-image x, it computes the subset-product on ct (followed by applying the hardcore predicate), where the subset selection is done as per x. That is, decryptor computes the key as $\mathsf{HC}(\prod_i c_{i,x_i})$.

Basically this notion of not setting up the $(i^*, 1 - b^*)^{th}$ term in the ciphertext is what we refer to as adding a "missing block". The intuition behind this is that the ciphertext should only be decryptable using pre-images x such that $x_{i^*} = b^*$, thus the ciphertext component corresponding to the pre-image bit $(i^*, 1-b^*)$ can be omitted. Here the omission of the $(i^*, 1-b^*)^{th}$ block is very crucial in proving security of encryption.

Limitations of the framework. Although the "missing block" framework has been instrumental in opening up a new pathway towards various cryptographic functionalities via the abstraction of OWFE (and its relatives), it has been accompanied with undesirable inefficiencies that has lead to large system parameters in most of the applications. In particular, the OWFE described above in this framework leads to large "ciphertexts" where the size grows linearly with the input length n of the OWF. Now this inefficiency gets amplified in a different way in each of its application. For instance, large OWFE ciphertexts lead to large public parameters of a trapdoor function (/deterministic encryption) [GH18, GGH19], since the public parameters as per those transformations consists of a polynomial number of OWFE ciphertexts which themselves grow linearly with n. Similar situations arise if we look at the Hinting PRG abstraction [KW19], where the existing constructions via the "missing block" framework lead to much worse public parameters, and the performance overhead gets significantly amplified if we look at its application to chosen-ciphertext security transformations [KW19].

Motivated by the surging importance of the abstraction of one-way function with encryption and its relatives, a natural question to ask is whether the existing approaches can be diversified to not only obtain more constructions from different assumptions, but also in developing newer frameworks. We believe answering this question will eventually lead to important and previously unexplored performance trade-offs in the overarching applications of this novel cryptographic paradigm.

1.1 Our Approach

In this work, we develop a new *accumulation-style* framework for building a new class of one-way function with encryption (as well as hinting PRG) constructions with a special focus on achieving shorter ciphertext size (and shorter public parameter size, respectively), which will parlay into shorter parameters in their corresponding applications. Concretely we explore the following performance trade-offs. For OWFE, our constructions based on this new framework outperform the existing ones in terms of ciphertext size as well as encryption time, but this comes at the cost of larger evaluation and setup times. In terms of applications of OWFE to deterministic encryption, this trade-off translates to a scheme with much smaller public parameters and setup time, but larger encryption/decryption times. For hinting PRGs, our constructions provide a rather dramatic trade-off between evaluation time versus parameter size compared to prior schemes, with our construction leading to significantly shorter public parameter size. In terms of applications of hinting PRG to chosen-ciphertext security transformations, the trade-off between public parameter size and evaluation time in the hinting PRG constructions carries forward to a trade-off between encryption key/ciphertext sizes and encryption/decryption times in the resultant CCA-secure construction. Next, we describe the main ideas behind our constructions, and later we give some concrete performance metrics.

OWF with Encryption from Φ-Hiding. The main inspiration behind our new framework are the number theoretic-based accumulators [BdM93, BP97, STY00, CL02, GR04, Ngu05, CKS09, ATSM09, CF13] that have been widely studied in the literature. For building an OWFE scheme from RSA-family assumptions, we look back at the RSA-based accumulators [BdM93, BP97, STY00, CL02, GR04] which are the earliest number theoretic instantiations available. Briefly our idea can be interpreted as follows. During setup, we sample n pairs of large primes $\{e_{i,b}\}_{i,b}$ and a random generator g in the base group of the RSA modulus. The one-way function associated with those parameters will simply correspond to an accumulation of half of these primes, depending upon the input x, with respect to generator g. Now a ciphertext for a pre-image bit (i, b) will look like an accumulation of only the $(i, b)^{th}$ prime, i.e. $e_{i,b}$. Now during decryption the decryptor simply accumulates rest of the primes as per remaining pre-image bits. Below we sketch our construction in more detail.

The public parameters pp consist of an RSA modulus N, n pairs of λ-bit primes $\{e_{i,b}\}_{(i,b)\in[n]\times\{0,1\}}$, a generator $g \in \mathbb{Z}_N^*$, and a pairwise independent hash H. (Here n is the input length.) Given an input $x \in \{0,1\}^n$, the one-way function $f_{\mathsf{pp}}(x)$ is computed as $g^{H(x)\cdot\prod_i e_{i,x_i}} (\bmod N)$. The encryption algorithm E_1 on input a pre-image bit (i^*, b^*) and randomness ρ, outputs ciphertext as $\mathsf{ct} = g^{\rho\cdot e_{i^*,b^*}} (\bmod N)$.[2] The corresponding KEM key is set as $k = y^\rho (\bmod N)$, where y is the output of the OWF. Lastly, the decryption procedure given a ciphertext ct and a pre-image x such that $f_{\mathsf{pp}}(x) = y$ and $x_{i^*} = b^*$, computes the key as $k' = \mathsf{ct}^{\prod_{i\neq i^*} e_{i,x_i}} (\bmod N)$. Next, we briefly sketch the main arguments behind the security of this construction.

The one-wayness argument proceeds as follows — suppose an adversary finds a collision $x \neq x'$, i.e. $f_{\mathsf{pp}}(x) = f_{\mathsf{pp}}(x')$, then a reduction algorithm can sample the λ-bit primes in such a way that, as long as n is larger than $\log N + \lambda$, it can break RSA assumption for one of the primes sampled as part of the public parameters. For proving security of encryption we need to slightly modify the construction wherein we need to apply an extractor on the KEM key to prove it looks indistinguishable from random, that is $k = \mathsf{Ext}(\mathsf{sd}, y^\rho)$ where Ext is a strong seeded extractor and seed $\mathfrak{s}$ is sampled during setup. Recall that security of encryption requires that for any index-bit pair (i^*, b^*) and input x such that $x_{i^*} \neq b^*$, given a ciphertext $\mathsf{ct} = E_1(\mathsf{pp}, (i^*, b^*); \rho)$ the associated KEM key $k = E_2(\mathsf{pp}, f_{\mathsf{pp}}(x), (i^*, b^*); \rho)$ must be indistinguishable from random. The idea behind proving the same for the above construction is the following — ciphertext looks like $\mathsf{ct} = g^{\rho\cdot e_{i^*,b^*}}$ whereas the key is computed as $k = \mathsf{Ext}(\mathsf{sd}, g^{\rho\cdot\prod_i e_{i,x_i}})$. Since $b^* \neq x_{i^*}$, thus the key can be re-written as $k = \mathsf{Ext}(\mathsf{sd}, (\mathsf{ct}^{\prod_i e_{i,x_i}})^{e^{-1}_{i^*,b^*}})$. Now under the Φ-hiding assumption, we can argue that an adversary can not distinguish between the cases where e_{i^*,b^*} is co-prime with respect to $\phi(N)$, and when e_{i^*,b^*} divides $\phi(N)$. Note that in the latter case, there are e_{i^*,b^*} many distinct $e^{th}_{i^*,b^*}$ roots of $\mathsf{ct}^{\prod_i e_{i,x_i}}$. Thus, by strong extractor guarantee we can conclude that key k looks uniformly random to the adversary as the underlying source has large (λ bits of) min-entropy.

Lastly we show that the above scheme satisfies the smoothness property for appropriate parameters ℓ, n. Formally, the (ℓ, n)-smoothness property says that for any two (ℓ, n)-sources S_0 and S_1, the distributions $\{f_{\mathsf{pp}}(x) : x \leftarrow S_0\}$ and $\{f_{\mathsf{pp}}(x) : x \leftarrow S_1\}$ should be computationally indistinguishable. A natural approach to proving smoothness is to first show that the function $t(x) = H(x) \cdot \prod_i e_{i,x_i} (\bmod \phi(N))$ is a 2-universal hash function, and then apply Leftover Hash Lemma (LHL) to argue that $t(x)$ is statistically close to uniform when x is sampled from an appropriate source S. Note that if we could prove this then smoothness property would follow directly, since we know that the distributions $\{g^{t(x)} : x \leftarrow S_0\}$ and $\{g^{t(x)} : x \leftarrow S_1\}$ will be indistinguishable if $\{t(x) : x \leftarrow S_0\}$ and $\{t(x) : x \leftarrow S_1\}$ are indistinguishable. Although similar strategies have been employed in all prior works, it turns out such an approach does not work in this case. This is because in our construction the exponents $\{e_{i,b}\}$ are sampled as random λ-bit primes instead of being sampled uniformly from $\mathbb{Z}_{\phi(N)}$, thus we can not show $t(\cdot)$ to be a 2-universal hash function. In this work, we devise novel number theoretic techniques which allow us to prove smoothness while getting around the above bottleneck. At a very high level, our approach is to provide a tight LHL-style proof which allows us to argue partial statistical indistinguishability and to complete the argument we rely on computational hardness of Φ-hiding assumption. Our proof is a mixture of a computational as well as statistical argument,

[2] Technically, the ciphertext should also include the index i^* but we drop it for ease of exposition.

and below we describe the structure at a high level.

Let $r_1 \cdot r_2 \cdots r_m$ (for some m) denote the prime factorization of $\phi(N)$, with $r_1 > r_2 > \cdots > r_m$[3]. Now suppose that a PPT adversary $\mathcal{A}$ can distinguish between distributions $\{g^{t(x)} : x \leftarrow S_0\}$ and $\{g^{t(x)} : x \leftarrow S_1\}$ with non-negligible probability ϵ. We show how to use such an adversary to break the Φ-hiding assumption with non-negligible probability. The proof proceeds in two steps, where in the first step we argue that $(t(x) \bmod r_i)$ is statistically close to random over $\mathbb{Z}_{r_i}$ for all prime factors r_i greater than a fixed threshold $\tau_{N,\epsilon}$. By a very tight LHL-style proof, we show that if we set the threshold $\tau_{N,\epsilon}$ appropriately as a certain large enough polynomial in $\log N$ and ϵ^{-1}, then we can show that the following two distributions are at most $\epsilon/2$-far (in terms of statistical distance):

$$\mathcal{D}_S = \{t(x) \pmod{\phi(N)} \ : \ x \leftarrow S\},$$
$$\widetilde{\mathcal{D}}_{S,\kappa_{N,\epsilon}} = \left\{\mathsf{CRT}(U_{\mathbb{Z}_{r_1}}, \ldots, U_{\mathbb{Z}_{r_{\kappa_{N,\epsilon}}}}, t(x) \bmod r_{\kappa_{N,\epsilon}+1}, \ldots, t(x) \bmod r_m) \ : \ x \leftarrow S\right\},$$

where $\kappa_{N,\epsilon}$ is such that $r_{\kappa_{N,\epsilon}}$ is smallest prime larger than $\tau_{N,\epsilon}$ and CRT represents the chinese remainder theorem representation. In the second part of the proof, we show that by relying on Φ-hiding assumption we can argue that that the hash function $t(\cdot)$ can be made lossy on all prime factors of $\phi(N)$ less than or equal to $\tau_{N,\epsilon}$. Combining these two statistical and computational arguments, the smoothness property follows. We point out that throughout this paper we need to use the aforementioned number theoretic techniques (which involve non-trivial applications of the Prime Number Theorem for Arithmetic Progressions [Dir37]) at multiple places, thus we abstract out and prove many useful number theoretic lemmas and theorem separately in Section 3.

Comparing with DDH-based constructions. Comparing the asymptotic efficiency of our Φ-Hiding based OWFE construction with the existing DDH-based constructions, we observe the following: (1) the size of the public parameters grows linearly with the input length n in both constructions, (2) both OWF evaluation and decryption operations require $O(n)$ group operations and $O(n)$ exponentiations (with λ-bit exponents) respectively, (3) for the Φ-hiding based construction, both E_1 and E_2 algorithms perform a single exponentiation, and outputs a ciphertext and key containing just one group element; whereas for DDH-based construction, the E_1 algorithm performs $O(n)$ exponentiations and outputs a ciphertext containing $O(n)$ group elements.

We implemented the above construction and observed that, at 128-bit security level, our Φ-hiding based construction have $\sim$80x shorter ciphertext size over the existing DDH-based construction [GH18]. Also, the E_1 algorithm of our Φ-hiding based construction is $\sim$14x faster than the DDH baseline. A detailed efficiency comparison for other security levels is discussed in Section 8.2.

Hinting PRGs from Φ-Hiding. To show general applicability of our *accumulation-style* framework, we also provide a hinting PRG [KW19] construction based on Φ-hiding that leads to similar performance trade-offs. Let us briefly recall the notion of hinting PRGs. It consists of two algorithms — Setup and Eval, where the setup algorithm generates the public parameters pp, and the PRG evaluation algorithm takes as input the parameters pp, a seed $s \in \{0,1\}^n$ and a block index $i \in \{0, 1, \ldots, n\}$. The security requirement from hinting PRGs is different from standard PRG indistinguishability wherein the hinting PRG scheme is secure if for a randomly choosen seed $s \in \{0,1\}^n$, the following two distributions over $\{r_{i,b}\}_{(i,b)\in[n]\times\{0,1\}}$ are indistinguishable: in the first distribution, $r_{i,s_i} = \mathsf{Eval}(\mathsf{pp}, s, i)$ and $r_{i,1-s_i}$ is sampled uniformly at random for every i; whereas in the second distribution, all $r_{i,b}$ terms are sampled uniformly at random.

Our hinting PRG construction is based on our OWFE construction, where the setup algorithm is identical, that is the public parameters pp consist of an RSA modulus N, n pairs of λ-bit primes $\{e_{i,b}\}_{(i,b)\in[n]\times\{0,1\}}$, a generator $g \in \mathbb{Z}_N^*$, and a pairwise independent hash H. And, the evaluation algorithm also bears strong resemblance with the one-way function f described previously. Concretely, the i^{*th} block of the PRG output, i.e. $\mathsf{Eval}(\mathsf{pp}, s, i^*)$, is computed as $g^{H(s)\cdot\prod_{i\neq i^*} e_{i,s_i}} \pmod N$. Proving security of this construction uses ideas

[3]For ease of exposition, we assume that all the prime factors $\{r_i\}_i$ are distinct. In the main body, we do not make this restriction and present a more general proof.

similar to those used for proving smoothness property of our OWFE construction. More details on this are provided later in Section 4.

Comparing the asymptotic efficiency of our Φ-Hiding based hinting PRG construction with the existing DDH-based constructions, we observe the following: (1) the public parameters consists of $2n$ (λ-bit) prime exponents along with the RSA modulus, extractor seed, group generator, and a hash key; whereas in the DDH-based constructions, it contains $O(n^2)$ group elements, (2) for evaluating a single hinting PRG block, the evaluator needs to perform $O(n)$ exponentiations in our new construction; whereas in the DDH case it performs $O(n)$ group operations. Additionally, using an elegant Dynamic Programming style algorithm (described in Section 4.1), we can reduce the number of exponentiation operations needed per block to grow only logarithmically in n. The intuition behind such an improvement is that we show how to re-use various intermediate exponentiations obtained during a single hinting PRG block evaluation for accelerating the PRG evaluation for other blocks.

Limitations of Φ-Hiding based constructions. A quick glance at the above constructions give us that these new constructions lead to much shorter ciphertext size (in the case of OWFE) and public parameters (in the case of hinting PRGs), therefore they will lead to better parameters in their corresponding applications such as deterministic encryption [GGH19] and chosen-ciphertext security transformations [KW19]. However, looking more closely we observe that our Φ-hiding based construction has an undesirable consequence which is the hinting PRG seed length (or equivalently input length for OWF) n is much larger for our Φ-hiding based scheme when compared with its DDH counterpart. This is due to the fact because of number field sieve attacks, the recommended RSA modulus length (and thereby the input/seed length n) increases super linearly with target security level for the Φ-based construction. While the recommended field size (and thereby the input/seed length n) will increase only linearly for the elliptic curve DDH-based constructions.

Luckily, the notion of accumulators has been well studied in prime order group setting [Ngu05, CKS09, ATSM09, CF13] as well, thus this gives us a different type of number theoretic accumulator. Pivoting to such accumulators, we show how to achieve performance improvements similar to that in the Φ-hiding setting while keeping the input/seed length n close to that in their existing counterparts. Next, we provide our OWFE construction which uses bilinear maps in the prime order group setting.

OWF with Encryption from DBDHI. Let us start by recalling the Decisional Bilinear Diffie-Hellman Inversion (DBDHI) assumption [BB04]. The strength of the assumption is characterized by a parameter ℓ, and it states that given a sequence of group elements as follows — $(g, g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^\ell})$, where g is a random group generator and α is a randomly chosen non-zero exponent, no PPT adversary should be able to distinguish $e(g, g)^{1/\alpha}$ from a random element in the target group. Below we describe our OWFE construction in which we directly include the sequence of elements as described above as part of the public parameters.

Concretely, the public parameters pp consist of $n+1$ group elements $(g, g^{\alpha}, \ldots, g^{\alpha^n})$ for a random exponent α and group generator g, and a pairwise independent hash H. (Here n is the input length.) Given an input $x \in \{0,1\}^n$, the one-way function $f_{\mathsf{pp}}(x)$ is computed in two stages. First, the evaluator symbolically evaluates (i.e. simplifies) the polynomial $p(z) = H(x) \cdot \prod_i (z + 2i + x_i)$. Let $p(z) = \sum_{j=0}^{n} c_j z^j$ be the evaluated polynomial. Next, the evaluator sets the output of the OWF as $\prod_j (g^{\alpha^j})^{c_j}$. The encryption algorithm E_1 on input a pre-image bit (i^*, b^*) and randomness ρ, outputs ciphertext as $\mathsf{ct} = (g^{\alpha + 2i^* + b^*})^{\rho}$.[4] The corresponding KEM key is set as $k = e(g^{\rho}, y)$, where y is the output of the OWF. Lastly, the decryption procedure given a ciphertext ct and a pre-image x such that $f_{\mathsf{pp}}(x) = y$ and $x_{i^*} = b^*$, also takes a two step approach where first it symbolically evaluates the polynomial $p'(z) = H(x) \cdot \prod_{i \neq i^*} (z + 2i + x_i)$. Let $p'(z) = \sum_{j=0}^{n-1} c'_j z^j$ be the evaluated polynomial. Lastly, the decryptor computes the key as $k' = e(\mathsf{ct}, \prod_j (g^{\alpha^j})^{c'_j})$.

The proof of one-wayness is similar to that in the case of Φ-hiding where if an adversary finds a collision $x \neq x'$, i.e. $f_{\mathsf{pp}}(x) = f_{\mathsf{pp}}(x')$, then a reduction algorithm can set the public parameters appropriately such that, as long as n is large enough, it can be used to not only distinguish the DBDHI challenge but also directly compute the DBDHI challenge. The proof of encryption security is also quite similar, where

[4] Technically, the ciphertext should also include the index i^* but we drop it for ease of exposition.

the main idea can be described as follows: the ciphertext looks like $\mathsf{ct} = (g^{\alpha+2i^*+b^*})^\rho$ whereas the key is computed as $k = e(g^\rho, \prod_j (g^{\alpha^j})^{c_j})$. Whenever $b^* \neq x_{i^*}$, then the key can be re-written such that it is of the form $k = e(g,g)^{c'/\beta} \cdot e(g, \prod_j (g^{\beta^j})^{c'_j})$ for some constants $c', c'_1, \ldots, c'_{n-1}$, and where β linearly depends on α. Now by a careful analysis we can reduce this to the DBDHI assumption. Lastly, the proof of smoothness for this construction is significantly simpler than that of its Φ-hiding based counterpart. This is primarily because in this case we can directly prove that the function $H(x) \cdot \prod_i (\alpha + 2i + x_i) (\mathrm{mod}\ p)$, where p is the order of the group, is an (almost) 2-universal hash function, therefore by applying LHL we can argue smoothness of the OWF. More details are provided later in Section 7.

We implemented the above construction and observed that, at 128-bit security level, our DBDHI-based construction have ∼340x shorter ciphertext size over the existing DDH-based construction [GH18] and ∼4x over our Φ-hiding based construction. Also, the E_1 algorithm of our DBDHI-based construction is ∼300x faster than the DDH baseline and ∼22x faster than our Φ-hiding construction. Note that even though Φ-hiding and DBDHI-based constructions have nearly identical asymptotic complexity, DBDHI-based construction still perform better as the recommended group size for elliptic curve groups is smaller than that for RSA.

Hinting PRGs from DDHI and OWFE without Bilinear Maps. Again to emphasize the general applicability of our *accumulation-style* framework, we provide a hinting PRG construction based on DDHI assumption as well. The translation from OWFE to hinting PRG is done analogous to that for Φ-hiding based constructions, except in our hinting PRG construction we do not require the bilinear map functionality. Briefly, this is because (unlike OWFE schemes) hinting PRGs do not provide any decryption-like functionality, and for evaluating the hinting PRG, standard group operations are sufficient. Our construction is described in detail later in Section 5. We also point out that in Appendix B we provide a OWFE construction in the prime order group setting without using bilinear maps, but the caveat is that it does not lead to better performance when compared with existing DDH-based constructions.

We implemented the above schemes and observed that, at 128 bit security level, the setup algorithm of our Φ-hiding and DDHI-based HPRGs are ∼1.35x and ∼200x respectively faster than the DDH baseline [KW19]. Our constructions also have ∼ 105x and ∼2100x shorter public parameters respectively than DDH baseline. However, our schemes have less efficient Eval algorithm, and thereby offer a noticeable trade-off between efficiency of Setup and Enc algorithm when used in chosen-ciphertext security transformation of [KW19]. More details are provided later in Section 8.1.

Roadmap. We recall the notions of Hinting PRG and OWFE in Section 2. We then present number-theoretic techniques introduced in this work in Section 3. We describe our HPRG constructions based on Φ-hiding and DDHI assumptions in Sections 4 and 5. We then present our OWFE constructions based on Φ-hiding, DBDHI and DDHI assumptions in Sections 6 and 7 and Appendix B. In Appendix A, we describe how to construct Hinting PRG generically from OWFE. Finally, we implement our schemes and analyze their performance in Section 8.

2 Preliminaries

Notations. Let PPT denote probabilistic polynomial-time. We denote the set of all positive integers upto n as $[n] := \{1, \ldots, n\}$. Throughout this paper, unless specified, all polynomials we consider are positive polynomials. For any finite set S, $x \leftarrow S$ denotes a uniformly random element x from the set S. Similarly, for any distribution $\mathcal{D}$, $x \leftarrow \mathcal{D}$ denotes an element x drawn from distribution $\mathcal{D}$. The distribution $\mathcal{D}^n$ is used to represent a distribution over vectors of n components, where each component is drawn independently from the distribution $\mathcal{D}$. We call any distribution on n-length bit strings with minimum entropy k as a (k, n) source.

2.1 One Way Function with Encryption

Here we recall the definition of recyclable one-way function with encryption from [GH18, GGH19]. We adapt the definiton to a setting where the KEM key is an ℓ-bit string instead of just a single bit. A recyclable (k, n, ℓ)-OWFE scheme consists of the PPT algorithms K, f, E_1, E_2 and D with the following syntax.

$K(1^\lambda) \to \mathsf{pp}$: Takes the security parameter 1^λ and outputs public parameters pp.

$f(\mathsf{pp}, x) \to y$: Takes a public parameter pp and a preimage $x \in \{0,1\}^n$, and deterministically outputs y.

$E_1(\mathsf{pp}, (i, b); \rho) \to \mathsf{ct}$: Takes public parameters pp, an index $i \in [n]$, a bit $b \in \{0,1\}$ and randomness ρ, and outputs a ciphertext ct.

$E_2(\mathsf{pp}, y, (i, b); \rho) \to k$: Takes a public parameter pp, a value y, an index $i \in [n]$, a bit $b \in \{0,1\}$ and randomness $\rho \in \{0,1\}^r$, and outputs a key $k \in \{0,1\}^\ell$. Notice that unlike E_1, which does not take y as input, the algorithm E_2 does take y as input.

$D(\mathsf{pp}, \mathsf{ct}, x) \to k$: Takes a public parameter pp, a ciphertext ct, a preimage $x \in \{0,1\}^n$, and deterministically outputs a key $k \in \{0,1\}^\ell$.

We require the following properties.

Correctness. For security parameter λ, for any choice of $\mathsf{pp} \in K(1^\lambda)$, any index $i \in [n]$, any preimage $x \in \{0,1\}^n$ and any randomness value ρ, the following holds: letting $y := f(\mathsf{pp}, x)$, and $\mathsf{ct} := E_1(\mathsf{pp}, (i, x_i); \rho)$, we have $E_2(\mathsf{pp}, y, (i, x_i); \rho) = D(\mathsf{pp}, \mathsf{ct}, x)$.

Definition 2.1 ((k, n)-One-wayness.). *For any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have*

$$\Pr\left[f(\mathsf{pp}, \mathcal{A}(\mathsf{pp}, y)) = y \ : \quad S \leftarrow \mathcal{A}(1^\lambda), \mathsf{pp} \to K(1^\lambda); x \leftarrow S; y = f(\mathsf{pp}, x) \ \right] \leq \mathit{negl}(\lambda).$$

Here, the adversary is constrained to output only a (k, n)-source.

Definition 2.2 (Security for encryption.). *For any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have*

$$\Pr\left[\mathcal{A}(\mathsf{pp}, x, \mathsf{ct}, k_b) = b \ : \ \begin{array}{c} (x, i) \leftarrow \mathcal{A}(1^\lambda); \mathsf{pp} \leftarrow K(1^\lambda); \\ b \leftarrow \{0,1\}; \rho \leftarrow \{0,1\}^r; \mathsf{ct} \leftarrow E_1(\mathsf{pp}, (i, 1 - x_i); \rho); \\ k_0 \to E_2(\mathsf{pp}, f(\mathsf{pp}, x), (i, 1 - x_i); \rho); k_1 \leftarrow \{0,1\}^n \end{array}\right] \leq 1/2 + \mathit{negl}(\lambda).$$

Definition 2.3 ((k, n)-Smoothness.). *We say that (K, f, E_1, E_2, D) is (k, n)-smooth if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$, such that for all $\lambda \in \mathbb{N}$, we have*

$$\Pr\left[\mathcal{A}(\mathsf{pp}, y) = b \ : \ \begin{array}{c} (S_0, S_1) \leftarrow \mathcal{A}(1^\lambda); \mathsf{pp} \leftarrow K(1^\lambda); \\ b \leftarrow \{0,1\}; x_0 \leftarrow S_0; x_1 \leftarrow S_1; y = f(\mathsf{pp}, x_b) \end{array}\right] \leq 1/2 + \mathit{negl}(\lambda).$$

where the distributions S_0 and S_1 output by the adversary $\mathcal{A}$ are constrained to be (k, n)-sources.

2.2 Hinting PRG

Next, we review the definition of Hinting PRG proposed in [KW19]. Let $n(\cdot)$ and $\ell(\cdot)$ be some polynomials. An (n, ℓ)-hinting PRG scheme consists of two PPT algorithms Setup, Eval with the following syntax.

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pp}, n)$: The setup algorithm takes as input the security parameter λ, and length parameter ℓ, and outputs public parameters pp and input length $n = n(\lambda)$.

$\mathsf{Eval}(\mathsf{pp}, s \in \{0,1\}^n, i \in [n] \cup \{0\}) \to y \in \{0,1\}^\ell$: The evaluation algorithm takes as input the public parameters pp, an n-bit string s, an index $i \in [n] \cup \{0\}$ and outputs an ℓ bit string y.

Definition 2.4. *An (n, ℓ)-hinting PRG scheme* $(\mathsf{Setup}, \mathsf{Eval})$ *is said to be secure if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds:*

$$\Pr\left[\mathcal{A}\Big(\mathsf{pp}, y_0^\beta, \{y_{i,b}^\beta\}_{i\in[n],b\in\{0,1\}}\Big) = \beta \ : \begin{array}{c} (\mathsf{pp}, n) \leftarrow \mathsf{Setup}(1^\lambda); s \leftarrow \{0,1\}^n; \\ \beta \leftarrow \{0,1\}; y_0^0 = \mathsf{Eval}(\mathsf{pp}, s, 0); y_0^1 \leftarrow \{0,1\}^\ell; \\ y_{i,s_i}^0 = \mathsf{Eval}(\mathsf{pp}, s, i); y_{i,\overline{s_i}}^0 \leftarrow \{0,1\}^\ell \ \forall i \in [n] \\ y_{i,b}^1 \leftarrow \{0,1\}^\ell \ \forall i \in [n], b \in \{0,1\}; \end{array}\right] \leq 1/2 + negl(\lambda)$$

2.3 Strong Extractors

Extractors are combinatorial objects used to 'extract' uniformly random bits from a source that has high randomness, but is not uniformly random. In this work, we will be using seeded extractors. In a seeded extractor, the extraction algorithm takes as input a sample point x from the high randomness source $\mathcal{X}$, together with a short seed $\mathfrak{s}$, and outputs a string that looks uniformly random. Here, we will be using strong extractors, where the extracted string looks uniformly random even when the seed is given.

Definition 2.5. *A (k, ϵ) strong extractor* $\mathsf{Ext} : \mathbb{D} \times \mathbb{S} \to \mathbb{Y}$ *is a deterministic algorithm with domain $\mathbb{D}$, range $\mathbb{Y}$ and seed space $\mathbb{S}$ such that for every source $\mathcal{X}$ on $\mathbb{D}$ with min-entropy at least k, the following two distributions have statistical distance at most ϵ:*

$$\mathcal{D}_1 = \{(\mathfrak{s}, \mathsf{Ext}(x, \mathfrak{s})) : \mathfrak{s} \leftarrow \mathbb{S}, x \leftarrow \mathcal{X}\}, \mathcal{D}_2 = \{(\mathfrak{s}, y) : \mathfrak{s} \leftarrow \mathbb{S}, y \leftarrow \mathbb{Y}\}$$

Using the Leftover Hash Lemma, we can construct strong extractors from pairwise-independent hash functions. More formally, let $\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^m\}$ be a family of pairwise independent hash functions, and let $m = k - 2\log(1/\epsilon)$. Then $\mathsf{Ext}(x, h) = h(x)$ is a strong extractor with h being the seed. Such hash functions can be represented using $O(n)$ bits.

2.4 Assumptions

Φ-Hiding Assumption. The Φ-Hiding assumption, introduced by Cachin et al. [CMS99], informally states that given an RSA modulus N, it is hard to find the factors of $\phi(N)$, or to distinguish a factor of $\phi(N)$ from an integer co-prime to $\phi(N)$. To formally state this assumption, we need to introduce some notations, and will be following the work of [HOR15] for the same. Let PRIMES(λ) denote the set of primes of bit-length λ, and let

$$\mathsf{RSA}(\lambda) = \{N : N = pq;\ p, q \in \mathrm{PRIMES}(\lambda/2);\ gcd(p-1, q-1) = 2\}.$$

For any $e \leq 2^\lambda$, let

$$\mathsf{RSA}_e(\lambda) = \{N \in \mathsf{RSA}(\lambda) : e \text{ divides } \phi(N)\}.$$

Assumption 1 (Φ-Hiding)**.** *The Φ-Hiding assumption states that for all $\epsilon > 0$, integers e such that $3 < e < 2^{\lambda/4-\epsilon}$ and PPT adversaries $\mathcal{A}$,*

$$\Pr[\mathcal{A}(N, e) = 1 : N \leftarrow \mathsf{RSA}(\lambda)] - \Pr[\mathcal{A}(N, e) = 1 : N \leftarrow \mathsf{RSA}_e(\lambda)] \leq negl(\lambda).$$

q-DDHI Assumption. A variant of this assumption is introduced in [BB04]. We say that a PPT algorithm GGen is a group generator if it takes a security parameter 1^λ as input and outputs a group description $\mathcal{G} := (\mathbb{G}, p)$ where $\mathbb{G}$ is a group with prime order $p = \Omega(2^\lambda)$, from which one can efficiently sample a generator uniformly at random.

Assumption 2 (q-DDHI)**.** *Let* GGen *be a group generator and $q = q(\lambda) =$* $\mathsf{poly}(\lambda)$*. We say that q-Decisional Diffie Hellman Inversion assumption holds with respect to* GGen *if for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every $\lambda \in \mathbb{N}$, we have*

$$\Pr\left[\mathcal{A}(\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^q}, T_b) = b \ : \begin{array}{c} (\mathbb{G}, p) = \mathcal{G} \leftarrow \mathsf{GGen}(1^\lambda); g \leftarrow \mathbb{G}; \alpha, r \leftarrow \mathbb{Z}_p^* \\ b \leftarrow \{0,1\}; T_0 = g^{1/\alpha}; T_1 \leftarrow g^r; \end{array}\right] \leq 1/2 + negl(\lambda).$$

q-DBDHI Assumption. This assumption is introduced in [BB04]. We say that a PPT algorithm GGen is a group generator if it takes a security parameter 1^λ as input and outputs a group description $\mathcal{G} := (\mathbb{G}_1, \mathbb{G}_T, e, p)$. Here, $\mathbb{G}_1$ and $\mathbb{G}_2$ are groups with prime order $p = \Omega(2^\lambda)$, from which one can efficiently sample a generator uniformly at random. $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is an efficiently computable pairing operation.

Assumption 3 (q-DBDHI)**.** *Let* GGen *be a group generator and* $q = q(\lambda) = \mathsf{poly}(\lambda)$*. We say that q-Decisional Bilinear Diffie Hellman Inversion assumption holds with respect to* GGen *if for every PPT adversary $\mathcal{A}$, there exists a negligible function $negl(\cdot)$ such that for every $\lambda \in \mathbb{N}$, we have*

$$\Pr\left[\mathcal{A}(\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^q}, T_b) = b \ : \ \begin{array}{c}(\mathbb{G}_1, \mathbb{G}_2, e, p) = \mathcal{G} \leftarrow \mathsf{GGen}(1^\lambda); g \leftarrow \mathbb{G}_1; \alpha, r \leftarrow \mathbb{Z}_p^* \\ b \leftarrow \{0,1\}; T_0 = e(g,g)^{1/\alpha}; T_1 \leftarrow e(g,g)^r;\end{array}\right] \leq 1/2 + negl(\lambda).$$

3 Hashing and Randomness Extraction under Φ-Hiding

In this section, we will prove two useful lemmas about universal hashing and randomness extraction under the Φ-hiding assumption. Here we consider special groups defined w.r.t. an RSA modulus N. These lemmas will be crucial in proving security of our Φ-hiding based constructions later in Sections 4 and 6.

3.1 Number Theory: Prime Number Theorems for Arithmetic Progressions

First, we recall some important theorems from the number theory literature about prime numbers that we will be relying on in this work.

In 1837, Dirichlet [Dir37] proved that for two co-prime positive integers a and q, the sequence $\{a + qn\}_{n=0}^\infty$ contains infinitely many primes. Further, Dirichlet and Legendre conjectured that the number of primes in this in this sequence less than x is around $\frac{1}{\phi(q)}\mathsf{Li}(x)$, where $\mathsf{Li}(\cdot)$ is the logarithmic integral function (and a good approximation of the prime counting function). In 1896, de la Vallée Poussin [DlVP97] proved the conjecture. Below we state the refined theorem statement as has been improved in a long line of works (to cite a few [DlVP97, New80, Zag97, Tao09, Sop10]).

Theorem 3.1 (Prime Number Theorem for Arithmetic Progressions (Paraphrased))**.** *For any two co-prime integers q, a. Define $\theta(x; q, a)$ to be number of primes p less than x such that $a = p \pmod q$. Concretely,*

$$\theta(x; q, a) = |\left\{p < x \ : \ p \in \cup_{i \leq \lceil \log x \rceil} \mathrm{PRIMES}(i) \ \wedge \ a = p \ (mod \ q)\right\}|.$$

Then, the following is true:

$$\forall x, \quad \theta(x; q, a) = (1 + o_q(1)) \frac{1}{\phi(q)} \frac{x}{\log x},$$

where the subscript q in the notation $o_q(1)$ denotes that the implied constant could depend q. And, $o_q(1) \to 0$ as $x \to \infty$.

Combining the above theorem with the famed prime number (density) theorem, we get the following corollary.

Corollary 3.1 (Prime Number Density in Arithmetic Progressions)**.** *For any two co-prime integers q, a, we have the following*

$$\forall x, \quad \Pr_p\left[a = p \ (mod \ q) \ : \ p < x \ \wedge \ \cup_{i \leq \lceil \log x \rceil} \mathrm{PRIMES}(i)\right] = (1 + o_{q,x}(1)) \frac{1}{\phi(q)},$$

where the subscripts q, x in the notation $o_{q,x}(1)$ denotes that the implied constant could depend q, x. And, $o_{q,x}(1) \to 0$ as $x \to \infty$.

In this work, we need a slightly stronger guarantee for our results in which the primes p we consider are in a specific range which is fixed bit length. Below we state the corollary which agains follows by combining Theorem 3.1 and prime number (density) theorem, and in turn suffices for our results.

Corollary 3.2 (Bounded Range Prime Number Density in Arithmetic Progressions)**.** *For any two co-prime integers q, a, we have the following*

$$\forall \lambda \in \mathbb{N}, \quad \Pr_p\left[a = p \ (mod\ q) \ : \ p \in \text{PRIMES}(\lambda)\right] = (1 + o_{q,\lambda}(1))\frac{1}{\phi(q)},$$

where the subscripts q, λ in the notation $o_{q,\lambda}(1)$ denotes that the implied constant could depend q, λ. And, $o_{q,\lambda}(1) \to 0$ as $\lambda \to \infty$.

Remark 3.1. *It turns out we actually need a much weaker guarantee than what is provided above. Concretely, any upper bound as long as it is a fixed inverse polynomial in $\phi(q)$ is sufficient for us.*

3.2 A New Hashing Lemma

Consider an RSA modulus $N = pq$ for $\kappa/2$-bit primes p, q, and let $g \in \mathbb{Z}_N^*$ be a random element in the multiplicative group $\mathbb{Z}_N^*$. For ℓ-bit inputs, consider the following family of hash functions:

$$\mathcal{K} = \left\{(a, b, \{e_{i,b}\}_{i\in[\ell], b\in\{0,1\}}) \in \mathbb{Z}_N^{2\ell+2} \ : \ a, b \in \mathbb{Z}_N; \forall\ i \in [\ell], b \in \{0,1\},\ e_{i,b} \in \text{PRIMES}(\lambda)\right\},$$
$$H \ : \ \mathcal{K} \times \mathcal{X} \to \mathbb{Z}_N, \qquad H\left((a, b, \{e_{i,b}\}_{i,b}), x\right) = g^{(ax+b)\prod_i e_{i,x_i}} \pmod N.$$

Here x is intepreted as an integer for arithmetic operations, and x_i denotes the i^{th} bit of x when intepreted as a binary string. Whenever it is clear from context, we will drop the hash key as an explicit input to the function and write either $H(x)$ or $H_K(x)$ instead of $H(K, x)$ for some hash key $K = (a, b, \{e_{i,b}\}_{i,b})$. Also, throughout we assume that ℓ is sufficiently large, i.e. $\ell > \kappa + 2\lambda$.

We show that under the Φ-hiding assumption, the above hash function provides useful guarantees. Concretely, we show the following:

Theorem 3.2. *Let p_i denote the i^{th} prime, i.e. $p_1 = 2, p_2 = 3, \ldots$, and $\widetilde{e_i} = \lceil \log_{p_i} N \rceil$. And, let f_i denote $p_i^{\widetilde{e_i}}$ for all i.*

Assuming the Φ-hiding assumption holds, for every PPT adversary $\mathcal{A}$, non-negligible function $\epsilon(\cdot)$, polynomial $v(\cdot)$, for all $\lambda, \kappa \in \mathbb{N}$, satisfying $\kappa \geq 5\lambda$ and $\epsilon = \epsilon(\lambda) > 1/v(\lambda)$, the following holds,

$$\Pr\left[\mathcal{A}(N, g, K, y) = 1 \ : \ \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\ a, b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ g \leftarrow \mathbb{Z}_N^*;\ K = (a, b, \{e_{i,b}\}_{i,b}); \\ x \leftarrow \mathcal{X};\ y = H_K(x)^{f_1 \cdot f_2} \end{array}\right]$$
$$- \Pr\left[\mathcal{A}(N, g, K, y) = 1 \ : \ \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\ a, b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ \widetilde{g} \leftarrow \mathbb{Z}_N^*;\ K = (a, b, \{e_{i,b}\}_{i,b}) \\ g = \widetilde{g}^{\prod_{i=3}^{j_\epsilon} f_i};\ h \leftarrow \mathbb{Z}_N^*;\ y = h^{\prod_{i=1}^{j_\epsilon} f_i} \end{array}\right] \leq \epsilon(\lambda)/2,$$

where j_ϵ is the smallest index such that $p_{j_\epsilon} > (2\sqrt{2}\log N/\epsilon)^4$.

Proof. Let the prime factorization of $\phi(N)$ be $\phi(N) = \prod_i r_i^{k_i}$ for $i = 1$ to ℓ_N, where $k_i \geq 1$, ℓ_N denotes number of distinct prime factors of $\phi(N)$, and r_i's are the distinct prime factors arranged in an increasing order. The proof is divided into two parts. First, we argue that $(ax+b)\prod_i e_{i,x_i} \bmod r_j^{k_j}$ is statistically close to random over $\mathbb{Z}_{r_j^{k_j}}$ for all prime factors of $\phi(N)$ greater than p_{j_ϵ}. In the second part of the proof, we show using Φ-hiding that the hash function H could be made lossy on all prime factors of $\phi(N)$ less than or equal to p_{j_ϵ}. Thus, the theorem follows. For proving the first part, we employ a tight Leftover Hash Lemma proof. And for the second part, we rely on Φ-hiding to introduce lossiness.

Notation. Here and throughout, for any ℓ-bit string x, we use $\mathbf{e}_x$ to denote the following product $\prod_{i\in[\ell]} e_{i,x_i}$.

Part 1. The statistical argument. Here we show that if we look at the congruent CRT representation of the exponent $(ax+b)\cdot \mathbf{e}_x$ corresponding to prime factors greater p_{j_ϵ}, then (for a randomly chosen hash key K and input x) they are at most $\epsilon/3$-statistically far from an integer that is chosen at random with the constraint that its congruent CRT representation corresponding to prime factors less than or equal to p_{j_ϵ} is same as for $(ax+b)\cdot \mathbf{e}_x$. Concretely, we show that following:

Theorem 3.3. *Let p_i denote the i^{th} prime, i.e. $p_1 = 2, p_2 = 3, \ldots,$ and $\widetilde{e_i} = \lceil \log_{p_i} N \rceil$. Also, for RSA modulus N, let $\phi(N) = \prod r_i^{k_i}$, where $k_i \geq 1$, ℓ_N denotes number of distinct prime factors of $\phi(N)$, and r_i's are the distinct prime factors arranged in an increasing order.*

For every (possibly unbounded) adversary $\mathcal{A}$, non-negligible function $\epsilon(\cdot)$, polynomial $v(\cdot)$, for all $\lambda, \kappa \in \mathbb{N}$, satisfying $\kappa \geq 5\lambda$ and $\epsilon = \epsilon(\lambda) > 1/v(\lambda)$, the following holds,

$$\Pr\left[\mathcal{A}(N,K,y)=1 \ : \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\ a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \mathrm{PRIMES}(\lambda) \\ K = (a,b,\{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{X};\ y = (ax+b)\cdot \mathbf{e}_x\ (mod\ \phi(N)) \end{array}\right]$$
$$-\Pr\left[\mathcal{A}(N,K,\widetilde{y})=1 \ : \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\ a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \mathrm{PRIMES}(\lambda) \\ K = (a,b,\{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{X};\ y = (ax+b)\cdot \mathbf{e}_x\ (mod\ \phi(N)) \\ y = (y^{(1)},\ldots,y^{(\ell_N)})\ where\ y^{(j)} = y\ (mod\ r_i^{k_i}) \\ \widetilde{y}^{(i)} = y^{(i)}\ for\ i\ such\ that\ r_i \leq p_{j_\epsilon} \\ \widetilde{y}^{(i)} \leftarrow \mathbb{Z}_{r_i^{k_i}}\ otherwise \\ \widetilde{y} = (\widetilde{y}^{(1)},\ldots,\widetilde{y}^{(\ell_N)}) \end{array}\right] \leq \epsilon(\lambda)/3,$$

where j_ϵ is the smallest index such that $p_{j_\epsilon} > (2\sqrt{2}\log N/\epsilon)^4$.

Proof. Let $\epsilon = \epsilon(\lambda)$, and the event bad correspond to the scenario when at least one of the λ-bit primes $\{e_{i,b}\}_{i,b}$ are *not* co-prime w.r.t. $\phi(N)$, or $a,b \geq \phi(N)$. Note that the probability of this event happening can be bounded as $\Pr[\mathsf{bad}] \leq 2\ell \cdot \frac{4\ell_N\lambda}{2^\lambda} + 2\cdot\frac{\phi(N)}{N} = \mathrm{negl}(\lambda)$. Now for the rest of the analysis, we condition on the event 'bad' not happening, but do not explicitly write it for ease of exposition. That is, in the remaining proof of this theorem we always assume that $\{e_{i,b}\}_{i,b}$ are co-prime w.r.t. $\phi(N)$, and $a,b < \phi(N)$.

Next, consider the following simpler case. For any prime r and exponent k, consider the hash function $h(x) = (ax+b)\prod_i e_{i,x_i} \pmod{r^k}$. Here a,b are sampled uniformly at random from $\mathbb{Z}_{r^k}$ and $e_{i,b}$'s are random λ-bit primes. We first claim the following:

Lemma 3.1. *For every prime $r > 3$, integer $k \geq 1$,*

$$\Pr_{\substack{a,b,\{e_{i,b}\}_{i,b},\\ x\neq y}}\left[(ax+b)\cdot \mathbf{e}_x = (ay+b)\cdot \mathbf{e}_y\ (mod\ r^k)\right] \leq \frac{1}{r^k}\left(1 + \frac{2k}{\sqrt{r}} + \frac{k\cdot r^k}{|\mathcal{X}|}\right).$$

Proof. Note that we have already conditioned on the event that all the λ-bit primes $\{e_{i,b}\}_{i,b}$ are co-prime w.r.t. $\phi(N)$. (This happens with all but negligible probability, thus does not affect the remaining analysis.) Now fix any $c \in \mathbb{Z}_{r^k}$. We have that if

$$(ax+b)\cdot \mathbf{e}_x = (ay+b)\cdot \mathbf{e}_y = c \implies \begin{array}{c} ax+b = c\cdot \mathbf{e}_x^{-1} \\ ay+b = c\cdot \mathbf{e}_y^{-1} \end{array} \implies a(x-y) = c\left(\mathbf{e}_x^{-1} - \mathbf{e}_y^{-1}\right).$$

Consider the following cases — (1) $r \nmid x-y$, (2) $r \mid x-y \wedge r^2 \nmid x-y$, (3) $\ldots$, (k) $r^{k-1} \mid x-y \wedge r^k \nmid x-y$, $(k+1)$ $r^k \mid x-y$. We know that $\Pr_{x\neq y}[\text{Case } (i)] \leq (\frac{1}{r^{i-1}} - \frac{1}{r^i} + \frac{r^i}{|\mathcal{X}|})$ for $i < k+1$ and $\Pr_{x\neq y}[\text{Case } (k+1)] \leq 1/r^k$. Now, in case (1), for every $c \in \mathbb{Z}_{r^k}$, there exists a unique (a,b) pair such that the aforementioned equations are satisfied. (Because if $r \nmid x-y$, then $(x-y)^{-1}$ is unique and always exists.) So, we can write that

$$\Pr_{a,b}\left[(ax+b)\cdot \mathbf{e}_x = (ay+b)\cdot \mathbf{e}_y \pmod{r^k}\ \middle|\ \text{Case } (1)\right] = \frac{1}{r^k}.$$

Next, consider case (i) for $i > 1$. There are $(i-1)$ sub-cases — $(i.1)$ $r \nmid \mathbf{e}_x - \mathbf{e}_y$, $(i.2)$ $r \mid \mathbf{e}_x - \mathbf{e}_y \ \wedge \ r^2 \nmid \mathbf{e}_x - \mathbf{e}_y$, (3) $\ldots$, $(i.(i-1))$ $r^{i-2} \mid \mathbf{e}_x - \mathbf{e}_y \ \wedge \ r^{i-1} \nmid \mathbf{e}_x - \mathbf{e}_y$, $(i.i)$ $r^{i-1} \mid \mathbf{e}_x - \mathbf{e}_y$. In case $(i.1)$, for a collision to occur it must hold that $c = 0 \bmod r^{i-1}$ since $x - y = 0 \bmod r^{i-1}$ (as $r^{i-1} \mid x - y$). We have that the number of such c values is r^{k-i+1}. Now for each such c, we can solve for r^{i-1} pairs of solutions for (a, b). Similarly in other cases, i.e. case $(i.j)$ for $1 < j \leq i$, we have that number of satisfying c values in $\mathbb{Z}_{r^k}$ will be r^{k-i+j}, and for each such c there will exist r^{i-1} pairs of solutions for (a, b). Now, for any i, let $\pi(r^i)$ denote the following probability

$$\pi(r^i) = \Pr_{\{e_{i,b}\}_{i,b}} \left[\mathbf{e}_x = \mathbf{e}_y \pmod{r^i}\right].$$

Next, combining all the above cases and sub-cases, we get the following:

$$\begin{aligned}
\Pr_{\substack{a,b,\{e_{i,b}\}_{i,b},\\ x\neq y}} \left[(ax+b)\cdot \mathbf{e}_x = (ay+b)\cdot \mathbf{e}_y \pmod{r^k}\right] &\leq \left(1 - \frac{1}{r} + \frac{r}{|\mathcal{X}|}\right)\frac{1}{r^k} \\
&+ \sum_{i=2}^{k} \left(\frac{1}{r^{i-1}} - \frac{1}{r^i} + \frac{r^i}{|\mathcal{X}|}\right)\left(r^{k-i+1}\cdot \frac{r^{i-1}}{r^{2k}} + k\cdot \pi(r^{i-1})\cdot r^k \cdot \frac{r^{i-1}}{r^{2k}}\right) \\
&+ \frac{1}{r^k}\left(\frac{r^k}{r^{2k}} + k\cdot \pi(r^k)\cdot r^k\cdot \frac{r^k}{r^{2k}}\right).
\end{aligned}$$

$$\Rightarrow \Pr_{\substack{a,b,\{e_{i,b}\}_{i,b},\\ x\neq y}} \left[(ax+b)\cdot \mathbf{e}_x = (ay+b)\cdot \mathbf{e}_y \pmod{r^k}\right] \leq \frac{1}{r^k} + \frac{k}{|\mathcal{X}|} + \frac{k}{r^k}\left(1 - \frac{1}{r}\right)\sum_{i=2}^{k}\pi(r^{i-1}) + \frac{1}{r^k}\cdot \pi(r^k). \tag{1}$$

$$\leq \frac{1}{r^k}\left(1 + k\cdot \sum_{i=1}^{k}\pi(r^i)\right) + \frac{k}{|\mathcal{X}|}. \tag{2}$$

Now let us try to bound the probability $\pi(r^i)$. Fix any $x \neq y$ and let j^* denote the first index such that $x_{j^*} \neq y_{j^*}$. Note that,

$$\pi(r^i) = \Pr_{\{e_{i,b}\}_{i,b}} \left[\mathbf{e}_x = \mathbf{e}_y \pmod{r^i}\right] = \Pr_{e_{j^*,x_{j^*}}} \left[e_{j^*,x_{j^*}} = \mathbf{e}_y \cdot \prod_{j\neq j^*} e_{j,x_j}^{-1} \pmod{r^i}\right].$$

Next, using the theorem on bounded range prime number density in arithmetic progressions (see Corollary 3.2), we get that $\pi(r^i) \leq \frac{1}{r^{i/2}}$.[5] Therefore, we get that

$$\sum_{i=1}^{k}\pi(r^i) \leq \frac{1 - r^{-(k+1)/2}}{r^{1/2} - 1} \leq 2r^{-1/2}.$$

Combining this with Equation (1), we get that

$$\Rightarrow \Pr_{\substack{a,b,\{e_{i,b}\}_{i,b},\\ x\neq y}} \left[(ax+b)\cdot \mathbf{e}_x = (ay+b)\cdot \mathbf{e}_y \pmod{r^k}\right] \leq \frac{1}{r^k}\left(1 + \frac{2k}{\sqrt{r}}\right) + \frac{k}{|\mathcal{X}|}. \tag{3}$$

This completes the proof of Lemma 3.1. ■

[5] Note that here we are using a very weak upper bound. Our analysis could be further tightened, but since a weaker guarantee is sufficient for the proof, thus we stick with it.

Next, using Lemma 3.1, we can claim the following.

Lemma 3.2. *For every prime $r > 3$, integer $k \geq 1$, every (possibly unbounded) adversary $\mathcal{A}$, the following holds*

$$\Pr\left[\mathcal{A}(r^k, K, y_\beta) = \beta \ : \ \begin{array}{c} a, b \leftarrow \mathbb{Z}_{r^k}; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K = (a, b, \{e_{i,b}\}_{i,b}); \quad x \leftarrow \mathcal{X} \\ y_0 = (ax+b) \cdot \mathbf{e}_x \ (mod\ r^k); \ y_1 \leftarrow \mathbb{Z}_{r^k} \end{array}\right] \leq \sqrt{\frac{k}{2\sqrt{r}}} + \sqrt{\frac{kr^k}{2|\mathcal{X}|}}$$

Proof. The proof of this lemma is similar to the proof of Leftover Hash Lemma [HILL99, DRS04, DORS08] where the collision probability is used as obtained in Lemma 3.1. Concretely, for any prime r and exponent k, consider the hash function $H^{(r,k)}(K^{(r,k)}, x) = (ax+b)\prod_i e_{i,x_i} \pmod{r^k}$. Here a, b are sampled uniformly at random from $\mathbb{Z}_{r^k}$ and $e_{i,b}$'s are random λ-bit primes. The key consists of $K^{(r,k)} = (a, b, \{e_{i,b}\}_{i,b})$. Note that hash function $H^{(r,k)} \ : \ \mathcal{K}^{(r,k)} \times \mathcal{X} \to \mathbb{Z}_{r^k}$.

Let $\delta = \Pr_{\substack{a,b,\{e_{i,b}\}_{i,b}, \\ x \neq y}}\left[(ax+b) \cdot \mathbf{e}_x = (ay+b) \cdot \mathbf{e}_y \pmod{r^k}\right]$. Now, we can write the following

$$\begin{aligned} \mathsf{SD}\left((H^{(r,k)}, H^{(r,k)}(X)), (H^{(r,k)}, U_{\mathbb{Z}_{r^k}})\right) &\leq \frac{1}{2}\sqrt{|\mathcal{K}^{(r,k)}| \cdot r^k}\sqrt{\frac{\delta}{|\mathcal{K}^{(r,k)}|} + \frac{1}{|\mathcal{X}| \cdot |\mathcal{K}^{(r,k)}|} - \frac{1}{|\mathcal{K}^{(r,k)}| \cdot r^k}} \\ &\leq \frac{1}{2}\sqrt{\delta \cdot r^k - 1 + \frac{r^k}{|\mathcal{X}|}} \\ &\leq \frac{1}{2}\left(\sqrt{\delta \cdot r^k - 1} + \sqrt{\frac{r^k}{|\mathcal{X}|}}\right) \end{aligned}$$

where SD corresponds to the statistical distance. Using Lemma 3.1, we can simply it further to $\sqrt{\frac{k}{2\sqrt{r}}} + \sqrt{\frac{kr^k}{2|\mathcal{X}|}}$. This completes the proof of Lemma 3.2. ■

Finally, using union bounds, congruence due to Chinese remainder theorem, and extending the analyses of Lemmas 3.1 and 3.2, we get the following

$$\begin{aligned} &\Pr\left[\mathcal{A}(N, K, y) = 1 \ : \ \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa); \ a, b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K = (a, b, \{e_{i,b}\}_{i,b}); \quad x \leftarrow \mathcal{X} \\ y = (ax+b) \cdot \mathbf{e}_x \pmod{\phi(N)} \end{array}\right] \\ &- \Pr\left[\mathcal{A}(N, K, \widetilde{y}) = 1 \ : \ \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa); \ a, b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K = (a, b, \{e_{i,b}\}_{i,b}); \quad x \leftarrow \mathcal{X} \\ y = (ax+b) \cdot \mathbf{e}_x \pmod{\phi(N)} \\ y = (y^{(1)}, \ldots, y^{(\ell_N)}) \text{ where } y^{(j)} = y \pmod{r_i^{k_i}} \\ \widetilde{y}^{(i)} = y^{(i)} \text{ for } i \text{ such that } r_i \leq p_{j_\epsilon} \\ \widetilde{y}^{(i)} \leftarrow \mathbb{Z}_{r_i^{k_i}} \text{ otherwise} \\ \widetilde{y} = (\widetilde{y}^{(1)}, \ldots, \widetilde{y}^{(\ell_N)}) \end{array}\right] \\ &\leq \Pr[\mathsf{Bad}] + \frac{\ell_N}{\sqrt{2p_{j_\epsilon}^{1/2}}} + \sqrt{\frac{\log N \cdot \phi(N)}{2|\mathcal{X}|}} \\ &\leq \text{negl}_1(\lambda) + \epsilon/4 + \text{negl}_2(\lambda) < \epsilon/3. \end{aligned}$$

Here to argue that $\sqrt{\frac{\log N \cdot \phi(N)}{2|\mathcal{X}|}}$ is negligible in λ, we use the fact that input domain $\mathcal{X} = \{0,1\}^\ell$ is quite large, i.e. $\ell > \kappa + 2\lambda$.

This completes the proof of the first part, which shows a tight bound on the statistical distance between the real and intermediate hybrid distribution. ■

Part 2. The computational argument. Here we show that, using Φ-hiding, the generator g instead of sampling uniformly at random could be sampled as $g^{\prod_{i=1}^{j_\epsilon} f_i}$, where j_ϵ is the smallest index such that $p_{j_\epsilon} > (2\sqrt{2}\log N/\epsilon)^4$. This removes information about the input x completely. Concretely, we show that following:

Theorem 3.4. *Let p_i denote the i^{th} prime, i.e. $p_1 = 2, p_2 = 3, \ldots,$ and $\widetilde{e}_i = \lceil \log_{p_i} N \rceil$ and let f_i denote $p_i^{\widetilde{e}_i}$ for all i. Also, for RSA modulus N, let $\phi(N) = \prod r_i^{k_i}$, where $k_i \geq 1$, ℓ_N denotes number of distinct prime factors of $\phi(N)$, and r_i's are the distinct prime factors arranged in an increasing order.*

Assuming the Φ-hiding assumption holds, for every PPT adversary $\mathcal{A}$, non-negligible function $\epsilon(\cdot)$, polynomial $v(\cdot)$, for all $\lambda, \kappa \in \mathbb{N}$, satisfying $\kappa \geq 5\lambda$ and $\epsilon = \epsilon(\lambda) > 1/v(\lambda)$, there exists a negligible function negl$(\cdot)$ such that the following holds,

$$\Pr\left[\mathcal{A}(N, g, K, g^{\widetilde{y}\cdot f_1 \cdot f_2}) = 1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a, b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \mathrm{PRIMES}(\lambda) \\ K = (a, b, \{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{X};\; g \leftarrow \mathbb{Z}_N^* \\ y = (ax+b)\cdot \mathbf{e}_x \;(mod\ \phi(N)) \\ y = (y^{(1)}, \ldots, y^{(\ell_N)}) \textit{ where } y^{(j)} = y \;(mod\ r_i^{k_i}) \\ \widetilde{y}^{(i)} = y^{(i)} \textit{ for } i \textit{ such that } r_i \leq p_{j_\epsilon} \\ \widetilde{y}^{(i)} \leftarrow \mathbb{Z}_{r_i^{k_i}} \textit{ otherwise} \\ \widetilde{y} = (\widetilde{y}^{(1)}, \ldots, \widetilde{y}^{(\ell_N)}) \end{array}\right]$$
$$- \Pr\left[\mathcal{A}(N, g^{\prod_{i=3}^{j_\epsilon} f_i}, K, h^{\prod_{i=1}^{j_\epsilon} f_i}) = 1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a, b \leftarrow \mathbb{Z}_N \\ g \leftarrow \mathbb{Z}_N^*;\; e_{i,b} \leftarrow \mathrm{PRIMES}(\lambda) \\ K = (a, b, \{e_{i,b}\}_{i,b});\; h \leftarrow \mathbb{Z}_N^* \end{array}\right] \leq \textit{negl}(\lambda),$$

where j_ϵ is the smallest index such that $p_{j_\epsilon} > (2\sqrt{2}\log N/\epsilon)^4$.

Proof. The proof proceeds by a sequence of $j_\epsilon - 2$ hybrids, where in the i^{th} hybrid we use Φ-hiding on $(i+2)^{th}$ prime p_{i+2} to sample g as $g = \widetilde{g}^{f_{i+2}}$ where $\widetilde{g} \leftarrow \mathbb{Z}_N^*$ instead of sampling it as $g \leftarrow \mathbb{Z}_N^*$. Let δ_{i^*} for $2 \leq i^* \leq j_\epsilon$ denote the following probability:

$$\delta_{i^*} = \Pr\left[\mathcal{A}(N, \widetilde{g}, K, \widetilde{h}) = 1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a, b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \mathrm{PRIMES}(\lambda) \\ K = (a, b, \{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{X};\; g \leftarrow \mathbb{Z}_N^* \\ y = (ax+b)\cdot \mathbf{e}_x \;(\mathrm{mod}\ \phi(N)) \\ y = (y^{(1)}, \ldots, y^{(\ell_N)}) \text{ where } y^{(j)} = y \;(\mathrm{mod}\ r_i^{k_i}) \\ \widetilde{y}^{(i)} = y^{(i)} \text{ for } i \text{ such that } r_i \leq p_{j_\epsilon} \\ \widetilde{y}^{(i)} \leftarrow \mathbb{Z}_{r_i^{k_i}} \text{ otherwise} \\ \widetilde{y} = (\widetilde{y}^{(1)}, \ldots, \widetilde{y}^{(\ell_N)}) \\ \widetilde{g} = g^{\prod_{i=3}^{i^*} f_i};\; \widetilde{h} = g^{\widetilde{y}\prod_{i=1}^{i^*} f_i} \end{array}\right]. \tag{4}$$

Formally, we prove the following under the Φ-hiding assumption:

Lemma 3.3. *Assuming the Φ-hiding assumption holds, for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda, \kappa \in \mathbb{N}$, satisfying $\kappa \geq 5\lambda$ and every index $2 \leq i^* < j_\epsilon$, the following holds: $\delta_{i^*} - \delta_{i^*+1} \leq$ negl(λ).*

Proof. The proof of this lemma follows directly from the Φ-hiding assumption. Suppose there exists a PPT adversary $\mathcal{A}$ such that can distinguish between the two hybrid distributions with non-negligible probability γ. We use $\mathcal{A}$ to construct a reduction algorithm $\mathcal{B}$ that breaks the Φ-hiding assumption for the $(i^*+1)^{th}$ prime p_{i^*+1} with advantage γ. The Φ-hiding challenger samples an RSA modulus N and sends to reduction algorithm $\mathcal{B}$. The reduction algorithm samples the key K by choosing parameters a, b, primes $\{e_{i,b}\}_{i,b}$ as in the hybrid distributions. It also chooses a random input $x \leftarrow \mathcal{X}$. It computes $y = (ax+b)\mathbf{e}_x$. (Here computation is done in an absolute sense, not as modular arithmetic.) Next, $\mathcal{B}$ samples a generator $g \leftarrow \mathbb{Z}_N^*$. (Here it actually samples $g \leftarrow \mathbb{Z}_N$ and aborts whenever $g \notin \mathbb{Z}_N^*$. This happens with only negligible probability.) It computes $\widetilde{g} = g^{\prod_{i=3}^{i^*+1} f_i}$. Next, it computes $\widetilde{h}$ as $\widetilde{h} = g^{y \prod_{i=1}^{i^*+1} f_i} \times h^{\prod_{i=1}^{j_\epsilon} f_i}$ where $h \leftarrow \mathbb{Z}_N^*$. Finally, it sends $(N, \widetilde{g}, K, \widetilde{h})$ to the adversary $\mathcal{A}$. If the adversary $\mathcal{A}$ outputs 1, then $\mathcal{B}$ guesses that p_{i^*+1} does not divide $\phi(N)$, else it guesses that $p_{i^*+1} \mid \phi(N)$.

Let us now analyze the advantage of the reduction algorithm $\mathcal{B}$. First, recall that $f_{i^*+1} = p_{i^*+1}^{e_{i^*+1}}$. Now if $p_{i^*+1} \nmid \phi(N)$, then the distributions $\{g \,:\, g \leftarrow \mathbb{Z}_N^*\}$ and $\left\{g^{f_{i^*+1}} \,:\, g \leftarrow \mathbb{Z}_N^*\right\}$ are identically distributed. Therefore, in this case, the reduction algorithm simulates hybrid distribution i^* for $\mathcal{A}$ (i.e., the distribution corresponding to δ_{i^*}). Otherwise, if $p_{i^*+1} \mid \phi(N)$, then $\mathcal{B}$ simulates hybrid distribution i^*+1 for $\mathcal{A}$. (Note that in both cases it perfectly simulates the element $\widetilde{h}$ as well. This is because note that the multiplicative term $h^{\prod_{i=1}^{j_\epsilon} f_i}$ for a random choice of h simply randomizes the congruent CRT representation corresponding to prime factors greater than p_{j_ϵ}. This is exactly what the distributions require, thus simulation is done perfectly.) Hence, if $\mathcal{A}$ distinguishes with non-negligible probability γ, then $\mathcal{B}$'s advantage in Φ-hiding is negligibly close to γ. Thus, the lemma follows. ∎

Lastly, to complete the proof of Theorem 3.4, we argue the following:

Observation 3.1. *For any PPT adversary $\mathcal{A}$,*

$$\Pr\left[\mathcal{A}(N, g^{\prod_{i=3}^{j_\epsilon} f_i}, K, h^{\prod_{i=1}^{j_\epsilon} f_i}) = 1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a, b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \mathrm{PRIMES}(\lambda) \\ g \leftarrow \mathbb{Z}_N^*;\; K = (a, b, \{e_{i,b}\}_{i,b});\; h \leftarrow \mathbb{Z}_N^* \end{array}\right] = \delta_{j_\epsilon},$$

where δ_{j_ϵ} is as defined in Eq. (4).

This follows directly from the definition of δ_{j_ϵ}. Note that the only difference in both the distributions is the way the element $\widetilde{h}$ is computed. To that end, look at the term $\widetilde{y} \prod_{i=1}^{i^*} f_i$ in the exponent of $\widetilde{h} = g^{\widetilde{y} \prod_{i=1}^{i^*} f_i}$. Also, observe that $\widetilde{y} \pmod{r_i^{k_i}}$ is sampled uniformly for all i such that $r_i > p_{j_\epsilon}$. Thus, sampling h at random and raising it to the exponent $\prod_{i=1}^{j_\epsilon} f_i$, is identical to sampling random exponents for the congruent CRT representation corresponding to prime factors greater than p_{j_ϵ} and raising it to the exponent $\prod_{i=1}^{j_\epsilon} f_i$.

Finally, combining above Observation 3.1 and Lemma 3.3, the Theorem 3.4 follows.

∎

Lastly, by combining Theorems 3.3 and 3.4, we obtain the proof of Theorem 3.2.

∎

3.2.1 Strengthening the Hash Lemma

In this section, we briefly provide a slight strengthening of the Theorem 3.2 where we argue that the indistinguishability holds even if the input $x \in \mathcal{X}$, instead of being sampled uniformly at random, is sampled from any arbitrary distribution with certain min-entropy. Formally, we prove the following.

Theorem 3.5. *Let p_i denote the i^{th} prime, i.e. $p_1 = 2, p_2 = 3, \ldots,$ and $\widetilde{e_i} = \lceil \log_{p_i} N \rceil$. And, let f_i denote $p_i^{\widetilde{e_i}}$ for all i.*

Assuming the Φ-hiding assumption holds, for every PPT adversary $\mathcal{A}$, non-negligible function $\epsilon(\cdot)$, polynomial $v(\cdot)$, for all $\lambda, \kappa \in \mathbb{N}$, satisfying $\kappa \geq 5\lambda$ and $\epsilon = \epsilon(\lambda) > 1/v(\lambda)$, and every (m, ℓ)-source $\mathcal{S}$ over $\mathcal{X}$

such that $\ell - m = O(\log\lambda)$, *the following holds,*

$$\Pr\left[\mathcal{A}(N,g,K,y)=1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ g \leftarrow \mathbb{Z}_N^*;\; K=(a,b,\{e_{i,b}\}_{i,b}); \\ x \leftarrow \mathcal{S};\; y = H_K(x)^{f_1 \cdot f_2} \end{array}\right]$$
$$-\Pr\left[\mathcal{A}(N,g,K,y)=1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ \widetilde{g} \leftarrow \mathbb{Z}_N^*;\; K=(a,b,\{e_{i,b}\}_{i,b}) \\ g = \widetilde{g}^{\prod_{i=3}^{j_\epsilon} f_i};\; h \leftarrow \mathbb{Z}_N^*;\; y = h^{\prod_{i=1}^{j_\epsilon} f_i} \end{array}\right] \leq \epsilon(\lambda)/2,$$

where j_ϵ *is the smallest index such that* $p_{j_\epsilon} > (2^{\ell-m+2}\log N/\epsilon)^4$.

Proof. The proof of the above theorem is nearly identical to the proof of Theorem 3.2, where we need to slightly adapt the statistical argument to account for the entropy loss. Here we briefly highlight the modifications necessary for proving the above theorem.

Similar to the proof of Theorem 3.2, we first prove the following statistical argument:

Theorem 3.6. *Let* p_i *denote the* i^{th} *prime, i.e.* $p_1 = 2, p_2 = 3, \ldots,$ *and* $\widetilde{e_i} = \lceil \log_{p_i} N \rceil$. *Also, for RSA modulus* N, *let* $\phi(N) = \prod r_i^{k_i}$, *where* $k_i \geq 1$, ℓ_N *denotes number of distinct prime factors of* $\phi(N)$, *and* r_i*'s are the distinct prime factors arranged in an increasing order.*

For every (possibly unbounded) adversary $\mathcal{A}$, *non-negligible function* $\epsilon(\cdot)$, *polynomial* $v(\cdot)$, *for all* $\lambda, \kappa \in \mathbb{N}$, *satisfying* $\kappa \geq 5\lambda$ *and* $\epsilon = \epsilon(\lambda) > 1/v(\lambda)$, *and every* (m,ℓ)*-source* $\mathcal{S}$ *over* $\mathcal{X}$ *such that* $\ell - m = O(\log\lambda)$, *the following holds,*

$$\Pr\left[\mathcal{A}(N,K,y)=1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K=(a,b,\{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{S};\; y = (ax+b)\cdot \mathbf{e}_x \;(\mathit{mod}\;\phi(N)) \end{array}\right]$$
$$-\Pr\left[\mathcal{A}(N,K,\widetilde{y})=1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K=(a,b,\{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{S};\; y = (ax+b)\cdot \mathbf{e}_x \;(\mathit{mod}\;\phi(N)) \\ y = (y^{(1)},\ldots,y^{(\ell_N)}) \text{ where } y^{(j)} = y \;(\mathit{mod}\; r_i^{k_i}) \\ \widetilde{y}^{(i)} = y^{(i)} \text{ for } i \text{ such that } r_i \leq p_{j_\epsilon} \\ \widetilde{y}^{(i)} \leftarrow \mathbb{Z}_{r_i^{k_i}} \text{ otherwise} \\ \widetilde{y} = (\widetilde{y}^{(1)},\ldots,\widetilde{y}^{(\ell_N)}) \end{array}\right] \leq \epsilon(\lambda)/3,$$

where j_ϵ *is the smallest index such that* $p_{j_\epsilon} > (2^{\ell-m+2}\log N/\epsilon)^4$.

Proof. The proof of above theorem is nearly identical to the proof of Theorem 3.3, with the following changes. Here the bad events as well as all the cases (and sub-cases) are identically defined. The difference is that first we need to prove an alternate version of Lemma 3.1, where the inputs x, y are now sampled as per $\mathcal{S}$ instead.

Lemma 3.4. *For every prime* $r > 3$, *integer* $k \geq 1$,

$$\Pr_{\substack{a,b,\{e_{i,b}\}_{i,b},\\ x,y \leftarrow \mathcal{S},\\ x \neq y}}\left[(ax+b)\cdot \mathbf{e}_x = (ay+b)\cdot \mathbf{e}_y \;(\mathit{mod}\; r^k)\right] \leq \frac{1}{r^k}\left(1 + \frac{2^{\ell-m+1}k}{\sqrt{r}} + \frac{2^{\ell-m} r^k k}{|\mathcal{X}|}\right).$$

Proof. The proof of this lemma is identical to that of Lemma 3.1, except the probability that any of the cases (1) to $(k+1)$ occur gets amplified by a multiplicative factor of $2^{\ell-m}$. This simply follows directly from the fact that the min-entropy of $\mathcal{S}$ is m and the length of inputs is ℓ bits. Concretely, now we will have the following:

$$\Pr_{\substack{x,y\leftarrow\mathcal{S},\\x\neq y}}[\text{Case }(1)] \leq 1 - \frac{2^{\ell-m}}{r} + \frac{2^{\ell-m}r}{|\mathcal{X}|},$$

$$\text{For } 1 < i < k+1, \qquad \Pr_{\substack{x,y\leftarrow\mathcal{S},\\x\neq y}}[\text{Case }(i)] \leq 2^{\ell-m}\left(\frac{1}{r^{i-1}} - \frac{1}{r^i} + \frac{r^i}{|\mathcal{X}|}\right),$$

$$\Pr_{\substack{x,y\leftarrow\mathcal{S},\\x\neq y}}[\text{Case }(k+1)] \leq \frac{2^{\ell-m}}{r^k}$$

Now the rest of analysis follows analogously where the only difference is that we have to take this extra multiplicative factor of $2^{\ell-m}$ into account in every equation. Thus, following the analysis instead of Eq. (3), we get the following:

$$\Pr_{\substack{a,b,\{e_{i,b}\}_{i,b},\\x,y\leftarrow\mathcal{S},\\x\neq y}}\left[(ax+b)\cdot\mathbf{e}_x = (ay+b)\cdot\mathbf{e}_y \pmod{r^k}\right] \leq \frac{1}{r^k}\left(1+\frac{2^{\ell-m+1}k}{\sqrt{r}}\right) + \frac{2^{\ell-m}k}{|\mathcal{X}|}. \tag{5}$$

This completes the proof of Lemma 3.4. ∎

Next, using Lemma 3.4, (as before) we can claim the following.

Lemma 3.5. *For every prime $r > 3$, integer $k \geq 1$, every (possibly unbounded) adversary $\mathcal{A}$, the following holds*

$$\Pr\left[\mathcal{A}(r^k, K, y_\beta) = \beta \ : \ \begin{array}{c} a,b\leftarrow\mathbb{Z}_{r^k}; e_{i,b}\leftarrow \text{PRIMES}(\lambda) \\ K = (a,b,\{e_{i,b}\}_{i,b});\quad x\leftarrow\mathcal{S} \\ y_0 = (ax+b)\cdot\mathbf{e}_x \ (mod\ r^k);\ y_1\leftarrow\mathbb{Z}_{r^k}\end{array}\right] \leq 2^{(\ell-m)/2}\left(\sqrt{\frac{k}{2\sqrt{r}}} + \sqrt{\frac{kr^k}{2|\mathcal{X}|}}\right).$$

Proof. The proof of this lemma is similar to that of Lemma 3.2. Concretely, when the input x is drawn as $x \leftarrow \mathcal{S}$. We can write the following. Let $\delta = \Pr_{\substack{a,b,\{e_{i,b}\}_{i,b},\\x,y\leftarrow\mathcal{S},\\x\neq y}}\left[(ax+b)\cdot\mathbf{e}_x = (ay+b)\cdot\mathbf{e}_y \pmod{r^k}\right]$.

$$\begin{aligned}\mathsf{SD}\left((H^{(r,k)}, H^{(r,k)}(X)), (H^{(r,k)}, U_{\mathbb{Z}_{r^k}})\right) &\leq \frac{1}{2}\sqrt{|\mathcal{K}^{(r,k)}|\cdot r^k}\sqrt{\frac{\delta}{|\mathcal{K}^{(r,k)}|} + \frac{2^{\ell-m}}{|\mathcal{X}|\cdot|\mathcal{K}^{(r,k)}|} - \frac{1}{|\mathcal{K}^{(r,k)}|\cdot r^k}} \\ &\leq \frac{1}{2}\sqrt{\delta\cdot r^k - 1 + \frac{2^{\ell-m}r^k}{|\mathcal{X}|}} \\ &\leq \frac{1}{2}\left(\sqrt{\delta\cdot r^k - 1} + \sqrt{\frac{2^{\ell-m}r^k}{|\mathcal{X}|}}\right)\end{aligned}$$

Using Lemma 3.4, we can simply it further to $\sqrt{\frac{2^{\ell-m}k}{2\sqrt{r}}} + \sqrt{\frac{2^{\ell-m}r^kk}{2|\mathcal{X}|}}$. This completes the proof of Lemma 3.5. ∎

Finally, using union bounds, congruence due to Chinese remainder theorem, and extending the analyses of Lemmas 3.4 and 3.5, we get the following

$$\Pr\left[\mathcal{A}(N,K,y)=1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K = (a,b,\{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{S} \\ y = (ax+b)\cdot \mathbf{e}_x \pmod{\phi(N)} \end{array}\right]$$
$$-\Pr\left[\mathcal{A}(N,K,\widetilde{y})=1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K = (a,b,\{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{S} \\ y = (ax+b)\cdot \mathbf{e}_x \pmod{\phi(N)} \\ y = (y^{(1)},\ldots,y^{(\ell_N)}) \text{ where } y^{(j)} = y \pmod{r_i^{k_i}} \\ \widetilde{y}^{(i)} = y^{(i)} \text{ for } i \text{ such that } r_i \le p_{j_\epsilon} \\ \widetilde{y}^{(i)} \leftarrow \mathbb{Z}_{r_i^{k_i}} \text{ otherwise} \\ \widetilde{y} = (\widetilde{y}^{(1)},\ldots,\widetilde{y}^{(\ell_N)}) \end{array}\right]$$
$$\le \Pr[\mathsf{Bad}] + \frac{2^{(\ell-m)/2}\ell_N}{\sqrt{2p_{j_\epsilon}^{1/2}}} + \sqrt{\frac{2^{\ell-m}\log N\cdot\phi(N)}{2|\mathcal{X}|}}$$
$$\le \text{negl}_1(\lambda) + \epsilon/4 + \text{negl}_2(\lambda) < \epsilon/3.$$

Here to argue that $\sqrt{\frac{2^{\ell-m}\log N\cdot\phi(N)}{2|\mathcal{X}|}}$ is negligible in λ, we use the fact that input domain $\mathcal{X} = \{0,1\}^\ell$ is quite large, i.e. $\ell > \kappa + 2\lambda$.

This completes the proof of the first part, which shows a tight bound on the statistical distance between the real and intermediate hybrid distribution. ■

Finally, to complete the proof of Theorem 3.5, we prove using the Φ-hiding assumption the computational part of the argument. Concretely, we show that following:

Theorem 3.7. *Let p_i denote the i^{th} prime, i.e. $p_1 = 2, p_2 = 3, \ldots,$ and $\widetilde{e_i} = \lceil \log_{p_i} N \rceil$ and let f_i denote $p_i^{\widetilde{e_i}}$ for all i. Also, for RSA modulus N, let $\phi(N) = \prod r_i^{k_i}$, where $k_i \ge 1$, ℓ_N denotes number of distinct prime factors of $\phi(N)$, and r_i's are the distinct prime factors arranged in an increasing order.*

Assuming the Φ-hiding assumption holds, for every PPT adversary $\mathcal{A}$, non-negligible function $\epsilon(\cdot)$, polynomial $v(\cdot)$, for all $\lambda, \kappa \in \mathbb{N}$, satisfying $\kappa \ge 5\lambda$ and $\epsilon = \epsilon(\lambda) > 1/v(\lambda)$, and every (m,ℓ)-source $\mathcal{S}$ over $\mathcal{X}$ such that $\ell - m = O(\log \lambda)$, there exists a negligible function negl$(\cdot)$ such that the following holds,

$$\Pr\left[\mathcal{A}(N,g,K,g^{\widetilde{y}f_1\cdot f_2})=1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a,b \leftarrow \mathbb{Z}_N; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K = (a,b,\{e_{i,b}\}_{i,b});\quad x \leftarrow \mathcal{S};\; g \leftarrow \mathbb{Z}_N^* \\ y = (ax+b)\cdot \mathbf{e}_x \;(\mathit{mod}\; \phi(N)) \\ y = (y^{(1)},\ldots,y^{(\ell_N)}) \textit{ where } y^{(j)} = y \;(\mathit{mod}\; r_i^{k_i}) \\ \widetilde{y}^{(i)} = y^{(i)} \textit{ for } i \textit{ such that } r_i \le p_{j_\epsilon} \\ \widetilde{y}^{(i)} \leftarrow \mathbb{Z}_{r_i^{k_i}} \textit{ otherwise} \\ \widetilde{y} = (\widetilde{y}^{(1)},\ldots,\widetilde{y}^{(\ell_N)}) \end{array}\right]$$
$$-\Pr\left[\mathcal{A}(N,g^{\prod_{i=3}^{j_\epsilon} f_i},K,h^{\prod_{i=1}^{j_\epsilon} f_i})=1 \;:\; \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\; a,b \leftarrow \mathbb{Z}_N \\ g \leftarrow \mathbb{Z}_N^*;\; e_{i,b} \leftarrow \text{PRIMES}(\lambda) \\ K = (a,b,\{e_{i,b}\}_{i,b});\; h \leftarrow \mathbb{Z}_N^* \end{array}\right] \le \mathit{negl}(\lambda),$$

where j_ϵ is the smallest index such that $p_{j_\epsilon} > (2^{\ell-m+2}\log N/\epsilon)^4$.

The proof of above theorem is identical to that of Theorem 3.4, and follows via a sequence of hybrids. Lastly, by combining Theorems 3.6 and 3.7, Theorem 3.5 follows.

■

3.3 Φ-Hiding based Extractor Lemma

In this section we prove a useful lemma that will aid in proving security of our Φ-hiding based constructions later. This has appeared (and implicity used) in most existing Φ-hiding based works. Here we abstract it out for ease of exposition.

Let $\mathsf{Ext} : \mathbb{Z}_N \times \mathbb{S} \to \mathcal{Y}$ be a $(\lambda - 1, \epsilon)$ strong extractor, where ϵ is negligible in the parameter λ. Informally, the lemmas states that, for every λ-bit prime e, applying extractor on an e^{th} root of a generator $g \in \mathbb{Z}_N^*$ is indistinguishable from random. Formally, we claim the following:

Lemma 3.6. *Assuming the Φ-hiding assumption holds, then for every admissible stateful PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda, \kappa \in \mathbb{N}$, such that $\kappa \geq 5\lambda$, the following hold,*

$$\Pr\left[\mathcal{A}(y_b) = b \ : \ \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\ \mathfrak{s} \leftarrow \mathbb{S} \\ e \leftarrow \mathrm{PRIMES}(\lambda);\ g \leftarrow \mathbb{Z}_N^* \\ F \leftarrow \mathcal{A}(N, \mathfrak{s}, e, g);\ b \leftarrow \{0,1\} \\ y_0 = \mathsf{Ext}(g^{F/e}, \mathfrak{s});\ y_1 \leftarrow \mathcal{Y} \end{array}\right] \leq \mathit{negl}(\lambda),$$

where $\mathcal{A}$ is an admissible adversary as long as $e \nmid F$.

Proof. The proof of this lemma follows a simple sequence of hybrids. First, using Φ-hiding we can indistinguishably switch to sampling (e, N) such that $e \mid \phi(N)$. Once we have that $e \mid \phi(N)$, we know that there will exist e e^{th}-roots of generator g with all but negligible probability. Thus, using the strong extractor guarantee, we get that y_0 is indistinguishable from random.

Claim 3.1. *Assuming the Φ-hiding assumption holds, then for every admissible stateful PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda, \kappa \in \mathbb{N}$, such that $\kappa \geq 5\lambda$, the following holds*

$$\Pr\left[\mathcal{A}(y_b) = b \ : \ \begin{array}{c} N \leftarrow \mathsf{RSA}(\kappa);\ \mathfrak{s} \leftarrow \mathbb{S} \\ e \leftarrow \mathrm{PRIMES}(\lambda);\ g \leftarrow \mathbb{Z}_N^* \\ F \leftarrow \mathcal{A}(N, \mathfrak{s}, e, g);\ b \leftarrow \{0,1\} \\ y_0 = \mathsf{Ext}(g^{F/e}, \mathfrak{s});\ y_1 \leftarrow \mathcal{Y} \end{array}\right]$$
$$- \Pr\left[\mathcal{A}(y_b) = b \ : \ \begin{array}{c} e \leftarrow \mathrm{PRIMES}(\lambda);\ \mathfrak{s} \leftarrow \mathbb{S} \\ N \leftarrow \mathsf{RSA}_e(\kappa);\ h \leftarrow \mathbb{Z}_N^* \\ F \leftarrow \mathcal{A}(N, \mathfrak{s}, e, h^e);\ b \leftarrow \{0,1\} \\ y_0 = \mathsf{Ext}(h^F, \mathfrak{s});\ y_1 \leftarrow \mathcal{Y} \end{array}\right] \leq \mathit{negl}(\lambda),$$

where $\mathcal{A}$ is an admissible adversary as long as $e \nmid F$.

Proof. The proof of this lemma follows directly from the Φ-hiding assumption. Suppose there exists a PPT adversary $\mathcal{A}$ such that can distinguish between the two hybrid distributions with non-negligible probability γ. We use $\mathcal{A}$ to construct a reduction algorithm $\mathcal{B}$ that breaks the Φ-hiding assumption. Let e be a randomly chosen λ-bit prime. The Φ-hiding challenger samples an RSA modulus N and sends to reduction algorithm $\mathcal{B}$. Given inputs N, e, the reduction algorithm samples a random seed $\mathfrak{s}$, and element $h \leftarrow \mathbb{Z}_N^*$. (Here it actually samples $h \leftarrow \mathbb{Z}_N$ and aborts whenever $h \notin \mathbb{Z}_N^*$. This happens with only negligible probability.) Next, it computes generator as $g = h^e$, and sends parameters $(N, \mathfrak{s}, e, g)$ to $\mathcal{A}$. The adversary $\mathcal{A}$ sends an integer F to $\mathcal{B}$, and $\mathcal{B}$ first samples a random bit b, output $y_1 \leftarrow \mathcal{Y}$, and computes $y_0 = \mathsf{Ext}(h^F, \mathfrak{s})$. $\mathcal{B}$ sends y_b as the challenge to the adversary $\mathcal{A}$. If the adversary $\mathcal{A}$ outputs b, then $\mathcal{B}$ guesses that $e \nmid \phi(N)$, else it guesses that $e \mid \phi(N)$.

Let us now analyze the advantage of the reduction algorithm $\mathcal{B}$. Note that if $e \nmid \phi(N)$, then the distributions $\left\{(g, g^{F/e}) \ : \ g \leftarrow \mathbb{Z}_N^*\right\}$ and $\left\{(h^e, h^F) \ : \ h \leftarrow \mathbb{Z}_N^*\right\}$ are identically distributed as long as $e \nmid F$. Therefore, the reduction algorithm perfectly simulates the hybrid distributions for $\mathcal{A}$ depending on whether $e \mid \phi(N)$ or not. Hence, if $\mathcal{A}$ distinguishes with non-negligible probability γ, then $\mathcal{B}$'s advantage in Φ-hiding is also γ. Thus, the claim follows. ■

Claim 3.2. *If* Ext *is a* $(\lambda - 1, \epsilon)$ *strong extractor, where* ϵ *is negligible in the parameter* λ*, then for every admissible stateful adversary* $\mathcal{A}$*, there exists a negligible function* $negl(\cdot)$ *such that for all* $\lambda, \kappa \in \mathbb{N}$*, such that* $\kappa \geq 5\lambda$*, the following holds*

$$\Pr\left[\mathcal{A}(y_b) = b \ : \ \begin{array}{c} e \leftarrow \mathrm{PRIMES}(\lambda);\ \mathfrak{s} \leftarrow \mathbb{S} \\ N \leftarrow \mathsf{RSA}_e(\kappa);\ h \leftarrow \mathbb{Z}_N^* \\ F \leftarrow \mathcal{A}(N, \mathfrak{s}, e, h^e);\ b \leftarrow \{0,1\} \\ y_0 = \mathsf{Ext}(h^F, \mathfrak{s});\ y_1 \leftarrow \mathcal{Y} \end{array}\right] \leq negl(\lambda),$$

where $\mathcal{A}$ *is an admissible adversary as long as* $e \nmid F$*.*

Proof. This follows directly from the strong extractor guarantee of Ext. The proof relies on the fact that for any given element $h \in \mathbb{Z}_N^*$, there exists $e - 1$ other distinct elements $\widetilde{h} \in \mathbb{Z}_N^*$, such that $h^e = \widetilde{h}^e$ when $e \mid \phi(N)$. Concretely, for any $h \in \mathbb{Z}_N^*$, prime e, and integer F such that $e \nmid F$, let $S_{e,h}$ and $T_{e,h,F}$ denote the following sets:

$$S_{e,h} = \left\{\widetilde{h} \in \mathbb{Z}_N^* \ : \ h^e = \widetilde{h}^e\right\}, \qquad T_{e,h,F} = \left\{\widetilde{h} \in \mathbb{Z}_N^* \ : \ \widetilde{h} = g^F, g \in S_{e,h}\right\}.$$

Let $\mathcal{D}_{e,h,F}$ denote the uniform distribution over $T_{e,h,F}$. Note that $\mathcal{D}_{e,h,F}$ has min-entropy $\log_2 e > \lambda - 1$, since e is a λ-bit prime. Therefore, by extractor security the claim follows since ϵ is negligible. ∎

From Claims 3.1 and 3.2, the lemma follows. ∎

4 Hinting PRGs based on Φ-Hiding

In this section, we provide our (n, ℓ)-hinting PRG (HPRG) construction based on the Φ-hiding assumption. For an RSA modulus N and parameters λ, ℓ, let $\mathsf{Ext} : \mathbb{Z}_N \times \mathbb{S} \rightarrow \{0,1\}^\ell$ be a $(\lambda - 1, \epsilon_{\mathsf{ext}})$ strong extractor, where ϵ_{ext} is negligible in the parameter λ.[6] Below we describe our construction.

$\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{pp}, n)$. The setup algorithm takes as input the security parameter λ. It sets RSA modulus bit length $\kappa = 5\lambda$ and number of blocks $n = \kappa + 2\lambda$.

Next, it samples modulus N as $N \leftarrow \mathsf{RSA}(\kappa)$, seed $\mathfrak{s} \leftarrow \mathbb{S}$, generator $g \leftarrow \mathbb{Z}_N^*$, $2n$ (λ-bit) primes $\{e_{i,b}\}_{i,b}$ as $e_{i,b} \leftarrow \mathrm{PRIMES}(\lambda)$ for $(i, b) \in [n] \times \{0,1\}$, and elements $a, b \leftarrow \mathbb{Z}_N$. Finally, it outputs the public parameters pp as $\mathsf{pp} = \left(N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, a, b\right)$.

$\mathsf{Eval}(\mathsf{pp}, x, j) \rightarrow y$. Let $\mathsf{pp} = \left(N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, a, b\right)$. The evaluation algorithm proceeds as follows. Let $\widetilde{e_1} = \lceil \log_2 N \rceil$, $\widetilde{e_2} = \lceil \log_3 N \rceil$, and $f_1 = 2^{\widetilde{e_1}}$, $f_2 = 3^{\widetilde{e_2}}$.

It computes $h = g^{f_1 \cdot f_2 \cdot (ax+b) \prod_{i \in [n] \setminus \{j\}} e_{i,x_i}} \pmod N$, and outputs $y = \mathsf{Ext}(h, \mathfrak{s})$.

4.1 Optimization by Sharing Computation

A naïve method of computing $\mathsf{Eval}(\mathsf{pp}, x, j)$ for all indices j involves $O(n^2)$ exponentations. However, a significant amount of these exponentiations are redundant. This is because for any indices $j_1 \neq j_2$, most of the operations involved in computing $\mathsf{Eval}(\mathsf{pp}, x, j_1)$ and $\mathsf{Eval}(\mathsf{pp}, x, j_2)$ are same. We now describe an efficient procedure that computes $\mathsf{Eval}(\mathsf{pp}, x, j)$ for all indices j using only $O(n \log n)$ exponentiations. We use the technique of dynamic programming to achieve the optimization. Consider the recursive procedure described in Algorithm 1. The procedure takes as input a base generator h, an integer n, a list of exponents exp of length n and RSA modulus N. It outputs a list $[h^{\prod_{i \neq j} \mathsf{exp}_i} \bmod N]_{j \in [n]}$. In order to compute

[6] Note that such an extractor exists when $\ell < c\lambda$ for any fixed constant c. The Hinting PRG construction can be extended for $\ell \geq \lambda$ by using standard PRG.

$\mathsf{Eval}(\mathsf{pp}, x, j)$ for all j, we invoke the procedure $\mathsf{ComputeHPRG}$ on generator $h = g^{f_1 \cdot f_2 \cdot (ax+b)}$ and exponents $\mathsf{exp} = [e_{i,x_i}]_{i \in [n]}$. The procedure $\mathsf{ComputeHPRG}$ divides the list of exponents into groups of 2, multiplies the exponents in each group and obtains the list subexp. It then recurses on the smaller list of exponents subexp, and then uses its output to compute $[h^{\prod_{i \neq j} \mathsf{exp}_i} \bmod N]_{j \in [n]}$. Let us now analyze the total time taken by the algorithm. Initially the $\mathsf{ComputeHPRG}$ procedure is invoked on the list exp consisting of n λ-bit exponents $[e_{i,x_i}]_{i \in [n]}$. The procedure involves n exponentiations with λ bit exponents (Lines 8-10) and calls the $\mathsf{ComputeHPRG}$ procedure recursively on $n/2$ 2λ-bit exponents subexp (Multiplying numbers as in Line 5 is asymptotically faster than exponentiation and therefore we ignore the time required for this operation for the sake of simplicity). Similarly, the i^{th} recursively call to the $\mathsf{ComputeHPRG}$ procedure involves $n/2^i$ exponentiations with $2^i \cdot \lambda$ bit exponents. As the computational effort required for this is same as the performing n exponentiations with λ bit exponents, and as there can be at most $\log n$ recursive calls, the algorithm totally involves $O(n \log n)$ exponentiations with λ bit exponents.

Algorithm 1 Recursive procedure for computing HPRG

1: **procedure** $\textsc{ComputeHPRG}(\textsc{generator}\ h,\ \textsc{int}\ n,\ \textsc{list}\ \mathsf{exp},\ \textsc{int}\ N)$
2:     **if** $n = 1$ **then return** $[h]$
3:     **else if** $n = 2$ **then return** $[h^{\mathsf{exp}_2} \bmod N,\ h^{\mathsf{exp}_1} \bmod N]$
4:     **else**
5:         $\mathsf{subexp} \leftarrow [\mathsf{exp}_{2 \cdot i - 1} \cdot \mathsf{exp}_{2 \cdot i}]_{1 \leq i \leq \lfloor n/2 \rfloor}$
6:         **if** $n \bmod 2 = 1$ **then** $\mathsf{subexp} \leftarrow \mathsf{subexp} || \mathsf{exp}_n$
7:         $\mathsf{suboutput} \leftarrow \mathrm{ComputeHPRG}(h, \lceil n/2 \rceil, \mathsf{subexp})$
8:         **for** $1 \leq i \leq 2 \cdot \lfloor n/2 \rfloor$ **do**
9:             **if** $i \bmod 2 = 0$ **then** $\mathsf{output}_i = \mathsf{suboutput}_{\lceil i/2 \rceil}^{\mathsf{exp}_{i-1}} \bmod N$
10:             **else if** $i \bmod 2 = 1$ **then** $\mathsf{output}_i = \mathsf{suboutput}_{\lceil i/2 \rceil}^{\mathsf{exp}_{i+1}} \bmod N$
11:         **if** $n \bmod 2 = 1$ **then** $\mathsf{output} \leftarrow \mathsf{output} || \mathsf{suboutput}_{\lceil n/2 \rceil}$
        return output

4.2 Security

Now we prove that the construction described above is a secure HPRG. Formally, we prove the following.

Theorem 4.1. *If the Φ-hiding assumption (Assumption 1) holds, then the HPRG construction described above is secure as per Definition 2.4.*

Proof. First, we introduce some useful notations. For any sequence $\{e_{i,b}\}_{i,b}$ and string $x \in \{0,1\}^n$, we use $\mathbf{e}_x$ as a shorthand for the subset product $\prod_{i=1}^{n} e_{i,x_i}$. Let p_i denote the i^{th} (smallest) prime, i.e. $p_1 = 2, p_2 = 3, \ldots$, and $\widetilde{e_i} = \lceil \log_{p_i} N \rceil$ for all i. And, let f_i denote $p_i^{\widetilde{e_i}}$ for all i. For any constant $\epsilon > 0$, let j_ϵ be the smallest index such that $p_{j_\epsilon} > (2\sqrt{2} \log N/\epsilon)^4$. Now we describe our proof.

The proof of security follows via a sequence of hybrids. Below we first describe the sequence of hybrids, and later argue indistinguishability to complete the proof. At a very high level, the proof structure is somewhat similar to that used in [Zha16], where for proving security one first assumes (for the sake of contradiction) that the adversary wins with some non-negligible probability δ and then depending upon δ, one could describe a sequence of hybrids such that no PPT adversary can win with probability more than δ. This acts as a contradiction, thereby completing the proof.

Intuitively, the main idea is to first switch the randomly sampled terms $y^0_{j,1-x_j}$ to be instead sampled in a very structured way where $y^0_{j,1-x_j} = \mathsf{Ext}(h_{j,1-x_j}, \mathfrak{s})$ and $h_{j,1-x_j} = g^{f_1 \cdot f_2 \cdot (ax+b) e^{-1}_{j,1-x_j} \prod_{i \in [n]} e_{i,x_i}}$. This follows from the Φ-hiding based extractor lemma (Lemma 3.6). Next, using our new hashing lemma (Theorem 3.2), we now instead of computing $g^{f_1 \cdot f_2 \cdot (ax+b) \prod_{i \in [n]} e_{i,x_i}}$, sample $z \leftarrow \mathbb{Z}_N^*$ and compute $z^{\prod_{k=1}^{j_\epsilon} f_k}$. In the next hybrid, we replace challenge with random elements in $\mathbb{Z}_N^*$ using the Φ-hiding based extractor lemma

(Lemma 3.6). Concretely, Hybrid 1 corresponds to HPRG game where the challenger always chooses $\beta = 0$, i.e. samples half of the challenge matrix as appropriate functions of the seed x and remaining randomly. Hybrid 4 corresponds to HPRG game where the challenger always chooses $\beta = 1$, i.e. challenge matrix consists of random entries.

For any PPT adversary $\mathcal{A}$, let the probability that $\mathcal{A}$ outputs 1 in Hybrid t be denoted as $p_t^{\mathcal{A}}$. For the sake of contradiction, we assume that the adversary $\mathcal{A}$ wins with non-negligible probability $\delta(\lambda)$, which suggests that $p_1^{\mathcal{A}} - p_4^{\mathcal{A}} = \delta(\lambda)$. This means that there exists a polynomial $v(\cdot)$ such that $\delta(\lambda) \geq 1/v(\lambda)$ infinitely often for $\lambda \in \mathbb{N}$. Let $\epsilon(\lambda) = 1/v(\lambda)$. We will drop the dependence on λ whenever clear from context.[7]

Hybrid 1. This is same as the original HPRG game, where the challenger always chooses $\beta = 0$.

1. The challenger sets $\kappa = 5\lambda$, $n = \kappa + 2\lambda$. It samples the public parameters $\mathsf{pp} = \left(N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, a, b\right)$ as:

$$N \leftarrow \mathsf{RSA}(\kappa), \quad a, b \in \mathbb{Z}_N, \quad g \in \mathbb{Z}_N^*, \quad \mathfrak{s} \leftarrow \mathbb{S}, \quad e_{i,b} \leftarrow \text{PRIMES}(\lambda)\ (\forall\ (i,b) \in [n] \times \{0,1\}).$$

2. Next, it samples a random HPRG seed $x \leftarrow \{0,1\}^n$, computes the HPRG challenge $(y_0, \{y_{i,b}\}_{i,b})$ as:

$$\begin{aligned} & y_0 = \mathsf{Ext}(g^{f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x}, \mathfrak{s}), \\ \forall\ i \in [n], \quad & y_{i,x_i} = \mathsf{Ext}(g^{f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x e_{i,x_i}^{-1}}, \mathfrak{s}), \\ \forall\ i \in [n], \quad & y_{i,1-x_i} \leftarrow \{0,1\}^{\ell}. \end{aligned}$$

3. The challenger sends public parameters pp, n and the challenge $(y_0, \{y_{i,b}\}_{i,b})$ to the adversary. Finally, the adversary outputs a bit β'.

Hybrid $1.i^*$ ($i^* \in [n]$). This hybrid is same as hybrid 1, except that the challenger chooses $y_{i,1-x_i}$ in a structured way for all $i \leq i^*$.

2. Next, it samples a random HPRG seed $x \leftarrow \{0,1\}^n$, computes the HPRG challenge $(y_0, \{y_{i,b}\}_{i,b})$ as:

$$\begin{aligned} & y_0 = \mathsf{Ext}(g^{f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x}, \mathfrak{s}), \\ \forall\ i \in [n], \quad & y_{i,x_i} = \mathsf{Ext}(g^{f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x e_{i,x_i}^{-1}}, \mathfrak{s}), \\ \forall\ i \in [i^*], \quad & y_{i,1-x_i} = \mathsf{Ext}(g^{f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x e_{i,1-x_i}^{-1}}, \mathfrak{s}), \\ \forall\ i \in [n] \setminus [i^*], \quad & y_{i,1-x_i} \leftarrow \{0,1\}^{\ell}. \end{aligned}$$

Note that the challenger knows $\phi(N)$, thus it can compute all the necessary inverses.

Hybrid 2. This hybrid is similar to Hybrid $1.n$, except none of the challenge terms have any dependence on the HPRG seed x.

1. The challenger sets $\kappa = 5\lambda$, $n = \kappa + 2\lambda$. It samples the public parameters $\mathsf{pp} = \left(N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, a, b\right)$ as:

$$N \leftarrow \mathsf{RSA}(\kappa), \quad a, b \in \mathbb{Z}_N, \quad \mathfrak{s} \leftarrow \mathbb{S}, \quad e_{i,b} \leftarrow \text{PRIMES}(\lambda)\ (\forall\ (i,b) \in [n] \times \{0,1\}),$$
$$h \in \mathbb{Z}_N^*, \quad g = h^{\prod_{k=3}^{j_\epsilon} f_k}.$$

[7]Note that, throughout the analysis, we provide a non-uniform reduction where the description of hybrids and the reduction algorithm depends upon ϵ. Note that one could instead avoid such a non-uniform choice by first running the adversary sufficiently many times to estimate the advantage ϵ as $\widetilde{\epsilon}$, and later on use the estimated advantage $\widetilde{\epsilon}$ instead. Therefore, for ease of exposition, we simply give a non-uniform reduction.

2. Next, it samples a random HPRG seed $x \leftarrow \{0,1\}^n$, computes the HPRG challenge $(y_0, \{y_{i,b}\}_{i,b})$ as:

$$\begin{aligned} & z \leftarrow \mathbb{Z}_N^*, \\ & y_0 = \mathsf{Ext}(z^{\prod_{k=1}^{j_\epsilon} f_k}, \mathfrak{s}), \\ \forall\ i \in [n], b \in \{0,1\}, \quad & y_{i,b} = \mathsf{Ext}(z^{e_{i,b}^{-1} \prod_{k=1}^{j_\epsilon} f_k}, \mathfrak{s}). \end{aligned}$$

Hybrid $3.i^*.b^*$ $(i^* \in [n], b^* \in \{0,1\})$. This hybrid is similar to Hybrid 2, except that the challenger chooses $y_{i,b}$ randomly for all $(i,b) \preceq (i^*, b^*)$.[8]

2. Next, it samples a random HPRG seed $x \leftarrow \{0,1\}^n$, computes the HPRG challenge $(y_0, \{y_{i,b}\}_{i,b})$ as:

$$\begin{aligned} & z \leftarrow \mathbb{Z}_N^*, \\ & y_0 = \mathsf{Ext}(z^{\prod_{k=1}^{j_\epsilon} f_k}, \mathfrak{s}), \\ \forall\ (i,b) \preceq (i^*, b^*), \quad & y_{i,b} \leftarrow \{0,1\}^\ell, \\ \forall\ (i,b) \succ (i^*, b^*), \quad & y_{i,b} = \mathsf{Ext}(z^{e_{i,b}^{-1} \prod_{k=1}^{j_\epsilon} f_k}, \mathfrak{s}). \end{aligned}$$

Hybrid 4. This hybrid is similar to Hybrid $3.n.1$ except that y_0 is also generated randomly.

2. Next, it samples a random HPRG seed $x \leftarrow \{0,1\}^n$, computes the HPRG challenge $(y_0, \{y_{i,b}\}_{i,b})$ as:

$$\begin{aligned} & y_0 \leftarrow \{0,1\}^\ell, \\ \forall\ i \in [n], b \in \{0,1\}, \quad & y_{i,b} \leftarrow \{0,1\}^\ell. \end{aligned}$$

Now we argue indistinguishability between the hybrids described above. Below we use Hybrid 1.0 to correspond to Hybrid 1, and Hybrid 3.0.1 to correspond to Hybrid 2.

Lemma 4.1. *Assuming the Φ-hiding assumption holds, then for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $i^* \in [n]$, $p^{\mathcal{A}}_{1.(i^*-1)} - p^{\mathcal{A}}_{1.i^*} \leq$ negl(λ).*

Proof. The proof of this lemma follows directly from our Φ-hiding based extractor lemma (Lemma 3.6). Suppose that $\mathcal{A}$ distinguishes between Hybrids $1.(i^*-1)$ and $1.i^*$ with non-negligible probability γ, for some $i^* \in [n]$. We use $\mathcal{A}$ to build a reduction algorithm $\mathcal{B}$ that violates our Φ-hiding based extractor lemma, thereby leading us to a contradiction. Below we provide more details.

The reduction algorithm $\mathcal{B}$ first receives the parameters $(N, \mathfrak{s}, e, \widetilde{g})$ where e is a λ-bit prime and $\widetilde{g}$ is a random element in $\mathbb{Z}_N^*$. $\mathcal{B}$ then samples parameters a, b and HPRG seed x randomly as in Hybrid 1. It also samples primes $e_{i,b}$ for all $(i,b) \neq (i^*, 1-x_{i^*})$ as in Hybrid 1. It sets $e_{i^*,1-x_{i^*}}, g$ as $e_{i^*,1-x_{i^*}} = e$ and $g = \widetilde{g}^{\prod_{i<i^*} e_{i,1-x_i}}$. Now it sets the public parameters pp as $\mathsf{pp} = \left(N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, a, b\right)$. Next, $\mathcal{B}$ sets exponent F as $F = f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x$. If $e \mid F$, then $\mathcal{B}$ aborts and guesses randomly, otherwise it sends F to the challenger and continues. The challenger responds with a challenge bit string y, and then $\mathcal{B}$ computes the HPRG challenge $(y_0, \{y_{i,b}\}_{i,b})$ as:

$$\begin{aligned} & y_0 = \mathsf{Ext}(g^{f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x}, \mathfrak{s}), \\ \forall\ i \in [n], \quad & y_{i,x_i} = \mathsf{Ext}(g^{f_1 \cdot f_2 \cdot (ax+b) \prod_{j \neq i} e_{j,x_j}}, \mathfrak{s}), \\ \forall\ i \in [i^*-1], \quad & y_{i,1-x_i} = \mathsf{Ext}(\widetilde{g}^{f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x \prod_{j<i^* \wedge j \neq i} e_{j,1-x_j}}, \mathfrak{s}), \\ & y_{i^*,1-x_{i^*}} = y, \\ \forall\ i \in [n] \setminus [i^*], \quad & y_{i,1-x_i} \leftarrow \{0,1\}^\ell. \end{aligned}$$

[8]Here and throughout this section, $\preceq$ is a shorthand for the following relation: $(i,b) \preceq (i^*, b^*) \equiv i < i^* \vee (i = i^* \wedge b \leq b^*)$. And, $\succ$ is analogously defined, but as the converse of $\preceq$.

Finally, $\mathcal{B}$ sends pp, n and $(y_0, \{y_{i,b}\}_{i,b})$ to $\mathcal{A}$, and $\mathcal{A}$ outputs its guess β'. If $\beta' = 1$, then $\mathcal{B}$ outputs 0 (i.e., y is computed honestly) as its guess, otherwise it outputs 1 (i.e., y is random bit string) as its guess.

Let us now analyze the advantage of the reduction algorithm $\mathcal{B}$. First, note that $\mathcal{B}$ samples a, b, x and $e_{i,b}$ for $(i, b) \neq (i^*, 1 - x_{i^*})$ randomly (from appropriate distributions). Therefore, we have that the event $e \mid F$ occurs with only negligible probability since e is a random λ-bit prime, thus $\mathcal{B}$ aborts with atmost negligible probability. Second, since $e_{i,1-x_i}$ are randomly drawn λ-bit primes for $i \neq i^*$, thus sampling g as $g = \widetilde{g}^{\prod_{i<i^*} e_{i,1-x_i}}, \widetilde{g} \leftarrow \mathbb{Z}_N^*$ instead of $g \leftarrow \mathbb{Z}_N^*$ is statistically indistinguishable. Therefore, $\mathcal{B}$ simulates one of hybrids $1.(i^*-1)$ and $1.i^*$ (in a statistically indistinguishable way) depending upon whether the challenge y is computed as $y = \mathsf{Ext}(\widetilde{g}^{F/e}, \mathfrak{s})$ or $y \leftarrow \{0,1\}^\ell$. Hence, if $\mathcal{A}$ distinguishes with non-negligible probability γ, then $\mathcal{B}$'s advantage is also negligibly close to γ. Thus, the lemma follows. ■

Lemma 4.2. *Assuming the Φ-hiding assumption holds, then for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, satisfying $\delta(\lambda) \geq \epsilon(\lambda) = 1/v(\lambda)$, $p^{\mathcal{A}}_{1.n} - p^{\mathcal{A}}_2 \leq \epsilon(\lambda)/2 +$ negl(λ).*

Proof. The proof of this lemma follows directly from our new hashing lemma (Theorem 3.2). Suppose that $\mathcal{A}$ distinguishes between Hybrids $1.n$ and 2 with probability $\epsilon/2 + \gamma$, where γ is non-negligible. We use $\mathcal{A}$ to build a reduction algorithm $\mathcal{B}$ that violates the security requirement of our new hash lemma, thereby leading us to a contradiction. Below we provide more details.

The reduction algorithm $\mathcal{B}$ first receives the parameters $(N, \widetilde{g}, K, y)$, where $K = (a, b, \{e_{i,b}\}_{i,b})$. $\mathcal{B}$ then samples an extractor seed $\mathfrak{s}$ randomly. It computes exponent $E = \prod_{i,b} e_{i,b}$ and generator $g = \widetilde{g}^E$, and it sets the public parameters pp as $\mathsf{pp} = \left(N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, a, b\right)$. Next, $\mathcal{B}$ computes the HPRG challenge $(y_0, \{y_{i,b}\}_{i,b})$ as:

$$\begin{aligned} y_0 &= \mathsf{Ext}(y^E, \mathfrak{s}), \\ \forall\ i \in [n], b \in \{0,1\}, \quad y_{i,b} &= \mathsf{Ext}(y^{\prod_{(j,c)\neq(i,b)} e_{j,c}}, \mathfrak{s}). \end{aligned}$$

Finally, $\mathcal{B}$ sends pp, n and $(y_0, \{y_{i,b}\}_{i,b})$ to $\mathcal{A}$, and $\mathcal{A}$ outputs its guess β'. If $\beta' = 1$, then $\mathcal{B}$ outputs 0 (i.e., y is computed honestly) as its guess, otherwise it outputs 1 (i.e., y is random bit string) as its guess.

Let us now analyze the advantage of the reduction algorithm $\mathcal{B}$. First, note that the challenger samples all the primes $e_{i,b}$'s randomly, thus we have that with all-but-negligible probability, all $e_{i,b}$'s are co-prime w.r.t. $\phi(N)$. Therefore, sampling g as $g = \widetilde{g}^E, \widetilde{g} \leftarrow \mathbb{Z}_N^*$ instead of $g \leftarrow \mathbb{Z}_N^*$ is statistically indistinguishable. Similarly, sampling g as $g = \widetilde{g}^E, \widetilde{g} = h^{\prod_{k=3}^{j_\epsilon} f_k}, h \leftarrow \mathbb{Z}_N^*$ instead of $g = \widetilde{g}^{\prod_{k=3}^{j_\epsilon} f_k}, \widetilde{g} \leftarrow \mathbb{Z}_N^*$ is statistically indistinguishable. Therefore, $\mathcal{B}$ simulates one of hybrids $1.n$ and 2 (in a statistically indistinguishable way) depending upon whether the challenge y is computed as $y = \widetilde{g}^{f_1 \cdot f_2 \cdot (ax+b)\mathbf{e}_x}$ for a random x, or $y = z^{\prod_{k=1}^{j_\epsilon} f_k}$ for a random $z \in \mathbb{Z}_N^*$. Hence, if $\mathcal{A}$ distinguishes with non-negligible probability $\epsilon/2 + \gamma$, then $\mathcal{B}$'s advantage is also negligibly close to $\epsilon/2 + \gamma$. Thus, the lemma follows. ■

Lemma 4.3. *Assuming the Φ-hiding assumption holds, then for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $i^* \in [n]$, $p^{\mathcal{A}}_{3.i^*.0} - p^{\mathcal{A}}_{3.i^*.1} \leq$ negl(λ).*

Proof. The proof of this lemma follows directly from our Φ-hiding based extractor lemma (Lemma 3.6), and is quite similar to the proof of Lemma 4.1. Suppose that $\mathcal{A}$ distinguishes between Hybrids $3.i^*.0$ and $3.i^*.1$ with non-negligible probability γ, for some $i^* \in [n]$. We use $\mathcal{A}$ to build a reduction algorithm $\mathcal{B}$ that violates our Φ-hiding based extractor lemma, thereby leading us to a contradiction. Below we provide more details.

The reduction algorithm $\mathcal{B}$ first receives the parameters $(N, \mathfrak{s}, e, \widetilde{g})$ where e is a λ-bit prime and $\widetilde{g}$ is a random element in $\mathbb{Z}_N^*$. $\mathcal{B}$ then samples parameters a, b and HPRG seed x randomly as in Hybrid 2. It also samples primes $e_{i,b}$ for all $(i, b) \neq (i^*, 1)$ as in Hybrid 2. It sets $e_{i^*,1}$ as $e_{i^*,1} = e$ and samples $g \leftarrow \mathbb{Z}_N^*$. Now it sets the public parameters pp as $\mathsf{pp} = \left(N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, a, b\right)$. Next, $\mathcal{B}$ sets exponent F as $F = \prod_{k=1}^{j_\epsilon} f_k \prod_{i>i^*, b\in\{0,1\}} e_{i,b}$. If $e \mid F$, then $\mathcal{B}$ aborts and guesses randomly, otherwise it sends F to the challenger and continues. The challenger responds with a challenge bit string y, and then $\mathcal{B}$ computes the HPRG challenge $(y_0, \{y_{i,b}\}_{i,b})$ as:

$$
\begin{aligned}
& z = \widetilde{g}^{\prod_{i>i^*, b\in\{0,1\}} e_{i,b}}, \\
& y_0 = \mathsf{Ext}(z^{\prod_{k=1}^{j_\epsilon} f_k}, \mathfrak{s}), \\
\forall\ (i,b) \preceq (i^*, 0), \quad & y_{i,b} \leftarrow \{0,1\}^\ell, \\
& y_{i^*,1} = y, \\
\forall\ (i,b) \succ (i^*, 1), \quad & y_{i,b} = \mathsf{Ext}(\widetilde{g}^{\prod_{k=1}^{j_\epsilon} f_k \prod_{(j,c)\neq(i,b)\wedge j>i^*, c\in\{0,1\}} e_{j,c}}, \mathfrak{s}).
\end{aligned}
$$

Finally, $\mathcal{B}$ sends pp, n and $(y_0, \{y_{i,b}\}_{i,b})$ to $\mathcal{A}$, and $\mathcal{A}$ outputs its guess β'. If $\beta' = 1$, then $\mathcal{B}$ outputs 0 (i.e., y is computed honestly) as its guess, otherwise it outputs 1 (i.e., y is random bit string) as its guess.

Let us now analyze the advantage of the reduction algorithm $\mathcal{B}$. First, note that $\mathcal{B}$ samples $e_{i,b}$ for $(i,b) \neq (i^*, 1)$ randomly, thus we have that the event $e \mid F$ occurs with only negligible probability since e is a random λ-bit prime. So, $\mathcal{B}$ aborts with atmost negligible probability. Second, since $e_{i,b}$ are randomly drawn λ-bit primes for $i > i^*$, thus sampling z as $z = \widetilde{g}^{\prod_{i>i^*,b} e_{i,b}}, \widetilde{g} \leftarrow \mathbb{Z}_N^*$ instead of $z \leftarrow \mathbb{Z}_N^*$ is statistically indistinguishable. Therefore, $\mathcal{B}$ simulates one of hybrids $3.i^*.0$ and $3.i^*.1$ (in a statistically indistinguishable way) depending upon whether the challenge y is computed as $y = \mathsf{Ext}(\widetilde{g}^{F/e}, \mathfrak{s})$ or $y \leftarrow \{0,1\}^\ell$. Hence, if $\mathcal{A}$ distinguishes with non-negligible probability γ, then $\mathcal{B}$'s advantage is also negligibly close to γ. Thus, the lemma follows. ∎

Lemma 4.4. *Assuming the Φ-hiding assumption holds, then for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $i^* \in [n]$, $p^{\mathcal{A}}_{3.(i^*-1).1} - p^{\mathcal{A}}_{3.i^*.0} \leq \mathit{negl}(\lambda)$.*

Proof. The proof of this lemma is identical to the proof of Lemma 4.3. ∎

Lemma 4.5. *For every adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $p^{\mathcal{A}}_{3.n.1} - p^{\mathcal{A}}_4 \leq \mathit{negl}(\lambda)$.*

Proof. This follows directly from the strong extractor guarantee of Ext. Let $S \subset \mathbb{Z}_N^*$ denote the set $\left\{z \in \mathbb{Z}_N^* : z = \widetilde{z}^{\prod_{k=1}^{j_\epsilon} f_k} \wedge \widetilde{z} \in \mathbb{Z}_N^*\right\}$. Now since N is an RSA modulus of bit length κ, we have that $\log_2 |S| \geq \lambda$ with all-but-negligible probability. Note that y_0 is sampled as $y_0 = \mathsf{Ext}(z, \mathfrak{s})$ where $z \leftarrow S$. Thus, by extractor security, given $\log_2 |S| \geq \lambda$, the claim follows since ϵ_{ext} is negligible. ∎

Combining Lemmas 4.1 to 4.5, we get that $p^{\mathcal{A}}_1 - p^{\mathcal{A}}_4 \leq \epsilon(\lambda)/2 + \widetilde{\mathrm{negl}}(\lambda)$ for some negligible function $\widetilde{\mathrm{negl}}(\cdot)$ (whenever $\delta(\lambda) \geq \epsilon(\lambda)$). Thus, we can conclude that $p^{\mathcal{A}}_1 - p^{\mathcal{A}}_4 \leq 2\epsilon(\lambda)/3$ infinitely often. This contradicts our assumption that $p^{\mathcal{A}}_1 - p^{\mathcal{A}}_4 \geq \delta(\lambda) \geq \epsilon(\lambda)$. Thus, this completes the proof. ∎

5 Hinting PRG from q-DDHI Assumption

We now construct (n, ℓ)-hinting PRG from any $2n$-DDHI hard group generator GGen. Suppose $\mathsf{GGen}(1^\lambda)$ generates a group of order at most 2^κ. The below construction requires $n \geq k + 2\lambda$ and $k = \omega(\log \lambda)$. For the sake of simplicity, we construct a hinting PRG which outputs elements in a group. The construction can be extended to output ℓ-length bit strings for any polynomial ℓ by using standard PRGs and randomness extractors.

$\mathsf{Setup}(1^\lambda)$: Sample a group $\mathcal{G} = (\mathbb{G}, p) \leftarrow \mathsf{GGen}(1^\lambda)$. Sample a generator $g \leftarrow \mathbb{G}$ and random exponents $\alpha \leftarrow \mathbb{Z}_p^*$ and $d_0, d_1 \leftarrow \mathbb{Z}_p$. Output the public parameters $(\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$.

$\mathsf{Eval}(\mathsf{pp}, x, i)$: Parse public parameters pp as $\mathsf{pp} = (\mathcal{G}, g, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$. Set $f = g^{(d_0 x + d_1)\cdot \prod_{j\in[1,n]\setminus\{i\}}(\alpha+2j+x_j)}$. Expand the polynomial f as $\prod_{j=0}^{n} c_j \alpha^j$. Compute and output $\prod_{j=0}^{n}\left(g^{\alpha^j}\right)^{c_j}$.

5.1 Security

We now prove that the above construction is a secure Hinting PRG. Formally, we prove

Theorem 5.1. *If the $2n$-DDHI assumption (Assumption 2) holds on group generator* GGen*, then the hinting PRG construction described above is secure as per Definition 2.4.*

Proof. We now prove that the above construction is a secure hinting PRG via a sequence of hybrids. In the proof, we use F_x as a shorthand for $\prod_{j=1}^{n}(\alpha+2j+x_j)$. In the first hybrid, y_{i,x_i} are structured ($y_{i,x_i} = g^{(d_0x+d_1)\cdot(\alpha+2i+x_i)^{-1}\prod_{j=1}^{n}(\alpha+2j+x_j)}$) and $y_{i,1-x_i}$ are sampled randomly in the HPRG challenge. In the next hybrid, we switch $y_{i,1-x_i}$ to structured values, i.e., $y_{i,b} = g^{(d_0x+d_1)\cdot(\alpha+2i+b)^{-1}\prod_{j=1}^{n}(\alpha+2j+x_j)}$ using DDHI assumption. In the next hybrid, we erase the information about x in the challenge. Concretely, we show that $(d_0x+d_1)\cdot\prod_{j=1}^{n}(\alpha+2j+x_j) \bmod p$ is statistically indstinguishable from random value in $\mathbb{Z}_p$ and consequently switch $y_{i,b}$ to $g^{r\cdot(\alpha+2i+b)^{-1}}$ for a randomly sampled r. We then use DDHI assumption to switch the challenge to random group elements.

Hybrid H_0: This is same as the original hinting PRG game when the challenger always chooses $\beta = 0$.

1. The challenger first samples a group $\mathcal{G} = (\mathbb{G}, p) \leftarrow \mathsf{GGen}(1^\lambda)$ and generator $g \leftarrow \mathbb{G}$. It then samples random values $\alpha, d_0, d_1 \leftarrow \mathbb{Z}_p$. It then computes $\mathsf{pp} = (\mathcal{G}, g, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$.
2. It then samples a bit string $x \leftarrow \{0,1\}^n$, random values $r_i \leftarrow \mathbb{Z}_p$ for $i \in [n]$. It then computes the challenge $y_0 = g^{F_x}, y_{i,x_i} = g^{F_x/(\alpha+2i+x_i)}, y_{i,1-x_i} = g^{r_i}$ for $i \in [n]$.
3. The challenger sends public parameters pp and the challenge $\{y_0, \{y_{i,b}\}_{i,b}\}$ to the adversary.
4. The adversary outputs a bit β'.

Hybrid H_1: This is same as previous game, except that the challenger aborts if there exists an $i \in [2n+1]$ s.t. $\alpha + i = 0 \bmod p$.

1. The challenger first samples a group $\mathcal{G} = (\mathbb{G}, p) \leftarrow \mathsf{GGen}(1^\lambda)$ and generator $g \leftarrow \mathbb{G}$. It then samples random values $\alpha, d_0, d_1 \leftarrow \mathbb{Z}_p$. It aborts if $\alpha + i = 0 \bmod p$ for any $i \in [2n+1]$. It then computes $\mathsf{pp} = (p, g, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$.

We now define a sequence of $n+1$ hybrids. For the sake of simplicity, let Hybrid $H_{2.0}$ be same as Hybrid H_1.

Hybrid $H_{2.j}$ ($j \in [n]$): This is same as previous game, except that the challenger computes $y_{i,1-x_i}$ differently.

2. It then samples a bit string $x \leftarrow \{0,1\}^n$, random values $r_i \leftarrow \mathbb{Z}_p$ for $i \in [n]$. It then computes the challenge $y_0 = g^{F_x}$, $y_{i,b} = g^{F_x/(\alpha+2i+b)}$ for all (i,b) s.t. $i \le j$ or $b = x_i$, $y_{i,1-x_i} = g^{r_i}$ for $i > j$.

Hybrid H_3: This is same as Hybrid $H_{2.n}$ except that the challenger uses a random value r instead of F_x.

2. The challenger samples a random value $r \leftarrow \mathbb{Z}_p$. It then computes the challenge $y_0 = g^r$, $y_{i,b} = g^{r/(\alpha+2i+b)}$ for all $(i,b) \in [n]\times\{0,1\}$.

We next define a sequence of $2n+1$ hybrids. Let us define Hybrid $H_{4.0.1}$ be same as Hybrid H_3.

Hybrid $H_{4.j.b'}$ ($j \in [n], b' \in \{0,1\}$): This is same as Hybrid H_3 except that for $(i,b) \preceq (j,b')$, the challenger samples $y_{i,b}$ uniformly at random.

2. The challenger samples a random value $r \leftarrow \mathbb{Z}_p$ and $r_{i,b} \leftarrow \mathbb{Z}_p$ for $(i,b) \preceq (j,b')$. It then computes the challenge $y_0 = g^r$, $y_{i,b} = g^{r_{i,b}}$ for $(i,b) \preceq (j,b')$, and $y_{i,b} = g^{r/(\alpha+2i+b)}$ for $(i,b) \succ (j,b')$.

Hybrid H_5. This is same as Hybrid $H_{4.n.1}$ except that the challenger does not abort if there exists $i \in [2n+1]$ such that $\alpha + i = 0 \bmod p$.

1. The challenger first samples a group $\mathcal{G} = (\mathbb{G}, p) \leftarrow \mathsf{GGen}(1^\lambda)$ and generator $g \leftarrow \mathbb{G}$. It then samples random values $\alpha, d_0, d_1 \leftarrow \mathbb{Z}_p$. It then computes $\mathsf{pp} = (\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$.

Note that hybrid H_0 is the original hinting PRG game when challenger always chooses $\beta = 0$ and hybrid H_5 is the original hinting PRG game when challenger always chooses $\beta = 1$. We prove that these 2 hybrids are computationally indistinguishable using the following lemmas. For any PPT adversary $\mathcal{A}$, let $p_s^{\mathcal{A}}$ be the probability that $\mathcal{A}$ outputs 1 in Hybrid H_s.

Lemma 5.1. *There exists a negligible function negl$(\cdot)$ such that for every adversary $\mathcal{A}$ and every $\lambda \in \mathbb{N}$, we have $|p_0^{\mathcal{A}} - p_1^{\mathcal{A}}| \leq$ negl(λ).*

Proof. The distribution of challenger's output is same in Hybrids H_0 and H_1, except when $\alpha \in [p-1, p-2n-1]$. This event happens with probability $(2n+1)/p$. Assuming p is super-polynomial in λ, the event $\alpha \in [p-1, p-2n-1]$ happens with negligible probability. ∎

Lemma 5.2. *Assuming $2n$-DDHI assumption holds on group generator* GGen*, for every PPT adversary $\mathcal{A}$ and every index $j \in [n]$, there exists a negligible function negl$(\cdot)$ such that for every $\lambda \in \mathbb{N}$, we have $|p_{2.j}^{\mathcal{A}} - p_{2.j-1}^{\mathcal{A}}| \leq$ negl(λ).*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ and an index j s.t. $|p_{2.j}^{\mathcal{A}} - p_{2.j-1}^{\mathcal{A}}|$ is non-negligible. We construct a reduction algorithm $\mathcal{B}$ that breaks $2n$-DDHI assumption on group generator GGen.

The $2n$-DDHI game challenger $\mathcal{C}$ first sends challenge $(\mathcal{G}, h, h^\gamma, h^{\gamma^2}, \cdots, h^{\gamma^{2n}}, T)$ to the reduction algorithm $\mathcal{B}$. The reduction algorithm samples $x \leftarrow \{0,1\}^n$, implicitly sets $\alpha = \gamma - 2j + x_j - 1$ and aborts if there exists $i \in [2n+1]$ such that $h^{\alpha+i} = \mathbf{1}_{\mathbb{G}}$ ($\mathbf{1}_{\mathbb{G}}$ is identity element of the group $\mathbb{G}$). $\mathcal{B}$ then samples random elements $d_0, d_1 \leftarrow \mathbb{Z}_p$, computes a generator $g = h^{\prod_{k=1}^{j-1}(\alpha+2k+1-x_k)}$, and the public parameters $\mathsf{pp} = (\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots g^{\alpha^n}, d_0, d_1)$. $\mathcal{B}$ then computes Hinting PRG challenge as follows. It first computes $y_0 = g^{(d_0 x + d_1)\cdot \prod_{k=1}^{n}(\alpha+2k+x_k)}$, $y_{i,x_i} = g^{(d_0 x+d_1)\cdot\prod_{k\in[1,n]\setminus\{i\}}(\alpha+2k+x_k)}$ for $i \in [n]$. It then computes $y_{i,1-x_i} = h^{(d_0x+d_1)\cdot\prod_{k\in[j-1]\setminus\{i\}}(\alpha+2k+1-x_k)\cdot\prod_{k=1}^{n}(\alpha+2k+x_k)}$ for $i < j$. It then samples $r_i \leftarrow \mathbb{Z}_p$ and sets $y_{i,1-x_i} = g^{r_i}$ for $i > j$. In order to compute $y_{j,\overline{x_j}}$, $\mathcal{B}$ expands the polynomial

$$f(\alpha) = \frac{(d_0 x + d_1)\cdot \prod_{k=1}^{j-1}(\alpha+2k+1-x_k)\cdot\prod_{k=1}^{n}(\alpha+2k+x_k)}{(\alpha+2j+1-x_j)} = \frac{c}{\gamma} + \sum_{k=0}^{n+j-2} c_k \gamma^k,$$

where $\gamma = (\alpha+2j+1-x_j)$ and $c, \{c_k\}$ are some functions of x, d_0, d_1. $\mathcal{B}$ sets $y_{j,\overline{x_j}} = T^c \cdot \prod_{k=0}^{n+j-2} h^{c_k \gamma^k}$. Note that pp and $(y_0, \{y_{i,b}\}_{i,b})$ can be computed given the challenge sent by $\mathcal{C}$. Finally, $\mathcal{B}$ sends public parameters pp and challenge $(y_0, \{y_{i,b}\}_{i,b})$ to the adversary $\mathcal{A}$. The adversary outputs a bit β', which $\mathcal{B}$ outputs as its guess in the $2n$-DDHI game.

Note that $\alpha+2k+1-x_k \neq 0 \bmod p$ for all $k < j$. Therefore g is a generator of the group $\mathbb{G}$. Moreover, α is uniformly distributed over $\mathbb{Z}_p$ since γ is uniformly sampled from $\mathbb{Z}_p$. Therefore, the distribution of pp sent by $\mathcal{B}$ matches the distribution of pp sent by Hybrid $H_{2.j}$ and $H_{2.j-1}$ challengers. Moreover if T is a random group element, then $\mathcal{B}$ emulates Hybrid $H_{2.j-1}$ challenger to $\mathcal{A}$. If $T = h^{1/\gamma}$, then $\mathcal{B}$ emulates Hybrid $H_{2.j}$ challenger to $\mathcal{A}$. By our assumption, $|p_{2.j}^{\mathcal{A}} - p_{2.j-1}^{\mathcal{A}}| = |\Pr[\beta'=1|\delta=0] - \Pr[\beta'=1|\delta=1]|$ is non-negligible. Therefore, $\mathcal{B}$ breaks $2n$-DDHI assumption of GGen. ∎

Lemma 5.3. *For every adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every $\lambda \in \mathbb{N}$, we have $|p_{2.n}^{\mathcal{A}} - p_3^{\mathcal{A}}| \leq$ negl(λ).*

Proof. By applying Lemma C.1 with a uniform source S on $\{0,1\}^n$, the statistical difference between Hybrids $H_{2.n}$ and H_3 is negligible in λ. ∎

Lemma 5.4. *Assuming $2n$-DDHI assumption holds on group generator* GGen*, for every PPT adversary $\mathcal{A}$, every index $j \in [n]$ and bit b', there exists a negligible function negl$(\cdot)$ such that for every $\lambda \in \mathbb{N}$, we have $|p^{\mathcal{A}}_{4.j.b'} - p^{\mathcal{A}}_{4.j-1+b'.1-b'}| \leq negl(\lambda)$.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$, an index j and bit b' s.t. $|p^{\mathcal{A}}_{4.j.b'} - p^{\mathcal{A}}_{4.j-1+b'.1-b'}|$ is non-negligible. We construct a reduction algorithm $\mathcal{B}$ that breaks $2n$-DDHI assumption on group generator GGen.

The $2n$-DDHI game challenger $\mathcal{C}$ first sends the challenge $(\mathcal{G}, h, h^{\gamma}, h^{\gamma^2}, \cdots, h^{\gamma^{2n}}, T)$ to the reduction algorithm $\mathcal{B}$. The reduction algorithm samples $x \leftarrow \{0,1\}^n$, implicitly sets $\alpha = \gamma - 2j - b'$ and aborts if there exists $i \in [2n+1]$ such that $h^{\alpha+i} = \mathbf{1}_{\mathbb{G}}$ ($\mathbf{1}_{\mathbb{G}}$ is identity element of the group $\mathbb{G}$). $\mathcal{B}$ then samples random elements $d_0, d_1 \leftarrow \mathbb{Z}_p$, sets the public parameters $\mathsf{pp} = (\mathcal{G}, h, h^{\alpha}, h^{\alpha^2}, \cdots h^{\alpha^n}, d_0, d_1)$. $\mathcal{B}$ then samples $r' \leftarrow \mathbb{Z}_p$, $r_{i,b} \leftarrow \mathbb{Z}_p$ for $(i,b) \prec (j,b')$, and implicitly sets $r = r' \cdot \prod_{\{(k,\beta)\succ(j,b')\}}(\alpha + 2k + \beta)$. $\mathcal{B}$ then computes Hinting PRG challenge as follows. It first computes $y_0 = h^r = h^{r' \cdot \prod_{\{(k,\beta)\succ(j,b')\}}(\alpha+2k+\beta)}$, $y_{i,b} = g^{r_{i,b}}$ for $(i,b) \prec (j,b')$. It then computes $y_{i,b} = h^{r \cdot e^{-1}_{i,b}} = h^{r' \cdot \prod_{(k,\beta)\succ(j,b')\wedge(k,\beta)\neq(i,b)}(\alpha+2k+\beta)}$ for $(i,b) \succ (j,b')$. In order to compute $y_{j,b'}$, $\mathcal{B}$ expands the polynomial

$$f(\alpha) = \frac{r}{(\alpha + 2j + b')} = \frac{r' \cdot \prod_{(k,b)\succ(j,b')}(\alpha + 2k + b)}{(\alpha + 2j + b')} = \frac{c}{\gamma} + \sum_{k=0}^{2n} c_k \gamma^k,$$

where $\gamma = (\alpha + 2j + b')$ and $c, \{c_k\}$ are some functions of r'. $\mathcal{B}$ sets $y_{j,b'} = T^c \cdot \prod_{k=0}^{2n} h^{c_k \gamma^k}$. Note that pp and $(y_0, \{y_{i,b}\}_{i,b})$ can be computed given the challenge sent by $\mathcal{C}$. Finally, $\mathcal{B}$ sends public parameters pp and challenge $(y_0, \{y_{i,b}\}_{i,b})$ to the adversary $\mathcal{A}$. The adversary outputs a bit β', which $\mathcal{B}$ outputs as its guess in $2n$-DDHI game.

Note that the distribution of pp sent by $\mathcal{B}$ matches the distribution of pp sent by Hybrid $H_{4.j.b'}$ and $H_{4.j-1+b'.1-b'}$ challengers. As ρ' is uniformly sampled in $\mathbb{Z}_p$ and $(\alpha + 2j + b') \neq 0 \bmod p$, ρ is uniformly distributed in $\mathbb{Z}_p$. Moreover, if $T = h^{1/\gamma}$ then $\mathcal{B}$ emulates Hybrid $H_{4.j-1+b'.1.b'}$ challenger to $\mathcal{A}$. If T is a random group element, then $\mathcal{B}$ emulates Hybrid $H_{4.j.b'}$ challenger to $\mathcal{A}$. By our assumption, $|p^{\mathcal{A}}_{4.j.b'} - p^{\mathcal{A}}_{4.j-1+b'.1-b'}| = |\Pr[\beta' = 1|\delta = 0] - \Pr[\beta' = 1|\delta = 1]|$ is non-negligible. Therefore, $\mathcal{B}$ breaks $2n$-DDHI assumption of $\mathbb{G}$. ■

Lemma 5.5. *There exists a negligible function negl$(\cdot)$ such that for every adversary $\mathcal{A}$ and every $\lambda \in \mathbb{N}$, we have $|p^{\mathcal{A}}_{4.n.1} - p^{\mathcal{A}}_{5}| \leq negl(\lambda)$.*

Proof. This proof is same as proof of Lemma 5.1. ■

By the above sequence of lemmas and triangle inequality, Hybrids H_0 and H_5 are computationally indistinguishable. Therefore, the above construction is a secure Hinting PRG. ■

6 One-Way Function with Encryption from Φ-Hiding Assumption

In this section, we construct (k, n, ℓ)-recyclable One-Way Function with Encryption (OWFE) from Phi-Hiding assumption. The construction assumes $k \geq 7\lambda$ and $n - k \leq \alpha \log n$ for any fixed constant α. For any parameters λ, ℓ, let $\mathsf{Ext}_{\lambda,\ell} : \{0,1\}^{\lambda} \times \mathcal{S} \to \{0,1\}^{\ell}$ be a $(\lambda - 1, \epsilon_{\mathsf{Ext}})$ strong seeded extractor, where ϵ_{Ext} is negligible in λ. [9] Let p_i denote the i^{th} (smallest) prime, i.e. $p_1 = 2, p_2 = 3, \ldots,$ and $\widetilde{e_i} = \lceil \log_{p_i} N \rceil$ for all i. And, let f_i denote $p_i^{\widetilde{e_i}}$ for all i. The construction proceeds as follows.

[9] Note that such an extractor exists for $\ell = c \cdot \lambda$ for some constant $c < 1$. The construction can be extended for any $\ell \geq \lambda$ with the help of PRGs.

$K(1^\lambda)$: On input security parameter λ and length ℓ, set RSA modulus length $\kappa = 5\lambda$, and sample RSA modulus $N \leftarrow \mathsf{RSA}(\kappa)$. Next, sample a generator $g \leftarrow \mathbb{Z}_N^*$, $2n$ (λ-bit) primes $e_{i,b} \leftarrow \text{PRIMES}(\lambda)$ for $(i,b) \in [n] \times \{0,1\}$ and elements $d_0, d_1 \leftarrow \mathbb{Z}_N$. Then, sample a seed $\mathfrak{s} \leftarrow \mathcal{S}$ of extractor $\mathsf{Ext}_{\lambda,\ell}$ and output public parameters $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$.

$f(\mathsf{pp}, x)$: Let $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$. Output $y = g^{f_1 \cdot f_2 \cdot (d_0 x + d_1) \prod_i e_{i,x_i}} \bmod N$.

$E_1(\mathsf{pp}, (i,b); \rho)$: Parse pp as $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$. Output ciphertext $\mathsf{ct} = (g^{\rho \cdot e_{i,b}} \bmod N, i, b)$.

$E_2(\mathsf{pp}, (y, i, b); \rho)$: Let pp be $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$. Compute $h = y^\rho \bmod N$ and output $z = \mathsf{Ext}(h, \mathfrak{s})$.

$D(\mathsf{pp}, \mathsf{ct}, x)$: Let $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$. Parse ct as (t, i, b). If $b = x_i$, compute $h = t^{f_1 \cdot f_2 \cdot (d_0 x + d_1) \prod_{j \neq i} e_{j,x_j}} \bmod N$ and output $\mathsf{Ext}(h, \mathfrak{s})$. Otherwise, output $\perp$.

Correctness. For any public parameters $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$, any string $x \in \{0,1\}^n$, any index $i \in [n]$, any randomness ρ, we have $D(\mathsf{pp}, E_1(\mathsf{pp}, (i, x_i); \rho), x) = g^{\rho f_1 \cdot f_2 \cdot (d_0 x + d_1) \prod_j e_{j,x_j}} = f(\mathsf{pp}, x)^\rho = E_2(\mathsf{pp}, (f(\mathsf{pp}, x), i, x_i); \rho)$.

6.1 Security

We now prove the one-wayness, encryption security and smoothness properties of the above scheme.

One-Wayness. We now prove that the above construction satisifes (k, n)-one-wayness property when $k \geq 7\lambda$ and $n - k \leq \alpha \log n$ for any fixed constant α.

Theorem 6.1. *Assuming Φ-hiding assumption holds, the above construction satisfies (k, n, ℓ)-one-wayness property as per Definition 2.1.*

Proof. We first prove that no PPT adversary can win the following game with non-negligible advantage assuming Φ-hiding assumption. We then prove how a PPT adversary breaking one-wayness property of the above scheme can be used to break the following game.

Game G: The challenger chooses RSA modulus $\kappa = 5\lambda$, samples $N \leftarrow \mathsf{RSA}(\kappa)$, prime $e \leftarrow \text{PRIMES}(\lambda)$ and a value $z \leftarrow \mathbb{Z}_N^*$. The challenger sends (N, e, z) to the adversary, which then outputs w. The adversary wins if $w^e = z \bmod N$.

We now argue that no PPT adversary can win the above game with non-negligible probability.

Lemma 6.1. *Assuming Φ-hiding assumption holds, for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the probability that $\mathcal{A}$ wins in Game G is at most negl(λ).*

Proof. We prove the lemma using the following intermediate Game H.

Game H: The challenger chooses RSA modulus $\kappa = 5\lambda$, samples prime $e \leftarrow \text{PRIMES}(\lambda)$ and $N \leftarrow \mathsf{RSA}(\kappa)$ s.t. $e | \phi(N)$. It then samples an element $z \leftarrow \mathbb{Z}_N^*$. The challenger sends (N, e, z) to the adversary, which then outputs w. The adversary wins if $w^e = z \bmod N$.

Let the advantage of any adversary $\mathcal{A}$ in Game G be $\mathsf{Adv}_G^{\mathcal{A}}$ and in Game H be $\mathsf{Adv}_H^{\mathcal{A}}$.

Claim 6.1. *For every adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every $\lambda \in \mathbb{N}$, $\mathsf{Adv}_H^{\mathcal{A}} \leq$ negl(λ).*

Proof. As $e | \phi(N)$, only a negligible fraction of $z \in \mathbb{Z}_N^*$ have a w s.t. $w^e = z \bmod N$. Therefore, no PPT adversary can find a w s.t. $w^e = z \bmod N$ with non-negligible probability. ■

Claim 6.2. *Assuming Φ-hiding assumption holds, for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every $\lambda \in \mathbb{N}$, $|\mathsf{Adv}_G^{\mathcal{A}} - \mathsf{Adv}_H^{\mathcal{A}}| \leq negl(\lambda)$.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ such that $|\mathsf{Adv}_G^{\mathcal{A}} - \mathsf{Adv}_H^{\mathcal{A}}|$ is non-negligible. We construct a reduction algorithm that breaks Φ-hiding assumption. $\mathcal{B}$ samples $e \leftarrow \text{PRIMES}(\lambda)$ and plays Φ-hiding game for e. The challenger sends RSA modulus N to $\mathcal{B}$, which samples $z \leftarrow \mathbb{Z}_N^*$ and sends (N, e, z) to $\mathcal{A}$. If $\mathcal{A}$ outputs w s.t. $w^e = z \bmod N$, then $\mathcal{B}$ guesses that $\phi(N)$ is uniformly sampled from $\mathsf{RSA}(\kappa)$. Otherwise, it guesses that $e|\phi(N)$. ∎

By the above 2 claims and triangle inequality, no PPT adversary can win Game G with non-negligible advantage. ∎

Lemma 6.2. *Assuming Φ-hiding assumption holds, for every PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the advantage of $\mathcal{A}$ in (k, n)-one-wayness game is at most negl(λ).*

Proof. Suppose there exist a PPT adversary $\mathcal{A}$ that breaks (k, n)-one-wayness property of the encryption scheme with non-negligible probability ϵ. We construct a reduction algorithm $\mathcal{B}$ that wins against Game G challenger $\mathcal{C}$.

The adversary first sends a (k, n) source S to $\mathcal{B}$. The challenger $\mathcal{C}$ then sends (N, e, z) to $\mathcal{B}$. The reduction algorithm samples a bit string $x \leftarrow S$, an index $j \leftarrow [n]$, extractor seed $\mathfrak{s} \leftarrow \mathcal{S}$, exponents $d_0, d_1 \leftarrow \mathbb{Z}_p$ primes $e_{i,b'} \leftarrow \text{PRIMES}(\lambda)$ for $(i, b') \neq (j, 1-x_j)$. It then sets generator $g = z$ and prime $e_{j,1-x_j} = e$. $\mathcal{B}$ then sends public parameters $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b'}\}_{i,b'}, d_0, d_1)$ and challenge $y = z^{\prod_i e_{i,x_i}} \bmod N$ to the adversary. The adversary outputs x'. If $f(\mathsf{pp}, x') \neq f(\mathsf{pp}, x)$ or $x_j = x'_j$, then $\mathcal{B}$ aborts. Otherwise, we have $h^e = z^F \bmod N$, where $F = f_1 \cdot f_2 \cdot (d_0 x + d_1) \prod_i e_{i,x_i}$ and $h = z^{f_1 \cdot f_2 \cdot (d_0 x' + d_1) \prod_{i \neq j} e_{i,x'_i}} \bmod N$. As e is a randomly sampled λ-bit prime, $e \nmid F$ with overwhelming probability. $\mathcal{B}$ computes $z^{1/e} \bmod N$ using Shamir's trick [Sha83]. Concretely, $\mathcal{B}$ first computes integers a, b s.t. $a \cdot e + b \cdot F = 1$ and outputs $w = h^b \cdot z^a \bmod N$.

We now analyze the advantage of $\mathcal{B}$ in Game G. By our assumption, $f(\mathsf{pp}, x') = f(\mathsf{pp}, x)$ with non-negligible probability ϵ. We prove that $x' \neq x$ with non-negligible probability. As $k \geq \kappa + 2\lambda$, we know that for any pp, $\Pr_{x \leftarrow S}[\exists t \in \{0,1\}^n \text{ s.t. } x \neq t \wedge f(\mathsf{pp}, x) = f(\mathsf{pp}, t)] \geq 1 - \text{negl}(\lambda)$. Therefore, $\Pr[x' \neq x \wedge f(\mathsf{pp}, x) = f(\mathsf{pp}, x')] \geq \epsilon/2 - \text{negl}(\lambda)$ and $\Pr[x'_j \neq x_j \wedge f(\mathsf{pp}, x) = f(\mathsf{pp}, x')] \geq \epsilon/2n - \text{negl}(\lambda)$ as j is sampled uniformly from $[n]$. Note that if $\mathcal{B}$ does not abort, it outputs w s.t. $w^e = z \bmod N$ with overwhelming probability. Therefore, $\mathcal{B}$ breaks Game G security with non-negligible probability $\epsilon/2n - \text{negl}(\lambda)$. ∎

∎

Security of Encryption. We now prove that the above construction satisifes encryption security property.

Theorem 6.2. *Assuming Φ-hiding assumption holds, the above construction satisfies encryption security property as per Definition 2.2.*

Proof. We prove the above theorem via a sequence of following hybrids.

Hybrid H_0: This is same as original OWFE security of encryption game when the challenger chooses $\beta = 0$.

1. The adversary sends bit string $x \leftarrow \{0,1\}^n$ and index $j \in [n]$ to the challenger.
2. The challenger sets modulus length $\kappa = 5\lambda$ and samples $N \leftarrow \mathsf{RSA}(\kappa)$, generator $g \leftarrow \mathbb{Z}_N^*$, extractor seed $\mathfrak{s} \leftarrow \mathcal{S}$ and primes $e_{i,b} \leftarrow \text{PRIMES}(\lambda)$ for $i \in [n] \times \{0,1\}$.
3. The challenger samples $\rho \leftarrow \mathbb{Z}_N$, computes $\mathsf{ct} = g^{\rho \cdot e_{j,1-x_j}}$, $z = \mathsf{Ext}(g^{\rho f_1 \cdot f_2 \cdot (d_0 x + d_1) \cdot \prod_i e_{i,x_i}} \bmod N, \mathfrak{s})$.
4. The challenger sends $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}), \mathsf{ct}, z$ to the adversary $\mathcal{A}$, which outputs a bit α.

Hybrid H_1: This hybrid is similar to previous hybrid except for the following changes.

3. The challenger samples $\tilde{g} \leftarrow \mathbb{Z}_N^*$, computes $\mathsf{ct} = \tilde{g}$, $z = \mathsf{Ext}(\tilde{g}^{f_1 \cdot f_2 \cdot (d_0 x + d_1) \prod_i e_{i,x_i} \cdot e_{j,1-x_j}^{-1}} \bmod N, \mathfrak{s})$.

Hybrid H_2: This hybrid is same as previous game except that the challenger samples z uniformly at random.

3. The challenger samples $\tilde{g} \leftarrow \mathbb{Z}_N^*$, computes $\mathsf{ct} = \tilde{g}, z \leftarrow \{0,1\}^\ell$.

Hybrid H_3: This is same as original OWFE security of encryption game when the challenger chooses $\beta = 1$.

3. The challenger samples $\rho \leftarrow \mathbb{Z}_N$, computes $\mathsf{ct} = g^{\rho \cdot e_{j,1-x_j}}, z \leftarrow \{0,1\}^\ell$.

For any PPT adversary $\mathcal{A}$, let the probability that $\mathcal{A}$ outputs 1 in Hybrid H_s be $p_s^{\mathcal{A}}$. We prove that Hybrids H_0 and H_3 are computationally indstinguishable via the sequence of following lemmas.

Lemma 6.3. *For any adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every security parameter $\lambda \in \mathbb{N}$, we have $|p_0^{\mathcal{A}} - p_1^{\mathcal{A}}| \le$ negl(λ).*

Proof. We first observe that for any N, prime $e \nmid \phi(N)$ and generator $g \in \mathbb{Z}_N^*$, the distribution of $g^{\rho \cdot e} \bmod N$ for a randomly sampled $\rho \leftarrow \mathbb{Z}_{\phi(N)}$ is identical to the distribution $\tilde{g} \leftarrow \mathbb{Z}_N^*$. This follows from the fact that g and g^e are generators of $\mathbb{Z}_N^*$. For a randomly sampled λ bit prime e, we know that $e \nmid \phi(N)$ with overwhelming probability. Similarly, for a randomly sampled $\rho \leftarrow \mathbb{Z}_N$, we know that $\rho \in \mathbb{Z}_{\phi(N)}$ with overwhelming probability. As a result, $\{\tilde{g} : \tilde{g} \leftarrow \mathbb{Z}_N^*\}$ is statistically indistinguishable from $\{g^{\rho \cdot e} : g \leftarrow \mathbb{Z}_N^*, \rho \leftarrow \mathbb{Z}_N, e \leftarrow \text{PRIMES}(\lambda)\}$. By a similar argument, for any F, the distribution $\{(g^{\rho \cdot e} \bmod N, g^{\rho \cdot F} \bmod N) : g \leftarrow \mathbb{Z}_N^*, \rho \leftarrow \mathbb{Z}_N, e \leftarrow \text{PRIMES}(\lambda)\}$ is statistically indistinguishable from the distribution $\{(\tilde{g}, \tilde{g}^{F \cdot e^{-1}} \bmod N) : \tilde{g} \leftarrow \mathbb{Z}_N^*, e \leftarrow \text{PRIMES}(\lambda)\}$. Therefore, for every adversary $\mathcal{A}$, $|p_0^{\mathcal{A}} - p_1^{\mathcal{A}}| \le \text{negl}(\lambda)$. ■

Lemma 6.4. *Assuming Φ-hiding assumption holds, for any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every security parameter $\lambda \in \mathbb{N}$, we have $|p_1^{\mathcal{A}} - p_2^{\mathcal{A}}| \le$ negl(λ).*

Proof. The above lemma follows from ϕ-based Extractor lemma (Lemma 3.6). Suppose there exists a PPT adversary $\mathcal{A}$ such that $|p_1^{\mathcal{A}} - p_2^{\mathcal{A}}|$ is non-negligible. We construct a reduction algorithm $\mathcal{B}$ that violates ϕ-based extractor lemma.

The extractor lemma challenger first samples $N \leftarrow \mathsf{RSA}(\kappa)$, $\mathfrak{s} \leftarrow \mathcal{S}$, $e \leftarrow \text{PRIMES}(\lambda)$, $\tilde{g} \leftarrow \mathbb{Z}_N^*$ and sends $(N, \mathfrak{s}, e, \tilde{g})$ to reduction algorithm $\mathcal{B}$. The adversary $\mathcal{A}$ then sends a string $x \in \{0,1\}^n$ and index $j \in [n]$ to $\mathcal{B}$. $\mathcal{B}$ samples generator g, values $d_0, d_1 \leftarrow \mathbb{Z}_N$, and primes $e_{i,b} \leftarrow \text{PRIMES}(\lambda)$ for $(i,b) \neq (j, 1-x_j)$. $\mathcal{B}$ then sets $e_{j,1-x_j} = e$ and computes $F = f_1 \cdot f_2 \cdot (d_0 x + d_1) \prod_i e_{i,x_i}$. If $e | F$, the reduction algorithm aborts and guesses randomly. As e is a λ-bit prime, this happens with negligible probability. If $e \nmid F$, then $\mathcal{B}$ sends F to the challenger, which samples a bit $\gamma \leftarrow \{0,1\}$. If $\gamma = 0$, $\mathcal{C}$ computes $\tilde{z} \leftarrow \mathsf{Ext}(\tilde{g}^{F/e}, \mathfrak{s})$. If $\gamma = 1$, $\mathcal{C}$ samples $\tilde{z} \leftarrow \{0,1\}^\ell$. The challenger sends $\tilde{z}$ to $\mathcal{B}$. The reduction algorithm sets $\mathsf{ct} = \tilde{g}, z = \tilde{z}$ and sends $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1), \mathsf{ct}, z$ to $\mathcal{A}$. The adversary outputs a bit α. $\mathcal{B}$ outputs α as its guess in extractor lemma game.

Note that if $\gamma = 0$, then the distribution of $\mathsf{pp}, \mathsf{ct}, z$ sent by $\mathcal{B}$ is statistically indistinguishable from that of Hybrid H_1 challenger. If $\gamma = 1$, then the distribution of $\mathsf{pp}, \mathsf{ct}, z$ sent by $\mathcal{B}$ is statistically indistinguishable from that of H_2 challenger. Consequently if $\mathcal{B}$ does not abort, the advantage $|\Pr[\alpha = 1 | \gamma - 0] - \Pr[\alpha = 1 | \gamma = 1]| \ge |p_1^{\mathcal{A}} - p_2^{\mathcal{A}}| - \text{negl}(\lambda)$ is non-negligible. As $\mathcal{B}$ aborts with only negligible probability, it wins the extractor lemma game with non-negligible probability. ■

Lemma 6.5. *For any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for every security parameter $\lambda \in \mathbb{N}$, we have $|p_2^{\mathcal{A}} - p_3^{\mathcal{A}}| \le$ negl(λ).*

Proof. The distribution $\{\tilde{g} : \tilde{g} \leftarrow \mathbb{Z}_N^*\}$ is statistically indistinguishable from $\{g^{\rho \cdot e} : g \leftarrow \mathbb{Z}_N^*, \rho \leftarrow \mathbb{Z}_N, e \leftarrow \text{PRIMES}(\lambda)\}$ as mentioned in proof of Claim 6.3. ■

By the above lemmas and triangle theorem, no PPT adversary can distinguish between Hybrids H_0 and H_3 with non-negligible probability assuming Φ-hiding assumption. ■

Smoothness. We now prove that the above construction satisifes (k,n)-smoothness property when $k \geq 7\lambda$ and $n-k \leq \alpha \log n$ for any fixed constant α.

Theorem 6.3. *Assuming Φ-hiding assumption holds, the above construction satisfies (k,n)-smoothness security property as per Definition 2.3.*

Proof. First, we introduce a useful notation. For any constant $\epsilon > 0$, let j_ϵ be the smallest index such that $p_{j_\epsilon} > (2^{n-k+2}\log N/\epsilon)^4$. Note that $(2^{n-k+2}\log N/\epsilon)^4$ is polynomial in λ for the given setting of parameters. The proof of security follows via a sequence of hybrids. Below we first describe the sequence of hybrids, and later argue indistinguishability to complete the proof. At a very high level, the proof structure is somewhat similar to that used in [Zha16], where for proving security one first assumes (for the sake of contradiction) that the adversary wins with some non-negligible probability δ and then depending upon δ, one could describe a sequence of hybrids such that no PPT adversary can win with probability more than $2\delta/3$. This acts as a contradiction, thereby completing the proof.

For any PPT adversary $\mathcal{A}$, let $p_s^{\mathcal{A}}$ be the probability that $\mathcal{A}$ outputs 1 in Hybrid H_s. For the sake of contradiction, we assume that $\mathcal{A}$ breaks (k,n)-smoothness property with non-negligible advantage $\delta(\lambda)$ i.e., there exists a polynomial $v(\cdot)$ s.t. $|p_0^{\mathcal{A}} - p_2^{\mathcal{A}}| = \delta(\lambda) > \frac{1}{v(\lambda)}$ for infinitely often $\lambda \in \mathbb{N}$. Let $\epsilon = \frac{1}{2v(\lambda)}$. We provide a non-uniform reduction where the description of hybrids and the reduction algorithm depends on ϵ.

Hybrid H_0: This is same as the original smoothness security game, except that the challenger always chooses source S_0.

1. The adversary first sends two (k,n) sources S_0, S_1 to the challenger. The challenger sets modulus length $\kappa = 5\lambda$ and samples $N \leftarrow \mathsf{RSA}(\kappa)$, extractor seed $\mathfrak{s} \leftarrow \mathcal{S}$, elements $d_0, d_1 \leftarrow \mathbb{Z}_N$ and primes $e_{i,b} \leftarrow \mathrm{PRIMES}(\lambda)$ for $(i,b) \in [n] \times \{0,1\}$.
2. The challenger then samples a generator $g \leftarrow \mathbb{Z}_N^*$ and sets public parameters $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$.
3. The challenger samples $x \leftarrow S_0$ and sends $\mathsf{pp}, y = g^{f_1 \cdot f_2 \cdot (d_0 x + d_1)\prod_{i=1}^n e_{i,x_i}} \bmod N$ to the adversary.
4. The adversary outputs a bit b'.

Hybrid H_1: In this hybrid, the challenger does not sample x and picks the challenge y from a uniform distribution.

2. The challenger then samples a generator $\tilde{g} \leftarrow \mathbb{Z}_N^*$, sets $g = \tilde{g}^{\prod_{i=3}^{j_\epsilon} f_i}$ and sets public parameters $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$.
3. The challenger samples $z \leftarrow \mathbb{Z}_N^*$ and sends $\mathsf{pp}, y = z^{\prod_{i=1}^{j_\epsilon} f_i} \bmod N$ to the adversary.

Hybrid H_2: This is same as the original smoothness security game, except that the challenger always chooses source S_1.

2. The challenger then samples a generator $g \leftarrow \mathbb{Z}_N^*$ and sets public parameters $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$.
3. The challenger samples $x \leftarrow S_1$ and sends $\mathsf{pp}, y = g^{f_1 \cdot f_2 \cdot (d_0 x + d_1)\prod_{i=1}^n e_{i,x_i}} \bmod N$ to the adversary.

Lemma 6.6. *Assuming Φ-hiding assumption holds, for any PPT adversary $\mathcal{A}$, there exists a negligible function $negl(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\delta(\lambda) \geq 2\epsilon = 1/v(\lambda)$, we have $|p_0^{\mathcal{A}} - p_1^{\mathcal{A}}| \leq \epsilon/2 + negl(\lambda)$.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ that has a non-negligible advantage $\delta(\lambda)$ in smoothness game, and can distinguish between Hybrids H_0 and H_1 with probability $\epsilon/2 + \gamma$ for some non-negligible value γ. We construct a reduction algorithm $\mathcal{B}$ that breaks our strengthened hashing lemma (Theorem 3.5) and thereby breaking Φ-hiding assumption.

The adversary $\mathcal{A}$ sends two (k,n)-sources S_0, S_1 to the reduction algoritm $\mathcal{B}$. $\mathcal{B}$ plays hashing lemma game for source S_0 with the challenger $\mathcal{C}$. The hashing lemma challenger $\mathcal{C}$ sends $(N, g, a, b, \{e_{i,b}\}_{i,b}, y)$ to the reduction algorithm $\mathcal{B}$. The reduction algorithm samples a seed $\mathfrak{s} \leftarrow S$, sets $d_0 = a, d_1 = b$ and sends public parameters $\mathsf{pp} = (N, \mathfrak{s}, g, \{e_{i,b}\}_{i,b}, d_0, d_1)$, challenge y to the adverary $\mathcal{A}$. The adversary outputs a bit b'. $\mathcal{B}$ outputs b' as its guess in hashing lemma game.

Let us analyze advantage of $\mathcal{B}$ in hashing lemma game. If the challenger samples $g \leftarrow \mathbb{Z}_N^*, x \leftarrow S_0, y = g^{f_1 \cdot f_2 \cdot (ax+b) \prod_{i=1}^{n} e_{i,x_i}} \bmod N$, then $\mathcal{B}$ emulates Hybrid H_0 challenger to $\mathcal{A}$. If the challenger samples $\tilde{g} \leftarrow \mathbb{Z}_N^*, z \leftarrow \mathbb{Z}_N^*$ and sets $g = \tilde{g}^{\prod_{i=3}^{j_\epsilon} f_i}$, $y = z^{\prod_{i=1}^{j_\epsilon} f_i}$, then $\mathcal{B}$ emulates Hybrid H_1 challenger to $\mathcal{A}$. Therefore, $\mathcal{B}$ breaks hashing lemma game with advantage $|p_1^{\mathcal{A}} - p_2^{\mathcal{A}}| \geq \epsilon/2 + \gamma$. ■

Lemma 6.7. *Assuming Φ-hiding assumption holds, for any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$ satisfying $\delta(\lambda) \geq 2\epsilon = 1/v(\lambda)$, we have $|p_1^{\mathcal{A}} - p_2^{\mathcal{A}}| \leq \epsilon/2 + \text{negl}(\lambda)$.*

Proof. This proof is similar to proof of Lemma 6.6. ■

By the above 2 lemmas and triangle inequality, $\mathcal{A}$ can distinguish between Hybrids H_0 and H_2 with probability at most $\epsilon + \text{negl}(\lambda) < 2\delta/3$. This contradicts the assumption that $\mathcal{A}$ has an advantage of δ.[10] Therefore, no PPT adversary can break (k, n)-smoothness property of the above construction with non-negligible probability. ■

7 One-Way Function with Encryption from q-DBDHI Assumption

We now construct (k, n, ℓ)-OWFE from any n-DBDHI hard group generator GGen. Suppose $\mathsf{GGen}(1^\lambda)$ generates a group of size $\theta(2^m)$, the below construction requires $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$. For the sake of simplicity, we construct a OWFE scheme where the encryption algorithm outputs elements in a group. The construction can be extended to output ℓ-length bit strings by using PRGs and randomness extractors. We present a variant of this construction with longer ciphertext from n-DDHI assumption (without pairings) in the Appendix B.

$K(1^\lambda)$: Sample a group $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_T, e, p) \leftarrow \mathsf{GGen}(1^\lambda)$. Sample a generator $g \leftarrow \mathbb{G}_1$ and random values $\alpha, d_0, d_1 \leftarrow \mathbb{Z}_p$. Output the public parameters $(\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$.

$f(\mathsf{pp}, x)$: Parse public parameters pp as $\mathsf{pp} = (\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$. Let the polynomial $(d_0 x + d_1) \cdot \prod_{j=1}^{n} (\alpha + 2j + x_j) = \sum_{i=0}^{n} c_i \alpha^i$, where c_i is a function of d_0, d_1, x. Output $\prod_{i=0}^{n} \left(g^{\alpha^i}\right)^{c_i}$.

$E_1(\mathsf{pp}, (i, b); h)$: Compute and output $(h^{(\alpha+2i+b)}, i)$.

$E_2(\mathsf{pp}, (y, i, b); h)$: Compute and output $e(h, y)$.

$D(\mathsf{pp}, \mathsf{ct}, x)$: Let $\mathsf{ct} = (\mathsf{ct}', i)$. Consider the polynomial $(d_0 x + d_1) \cdot \prod_{j \neq i} (\alpha + 2j + x_j) = \sum_{j=0}^{n-1} c_j \alpha^j$, where c_j is a function of d_0, d_1, x. Compute and output $e\left(\mathsf{ct}', \prod_{j=0}^{n-1} \left(g^{\alpha^j}\right)^{c_j}\right)$.

Correctness. For any set of public parameters $\mathsf{pp} = (\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$, string $x \in \{0,1\}^n$, index $j \in [n]$ and randomness h, we have $\mathsf{ct} = E_1(\mathsf{pp}, (j, x_j); h) = (h^{(\alpha+2j+x_j)}, j)$ and $D(\mathsf{pp}, \mathsf{ct}, x) = e(g, h)^{(d_0 x + d_1) \prod_i (\alpha + 2i + x_i)} = e(h, f(\mathsf{pp}, x)) = E_2(\mathsf{pp}, (f(\mathsf{pp}, x), j, x_j); h)$.

7.1 Security

We now prove that the above construction satisfies one-wayness, encryption security and smoothness properties.

One-Wayness. We now prove that the above construction satisfies (k, n)-one-wayness property for any $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$.

[10] Note that the contradiction does not happen when δ is negligible. If δ is negligible, then j_ϵ is superpolynomial and the reduction algorithm takes superpolynomial time to execute.

Lemma 7.1. *Assuming n-DBDHI assumption holds (Assumption 3), the above construction satisfies* (k, n)*-one-wayness property for any* (k, n) *s.t.* $k \geq m + 2\lambda$ *and* $n \leq k + m - 2\lambda$ *as per Definition 2.1.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ that breaks one-wayness property of the above construction with non-negligible probability. We construct a reduction algorithm $\mathcal{B}$ that wins n-DBDHI game with non-negligible probability.

The adversary $\mathcal{A}$ first sends a (k, n)-source S to the reduction algorithm $\mathcal{B}$. The challenger then sends $(\mathcal{G}, h, h^{\beta}, h^{\beta^2}, \cdots, h^{\beta^n}, T)$ to the reduction algorithm $\mathcal{B}$. The reduction algorithm samples a string $x \leftarrow S$ and an index $j \in [n]$. It then implicity sets $\alpha = \beta - 2j - 1 + x_j$ and aborts if $\alpha + i = 0 \bmod p$ for any $i \in [2n+1]$. The reduction algorithm samples $d_0, d_1 \leftarrow \mathbb{Z}_p$, computes public parameters $\mathsf{pp} = (\mathcal{G}, h, h^{\alpha}, h^{\alpha^2}, \cdots h^{\alpha^n}, d_0, d_1)$ and sends $\mathsf{pp}, y = f(\mathsf{pp}, x)$ to the adversary $\mathcal{A}$. Note that these values can be computed given $(h, h^{\beta}, h^{\beta^2}, \cdots, h^{\beta^n})$. The adversary outputs x'. If $f(\mathsf{pp}, x) \neq f(\mathsf{pp}, x')$ or $x_j = x'_j$, then $\mathcal{B}$ aborts and guesses randomly. Otherwise, we know that $h^{(d_0 x + d_1) \cdot \prod_i (\alpha + 2i + x_i)} = h^{(d_0 x' + d_1) \prod_i (\alpha + 2i + x'_i)}$. Consider the polynomial

$$\frac{(d_0 x + d_1) \cdot \prod_i (\alpha + 2i + x_i)}{\alpha + 2j + 1 - x_j} = \frac{c}{\beta} + \sum_{i=0}^{n-1} c_i \beta^i$$

where $c, \{c_i\}_i$ are dependent only on x, d_0, d_1. We know that, $e\Big(h, h^{(d_0 x' + d_1) \cdot \prod_{i \neq j} (\alpha + 2i + x'_i)}\Big) = e(h, h)^{\frac{c}{\beta}} \cdot e\Big(h, \prod_{i=0}^{n-1} \left(g^{\beta^i}\right)^{c_i}\Big)$. $\mathcal{B}$ checks if $T^c \cdot e\Big(h, \prod_{i=0}^{n-1} \left(g^{\beta^i}\right)^{c_i}\Big) = e\Big(h, h^{(d_0 x' + d_1) \cdot \prod_{i \neq j} (\alpha + 2i + x'_i)}\Big)$ holds. If the condition holds, $\mathcal{B}$ outputs 0. Otherwise, it outputs 1.

We now analyze the advantage of $\mathcal{B}$ in n-DBDHI game. As β is sampled uniformly, α is also uniformly distributed. As p is superpolynomial in λ, $\alpha \in [1, p - 2n - 2]$ with overwhelming probability and $\mathcal{B}$ simulates one-wayness game challenger to $\mathcal{A}$ in statistically indistinguishable way. By our assumption, $f(\mathsf{pp}, x) \neq f(\mathsf{pp}, x')$ with non-negligible probability. By Lemma C.1, for any (k, n) source S s.t. $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$, the distribution of $f(\mathsf{pp}, S)$ is statistically close to uniform as α is sampled from $[1, p - 2n - 2]$. As a result, with overwhelming probability there exists many z s.t. $f(\mathsf{pp}, x) = f(\mathsf{pp}, z)$. As j is sampled uniformly from $[n]$, $f(\mathsf{pp}, x) = f(\mathsf{pp}, x'), x_j \neq x'_j$ with non-negligible probability. The final check performed by $\mathcal{B}$ always holds if $T = e(h, h)^{1/\beta}$, and holds with negligible probability if T is sampled randomly. Therefore, $\mathcal{B}$ wins n-DBDHI game with non-negligible probability. ∎

Security of Encryption. We now prove that the above construction satisfies encryption security property.

Lemma 7.2. *Assuming n-DBDHI assumption holds (Assumption 3), the above construction satisfies encryption security property as per Definition 2.2.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ that breaks encryption security of the above construction with non-negligible probability. We construct a reduction algorithm $\mathcal{B}$ that wins n-DBDHI game with non-negligible probability.

The challenger $\mathcal{C}$ first samples a group structure $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_T, e, p) \leftarrow \mathsf{GGen}(1^{\lambda})$, a generator $h \leftarrow \mathbb{G}_1$, a value $\beta \leftarrow \mathbb{Z}_p^*$ and a bit $\gamma \leftarrow \{0, 1\}$. If $\gamma = 0$, it sets $T = e(h, h)^{1/\beta}$. Otherwise, it samples $T \leftarrow \mathbb{G}_T$. The challenger then sends $(\mathcal{G}, h, h^{\beta}, h^{\beta^2}, \cdots, h^{\beta^n}, T)$ to the reduction algorithm $\mathcal{B}$. The adversary sends a string $x \in \{0, 1\}^n$ and an index j to $\mathcal{B}$. $\mathcal{B}$ samples $d_0, d_1 \leftarrow \mathbb{Z}_p$ and implicitly sets $\alpha = \beta - 2j - 1 + x_j$. It then computes public parameters $\mathsf{pp} = (\mathcal{G}, h, h^{\alpha}, h^{\alpha^2}, \cdots, h^{\alpha^n}, d_0, d_1)$, samples $\rho \leftarrow \mathbb{Z}_p$ and implicitly uses $h^{\rho/(\alpha + 2j + 1 - x_j)}$ as randomness for encryption. It computes $\mathsf{ct}^* = (h^{\rho}, j)$. Consider the polynomial

$$\frac{\rho \cdot (d_0 x + d_1) \cdot \prod_{i=1}^{n} (\alpha + 2i + x_i)}{\alpha + 2j + 1 - x_j} = \frac{c}{\beta} + \sum_{i=0}^{n-1} c_i \beta^i$$

where $c, \{c_i\}_i$ are dependent only on ρ, x, d_0, d_1. The reduction algorithm computes $k^* = T^c \cdot e\Big(h, \prod_{i=0}^{n-1} \left(h^{\beta^i}\right)^{c_i}\Big)$ and sends $\mathsf{pp}, \mathsf{ct}^*, k^*$ to the adversary. The adversary outputs a bit γ'. $\mathcal{B}$ outputs γ' as its guess in n-DBDHI game.

We now analyze the advantage of $\mathcal{B}$ in n-DBDHI game. As β is sampled uniformly, α is also uniformly distributed. As $\beta \neq 0 \mod p$ and ρ is uniformly distributed, $h^{\rho/\beta}$ is also uniformly distributed in $\mathbb{G}_1$. If $\gamma = 0$, then $(\mathsf{pp}, \mathsf{ct}^*, k^*)$ is same as $\Big(\mathsf{pp}, E_1(\mathsf{pp}, (j, 1 - x_j); \rho'), E_2(\mathsf{pp}, (f(\mathsf{pp}, x), j, 1 - x_j); \rho')\Big)$. If $\gamma = 1$, then k^* is uniformly random. As $\mathcal{A}$ distinguishes these 2 distributions with non-negligible probability, $|\Pr[\gamma' = 1 | \gamma = 0] - \Pr[\gamma' = 1 | \gamma = 1]|$ is non-negligible. Therefore, $\mathcal{B}$ breaks n-DBDHI assumption. ∎

Smoothness. We now prove that the above construction satisfies (k, n)-smoothness property for any $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$.

Lemma 7.3. *The above construction satisfies (k, n)-smoothness property for any $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$ as per Definition 2.3.*

Proof. We prove the theorem via a sequence of following hybrids.

Hybrid H_0: This is same as the original smoothness security game.

1. The adversary sends two (k, n)-sources S_0 and S_1 to the challenger. The challenger samples a group $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_T, e, p) \leftarrow \mathsf{Setup}(1^\lambda)$, a generator $g \leftarrow \mathbb{G}_1$ and exponents $d_0, d_1 \leftarrow \mathbb{Z}_p$.
2. It then samples exponent $\alpha \leftarrow \mathbb{Z}_p^*$ and computes $\mathsf{pp} = (\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$.
3. The challenger samples a bit $b \leftarrow \{0, 1\}$, a string $x \leftarrow S_b$ and sends $\mathsf{pp}, y = g^{(d_0 x + d_1) \cdot \prod_{j=1}^n (\alpha + 2j + x_j)}$ to the adversary.
4. The adversary outputs a bit b'.

Hybrid H_1: In this hybrid, the challenger samples α in public parameters from $[1, p - 2n - 2]$ instead of $\mathbb{Z}_p^*$

2. It then samples exponent $\alpha \leftarrow [1, p - 2n - 2]$ and computes $\mathsf{pp} = (\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$.

Hybrid H_2: In this hybrid, the challenger samples the challenge y uniformly at random.

3. The challenger samples $y \leftarrow \mathbb{G}_1$ and sends pp, y to the adversary.

For any adversary $\mathcal{A}$, let the probability that $b' = b$ in Hybrid H_s be $p_s^{\mathcal{A}}$. We know that, $p_2^{\mathcal{A}} = 1/2$ as y is independent of b. We prove that for every PPT adversary $\mathcal{A}$, $|p_0^{\mathcal{A}} - p_2^{\mathcal{A}}|$ is negligible.

Claim 7.1. *For every adversary $\mathcal{A}$, there exists a negligible function $negl(\cdot)$ such that for every $\lambda \in \mathbb{N}$, $|p_0^{\mathcal{A}} - p_1^{\mathcal{A}}| \leq negl(\lambda)$.*

Proof. The distribution of challenger's output is same in Hybrids H_0 and H_1, except when $\alpha \in [p - 1, p - 2n - 1]$. This event happens with probability $(2n + 1)/p$. Assuming p is super-polynomial in λ, the event $\alpha \in [p - 1, p - 2n - 1]$ happens with negligible probability. ∎

Claim 7.2. *For every adversary $\mathcal{A}$, there exists a negligible function $negl(\cdot)$ such that for every $\lambda \in \mathbb{N}$, $|p_1^{\mathcal{A}} - p_2^{\mathcal{A}}| \leq negl(\lambda)$.*

Proof. As the minimum entropy of the distribution $\{x : b \leftarrow \{0, 1\}, x \leftarrow S_b\}$ is $k \geq \log p + 2\lambda$ and as α is sampled from $[1, p - 2n - 2]$, by Lemma C.1, $(d_0 x + d_1) \cdot \prod_{j=1}^n (\alpha + 2j + x_j)$ for $x \leftarrow S_b$ is indistinguishable from uniform distribution on $\mathbb{Z}_p$. ∎

By the above claims and triangle inequality, the advantage of any adversary in the original smoothness game H_0 is negligible. ∎

8 Performance Evaluation

In this section, we discuss how our HPRG and OWFE constructions based on Φ-Hiding and D(B)DHI assumptions compare with the constructions based on DDH provided in [GH18, KW19]. The perfomance evaluation is split into two parts. First, we discuss the efficiency of our HPRG constructions and compare with existing schemes. Later, we look at out OWFE constructions and compare its performance.

8.1 Hinting PRG: Comparing with [KW19]

Here we discuss the efficiency of our HPRG constructions and compare it with existing constructions. First, we provide an asymptotic comparison and eventually give a more concrete comparison.

An asymptotic comparison. Let us start by recalling the HPRG construction based on DDH assumption, immediately derived from [KW19], which will serve as focal comparison point to prior work.[11] In their construction, the public parameters consists of $O(n^2)$ group elements, where n is the length of the HPRG seed (/number of blocks). Here each block is associated with $O(n)$ group elements. Now for computing the output of the HPRG for any given block, the evaluator simply picks half of these associated group elements and sets the output as the product of the chosen group elements. More formally, the evaluator performs $O(n)$ group operations per block during HPRG evaluation.

Comparing that to our Φ-Hiding based HPRG construction described in Section 4, the public parameters consists of $2n$ (λ-bit) prime exponents along with the RSA modulus, extractor seed, group generator, and a hash key. For evaluating a single HPRG block, the evaluator needs to perform $O(n)$ exponentiations. However, using our dynamic programming technique described in Section 4.1, we can reduce the number of exponentiation operations needed per block to grow only logarithmically in n.

In our DDHI based construction described in Section 5, the public parameters consists of n group elements along with the group generator, and a hash key. For evaluating a single HPRG block, the evaluator evaluates a degree-n polynomial symbolically and later on performs n exponentiation operations and n group operations. We summarize the asymptotic comparision of the Hinting PRG constructions in Table 1, where N and p are the recommended RSA modulus and elliptic curve field size for the target security λ.

Metric	DDH [KW19]	Φ-Hiding (§4)	DDHI (§5)
Seed length n	$\log p + 2\lambda$	$\log N + 2\lambda$	$\log p + 2\lambda$
pp size	$O(n^2)$ group elements	$O(n\lambda)$ bits and $O(1)$ elements in $\mathbb{Z}_N^*$	$O(n)$ group elements
Time (Setup)	Sampling $O(n^2)$ group elements	Sampling $O(n)$ λ-bit primes	$O(n)$ exponentiations
Time (Eval)	$O(n^2)$ group operations	$O(n \log n)$ exponentiations	$O(n^2)$ exponentiations

Table 1: Asymptotic Performance Comparision of Various Hinting PRG Constructions.

Concrete performance evaluation. The evaluations were performed on a 2015 Macbook Pro with Dual Core 2.7 GHz Intel Core i5 CPU and 8GB DDR3 RAM. We evaluated the performance of DDH and DDHI based constructions using MCL Library [Her19] (written in C++) on NIST standardized elliptic curves P-192, P-224, P-256, P-384 and P-521 providing 96, 112, 128, 192 and 260 bit security respectively. We evaluated our Φ-Hiding based construction using Flint Libary [Har10] written in C++ on 1024, 2048, 3072, 7680 and 15360 bit RSA modulus providing 80, 112, 128, 192 and 256 bit security respectively. The performance numbers are provided in Table 2. For Φ-hiding based HPRG, the numbers mentioned in the brackets indicate the

[11] Koppula and Waters [KW19] constructed Hinting PRG based on CDH assumption. In order to provide a more fair comparison with our constructions, we simplify their construction by instead relying on DDH assumption thereby removing the need for using hardcore predicates. This leads to a more efficient setup phase and shorter public parameters, and thus provides a more accurate baseline for comparing performance.

estimated evaluation times (based on benchmarks) without the dynamic programmming technique described in 4.1.

Metric	Security	DDH [KW19]	Φ-Hiding (§4)	DDHI (§5)
Seed Length n	80/96	384	1184	384
	112	448	2272	448
	128	512	3328	512
	192	768	8064	768
	256/260	1042	15872	1042
pp size	80/96	14.15 MB	0.07 MB	0.009 MB
	112	22.50 MB	0.19 MB	0.012 MB
	128	33.58 MB	0.32 MB	0.016 MB
	192	113.32 MB	1.16 MB	0.037 MB
	256/260	268.56 MB	3.05 MB	0.065 MB
Time (Setup)	80/96	7.57s	1.32s	0.0285s
	112	83.99s	6.22s	0.058s
	128	15.95s	11.86s	0.080s
	192	102.68s	98.74s	0.370s
	256/260	431.78s	443.84s	1.598s
Time (Eval)	80/96	0.042s	1.07s (112.5s w/o optimization)	14.24s
	112	0.086s	11.20s (2164.2s w/o optimization)	30.13s
	128	0.11s	40.79s (3.32 hrs w/o optimization)	46.77s
	192	0.48s	719.479s (5.46 days w/o optimization)	284.83s
	256/260	2.145s	5975.34s (84.27 days w/o optimization)	1515.68s
Eval time per block	80/96	0.109ms	0.904ms (0.11s w/o optimization)	37.09ms
	112	0.192ms	4.93ms (1.05s w/o optimization)	67.26ms
	128	0.215ms	12.26ms (3.88s w/o optimization)	91.35ms
	192	0.625ms	89.22ms (61.45s w/o optimization)	349.98ms
	256/260	2.06ms	376.47ms (473.36s w/o optimization)	1454.59ms

Table 2: Performance Comparison of Various Hinting PRG Constructions.

Now we present a few observations and interpretations of the above performance measures. We begin with the 112 bit security level as a focal point. The first thing that jumps out is that our constructions provide a rather dramatic trade-off between evaluation time versus parameter size compared to prior work. Observe that the size of the public parameters for our Φ-Hiding and DDHI based constructions are $\sim$120x and $\sim$1900x shorter than the public parameter size provided by the baseline [KW19] DDH-based construction. The setup phase of our DDHI based construction is also $\sim$1450x faster than that of the DDH-based construction. On the flip side, in our Φ-Hiding based construction the evaluation algorithm is $\sim$120x slower than that for the DDH-based construction. And, our DDHI-based construction has a further $\sim$3x slowdown compared to our Φ-Hiding based construction. From the numbers we can also see that our optimization for reusing computation across multiple RSA blocks is critical for the construction being viable as the unoptimized version would take $\sim$180x more evaluation time compared to the optimized version.

Thus, the clear trade-off between the two constructions is between optimizing the size of public parameters and reducing the running time of the HPRG evaluation algorithm. We can also observe that as we move up security parameters, our DDHI based construction begins to dominate the Φ-hiding based construction in all aspects. For 128 bit security the evaluation time of both the constructions is about even while the parameter size of the DDHI based construction is considerably lower. For larger security parameters DDHI based construction dominates completely. The reason for this is that due to number field sieve attacks, the recommended RSA modulus length (and thereby HPRG seed length n) increases super linearly with target security level for the Φ-based construction. Whereas, the recommended field size (and thereby HPRG seed length n) will increase linearly for the elliptic curve DDH and DDHI based constructions. In the Hinting

PRG context, this gives a double whammy to the Φ-hiding construction as an increase of n will increase both the number of blocks as well the cost of group exponentions required to compute each block. One can see that even for higher security parameters the amortized average computation time per block for the Φ-hiding construction remains lower, but the overall computation is higher due to the number of blocks needed.

Further reducing $|\mathsf{pp}|$ for Φ-hiding based construction. Observe that the public parameter size of the Φ-hiding based construction is dominated by the $2n$ λ-bit primes $e_{j,b}$. In order to reduce the parameter size, one could employ the PRF-trick suggested by Hohenberger and Waters [HW09] in a different context. Their idea was to generate all the prime values $e_{j,b}$ from a PRF with a randomly chosen but publicly known seed. Using this approach, we can significantly reduce the parameter size and thereby making our Φ-hiding based construction as the HPRG construction with smallest public parameter length even for large values of targetted security levels. (The reason we could use a similar approach is because in each of the hybrids in our proofs we only need to use exactly one of the $2n$ $e_{j,b}$ values for security and rest can be almost sampled arbitrarily.) An important and subsequent trade-off, however, is that since the $e_{j,b}$ values now would need to be computed on-the-fly during evaluation each time. This involves performing prime searches as part of the Eval algorithm and can lead to a further increase in evaluation time. Moreover, the proof of our construction would need to be adapted in a way similar to [HW09] to accomodate this change. One would additionally need to be extra careful in extending our hashing lemma (Theorem 3.2) to incorporate this change for the entire proof to work. Below in Table 3, we provide the performance metrics for such a modified construction.

Metric	80	112	128	192	256
Seed Length n	1184	2272	3328	8064	15872
pp size	0.77 KB	1.54 KB	2.30 KB	5.76 KB	11.52 KB
Time (Setup)	0.0079s	0.067s	0.247s	6.73s	82.61s
Time (Eval)	2.70s	17.60s	52.61s	813.28s	6342.10s

Table 3: Performance Metrics of the Φ-Hiding Based HPRG optimized for small pp size

CCA Security via Hinting PRGs. Although Hinting PRGs are an elegant cryptographic primitive, and therefore coming up with more efficient constructions is interesting in its own right. So far, the most prominent application of Hinting PRGs has been in upgrading any CPA-secure PKE/ABE scheme to be CCA-secure [KW19]. Here we thereby analyze how our Hinting PRG constructions affect the performance of the CPA to CCA-secure PKE/ABE transformation, and briefly compare with other approaches for similar transformations.

Let us first briefly recall some important aspects of the CCA-secure PKE construction proposed by [KW19][12]. In their construction, the setup involves performing an HPRG Setup for sampling HPRG public parameters pp which are included as part of the public encryption key. During encryption, the encryptor runs the HPRG Eval algorithm once for each block and the size of the ciphertext also grows linearly with the seed length n. Now observe that the recommended HPRG seed length n, for any given security parameter, is lowest for the DDH and DDHI-based constructions. Additionally, DDHI-based construction leads to shorter public parameters. Thus, if one uses the DDHI-based HPRG construction proposed in this work then it leads to much smaller public encryption key and ciphertext sizes with the trade-off being higher encryption/decryption times. (On the other hand, if the goal is to minimize the size of the public encryption key, then our Φ-Hiding construction with the aforementioned optimization technique could be used instead.) In conclusion, the trade-off between public parameter size and evaluation time in the HPRG constructions carries forward to a trade-off between encryption key/ciphertext sizes and encryption/decryption times in the resultant CCA-secure construction.

[12] In the original construction by [KW19], the setup algorithm of CCA-secure PKE samples $2n$ public-secret key pairs of the underlying CPA-secure PKE. Later in [KMT19], it was suggested that one could instead sample only 2 public-secret key pairs of the underlying CPA-secure PKE. For an adequate analysis, here we always consider the optimized [KW19] construction as suggested in [KMT19].

Comparing with [KMT19] (CCA Security via KDM Security). In a follow-up work to [KW19], Kitagawa et al. [KMT19] provided a similar transformation for achieving CCA security but by relying on Key Dependent Message secure SKE [BRS02, HK07, HU08, BPS08] instead of Hinting PRGs.A natural question would be whether this approach would outperform the [KW19] construction after we plug in the HPRGs proposed in this work. It turns out that the [KW19] construction is asymptotically a lot more efficient than [KMT19] in terms of key sizes, ciphertext sizes, setup, encryption and decryption times. The reason is that in [KMT19] construction, most of the efficiency metrics (such as public key and ciphertext size, setup/encryption/decryption times) grow linearly with the key length ℓ of the underlying KDM-secure system. In most existing KDM-secure schemes [BHHO08, BG10, BLSV17], the key length ℓ is at least $O(\lambda^2)$ bits. Therefore, this approach leads to much worse (a quadratic slowdown) parameters when compared with HPRG-based constructions. Thus, this further motivates the problem of improving efficiency of Hinting PRGs.

8.2 OWF with Encryption: Comparing with [GH18]

We now discuss the efficiency of our OWFE constructions and compare it with existing constructions. First, we provide an asymptotic comparison and then give a more concrete performance evaluation.

An asymptotic comparison. In the [GH18] construction, the public parameters consist of $O(n)$ group elements, where n is at least $\log p + 2\lambda$, and p is the group size. The function evaluation and decryption algorithm performs $O(n)$ group operations. The E_1 algorithm performs $O(n)$ exponentiations and outputs a ciphertext containing $O(n)$ group elements. The E_2 algorithm performs one exponentiation and outputs a key containing one group element.

Comparing that to our Φ-Hiding based OWFE construction described in Section 6, the public parameters consists of $2n$ (λ-bit) prime exponents along with the RSA modulus N, extractor seed, group generator, and a hash key. The function evaluation and decryption algorithm performs $O(n)$ exponentiations with λ-bit exponents, where n is at least $\log N + 2\lambda$. Both E_1 and E_2 algorithms perform a single exponentiation, and outputs a ciphertext and key containing just one group element, respectively.

In our DDHI based construction described in Appendix B, the setup phase performs $O(n)$ exponentiations and outputs public parameters containing n group elements, where n is at least $\log p + 2\lambda$, and p is the group size. The function evaluation and decryption algorithms evaluate a degree-n polynomial symbolically and later on performs n exponentiation operations and n group operations. The E_1 algorithm performs $O(n)$ exponetiations and outputs a ciphertext containing $O(n)$ group elements. The E_2 algorithm performs one exponentiation and outputs a key containing 1 group element. We also provide a more efficient OWFE construction Section 7 by relying on bilinear maps and prove it secureunder DBDHI. It is similar to the DDHI based OWFE, except that E_1 algorithm only performs $O(1)$ exponentiations, E_2 and decryption algorithms additionally perform a pairing operation, and ciphertext contains only one group element.

Concrete performance evaluation. The evaluations are performed in a computational environment similar to that of HPRG evaluation. In addition, we evaluated the performance of DBDHI based OWFE using MCL Library [Her19] on BN-254, BN-381, BN-462 pairing-friendly elliptic curves [BN05] (providing 100, 128, 140 bit security after the recent tower number field sieve attacks [KB16, MSS16, FK18]).

It turns out that the baseline DDH based OWFE offers the shortest setup, evaluation, and decryption times. Whereas the Φ-hiding based OWFE outperforms in terms of E_1 time and ciphertext size. And, due to smaller group size (and thereby smaller n), DBDHI based OWFE leads to shortest E_1 time and ciphertext size. Lastly, for shortest E_2 time and key size, both the DDH and DDHI based constructions are equally useful. The concrete performance numbers are provided in Table 4.

Note that even though both DDHI and DBDHI based OWFE schemes have the same one-way function, DDHI based scheme has faster evaluation time. In fact, the DDHI based construction is more efficient than DBDHI construction in all aspects other than E_1 time and ciphertext size. This is because the recommended group size of pairing based elliptic curves grows super linearly in the security parameter due to the number

Metric	Security	DDH [GH18]	Φ-Hiding (§6)	DDHI (§B)	DBDHI (§7)
pp Size	80/96/BN254	18.4 KB	71.8 KB	9.2 KB	14.4 KB
	112	25.1 KB	192.4 KB	12.6 KB	-
	128/BN381	32.7 KB	321.8 KB	16.4 KB	30.4 KB
	140/BN462	-	-	-	42.85 KB
	192	73.7 KB	1167 KB	36.9 KB	-
	256	131.1 KB	3059 KB	65.7 KB	-
ct Size	80/96/BN254	18.4KB	128 Bytes	9.2 KB	64 Bytes
	112	25 KB	256 Bytes	12.4 KB	-
	128/BN381	32.7KB	384 Bytes	16.3 KB	96 Bytes
	140/BN462	-	-	-	116 Bytes
	192	73.68KB	960 Bytes	36.9 KB	-
	256	131KB	1920 Bytes	65.5 KB	-
Key Size	80/96/BN254	24 Bytes	128 Bytes	24 Bytes	381 Bytes
	112	28 Bytes	256 Bytes	28 Bytes	-
	128/BN381	32 Bytes	384 Bytes	32 Bytes	573 Bytes
	140/BN462	-	-	-	593 Bytes
	192	48 Bytes	960 Bytes	48 Bytes	-
	256	64 Bytes	1920 Bytes	64 Bytes	-
Time (Setup)	80/96/BN254	0.0096s	1.40s	0.026s	0.0435s
	112	0.093s	6.69s	0.052s	-
	128/BN381	0.016s	12.43s	0.070s	0.158s
	140/BN462	-	-	-	0.493s
	192	0.065s	101.38s	0.307s	-
	256	0.203s	475.55s	1.326s	-
Time (f)	80/96/BN254	0.0001s	0.11s	0.037s	0.059s
	112	0.0002s	1.06s	0.068s	-
	128/BN381	0.0002s	3.67s	0.090s	0.19s
	140/BN462	-	-	-	0.54s
	192	0.0006s	59.14s	0.353s	-
	256	0.0020s	473.36s	1.41s	-
Time (E_1)	80/96/BN254	49.1ms	0.69ms	29.44ms	0.188ms
	112	100.87ms	3.10ms	56.80ms	-
	128/BN381	134.90ms	9.40ms	76.40ms	0.45ms
	140/BN462	-	-	-	1.435ms
	192	600.84ms	106.57ms	326.49ms	-
	256	2590.14ms	601.5ms	1357.93ms	-
Time (E_2)	80/96/BN254	0.067ms	0.40ms	0.066ms	0.68ms
	112	0.12ms	2.80ms	0.11ms	-
	128/BN381	0.14ms	8.38ms	0.136ms	1.79ms
	140/BN462	-	-	-	4.52ms
	192	0.40ms	99.50ms	0.40ms	-
	256	1.26ms	600.03ms	1.29ms	-
Time (D)	80/96/BN254	0.0001s	0.109s	0.036s	0.059s
	112	0.0003s	1.09s	0.067s	-
	128/BN381	0.0003s	3.57s	0.090s	0.19s
	140/BN462	-	-	-	0.54s
	192	0.00083s	58.96s	0.355s	-
	256	0.00286s	466.84s	1.41s	-

Table 4: Concrete performance evaluation of various OWFE constructions

field sieve attacks. And, the function evaluation and decryption procedures of Φ-hiding based scheme performs $O(n)$ exponentiations, when compared to $O(n)$ group operations performed by other schemes. As a result, Φ-hiding based scheme has the slowest function evaluation and decryption procedures.

Deterministic Encryption from OWFE. A very interesting application of OWFE is of deterministic encryption as shown by [GGH19]. In the determinstic encryption scheme of [GGH19], the setup phase invokes OWFE setup phase once and E_1 algorithm $O(\ell)$ times, where ℓ is proportional to the length of message being encrypted. The encryption key includes OWFE public parameters and $O(\ell)$ OWFE ciphertexts. The encryption algorithm invokes OWFE f algorithm once and OWFE D algorithm $O(\ell)$ times. The decryption algorthm invokes OWFE E_2 algorithm $O(\ell)$ times. Consequently, our DBDHI based OWFE leads to a determinstic encryption scheme with much smaller public parameters and setup time. Concretely, at 128 bit security, the setup phase and public parameters of our DBDHI based deterministic encryption scheme for 128 bit messages is more than 200x faster and 240x shorter respectively than the baseline DDH based determinsitic encryption described in [GGH19].

References

[AMP19] Navid Alamati, Hart Montgomery, and Sikhar Patranabis. Symmetric primitives with structured secrets. In *Advances in Cryptology - CRYPTO 2019*, 2019.

[AMPR19] Navid Alamati, Hart Montgomery, Sikhar Patranabis, and Arnab Roy. Minicrypt primitives with algebraic structure and applications. In *Advances in Cryptology - EUROCRYPT 2019*, pages 55–82, 2019.

[ATSM09] Man Ho Au, Patrick P. Tsang, Willy Susilo, and Yi Mu. Dynamic universal accumulators for DDH groups and their application to attribute-based anonymous credential systems. In *Topics in Cryptology - CT-RSA 2009*, pages 295–308, 2009.

[BB04] Dan Boneh and Xavier Boyen. Efficient selective-ID secure Identity-Based Encryption without random oracles. In *EUROCRYPT '04*, volume 3027 of LNCS, pages 223–238, 2004.

[BdM93] Josh Cohen Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In *Advances in Cryptology - EUROCRYPT '93*, pages 274–285, 1993.

[BG10] Zvika Brakerski and Shafi Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability (or: Quadratic residuosity strikes back). *IACR Cryptology ePrint Archive*, 2010:226, 2010.

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-Secure Encryption from Decision Diffie-Hellman. In *CRYPTO '08*, volume 5157 of LNCS, pages 108–125, 2008.

[BLSV17] Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. Anonymous ibe, leakage resilience and circular security from new assumptions. Cryptology ePrint Archive, Report 2017/967, 2017. http://eprint.iacr.org/2017/967.

[BN05] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In *Selected Areas in Cryptography, 12th International Workshop, SAC 2005*, pages 319–331, 2005.

[BP97] Niko Baric and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology - EUROCRYPT '97*, pages 480–494, 1997.

[BPS08] M. Backes, B. Pfitzmann, and A. Scedrov. Key-dependent message security under active attacks -BRSIM/UC-soundness of Dolev-Yao-style encryption with key cycles. *J.of Comp.Security*, 16(5):497–530, 2008.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002*, pages 62–75, 2002.

[CDG+17] Chongwon Cho, Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, and Antigoni Polychroniadou. Laconic oblivious transfer and its applications. In *Advances in Cryptology - CRYPTO 2017*, pages 33–65, 2017.

[CF13] Dario Catalano and Dario Fiore. Vector commitments and their applications. In *Public-Key Cryptography - PKC 2013*, pages 55–72, 2013.

[CKS09] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In *Public Key Cryptography - PKC 2009*, pages 481–500, 2009.

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Advances in Cryptology - CRYPTO 2002*, pages 61–76, 2002.

[CMS99] Christian Cachin, Silvio Micali, and Markus Stadler. Computationally private information retrieval with polylogarithmic communication. In *EUROCRYPT '99*, pages 402–414, 1999.

[DG17a] Nico Döttling and Sanjam Garg. Identity-based encryption from the diffie-hellman assumption. In *Advances in Cryptology - CRYPTO 2017*, pages 537–569, 2017.

[DG17b] Nico Dttling and Sanjam Garg. From selective ibe to full ibe and selective hibe. TCC, 2017.

[DGHM18] Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, and Daniel Masny. New constructions of identity-based and key-dependent message secure encryption schemes. In *Public-Key Cryptography - PKC 2018*, pages 3–31, 2018.

[DGI+19] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In *Advances in Cryptology - CRYPTO 2019*, 2019.

[Dir37] PG Lejeune Dirichlet. Beweis eines satzes über die arithmetische progression. *Bericht über die Verhandlungen der königlich Presussischen Akademie der Wissenschaften Berlin*, pages 309–312, 1837.

[DlVP97] Charles-Jean De la Vallée Poussin. *Recherches analytiques sur la théorie des nombres premiers.* Hayez, Imprimeur de l'Académie royale de Belgique, 1897.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, pages 523–540, 2004.

[FK18] Georgios Fotiadis and Elisavet Konstantinou. TNFS resistant families of pairing-friendly elliptic curves. *IACR Cryptology ePrint Archive*, 2018:1017, 2018.

[GGH19] Sanjam Garg, Romain Gay, and Mohammad Hajiabadi. New techniques for efficient trapdoor functions and applications. In *Advances in Cryptology - EUROCRYPT 2019*, 2019.

[GH18] Sanjam Garg and Mohammad Hajiabadi. Trapdoor functions from the computational diffie-hellman assumption. In *CRYPTO 2018*, pages 362–391, 2018.

[GHM+19] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, Ahmadreza Rahimi, and Sruthi Sekar. Registration-based encryption from standard assumptions. In *Public-Key Cryptography - PKC 2019*, pages 63–93, 2019.

[GHMR18] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ahmadreza Rahimi. Registration-based encryption: Removing private-key generator from IBE. In *Theory of Cryptography - 16th International Conference, TCC 2018*, pages 689–718, 2018.

[GOS18] Sanjam Garg, Rafail Ostrovsky, and Akshayaram Srinivasan. Adaptive garbled RAM from laconic oblivious transfer. In *Advances in Cryptology - CRYPTO 2018*, pages 515–544, 2018.

[GR04] Craig Gentry and Zulfikar Ramzan. Rsa accumulator based broadcast encryption. In *International Conference on Information Security*, pages 73–86. Springer, 2004.

[GS17] Sanjam Garg and Akshayaram Srinivasan. Garbled protocols and two-round MPC from bilinear maps. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 588–599, 2017.

[GS18] Sanjam Garg and Akshayaram Srinivasan. Adaptively secure garbling with near optimal online complexity. In *Advances in Cryptology - EUROCRYPT 2018*, pages 535–565, 2018.

[Har10] W. B. Hart. Fast library for number theory: An introduction. In *Proceedings of the Third International Congress on Mathematical Software*, ICMS'10, pages 88–91, Berlin, Heidelberg, 2010. Springer-Verlag. http://flintlib.org.

[Her19] Herumi. A portable and fast pairing-based cryptography library. https://github.com/herumi/mcl, 2019.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[HK07] Shai Halevi and Hugo Krawczyk. Security under key-dependent inputs. In *ACM CCS '07*, pages 466–475, 2007.

[HOR15] Brett Hemenway, Rafail Ostrovsky, and Alon Rosen. Non-committing encryption from Φ-hiding. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015*, pages 591–608, 2015.

[HU08] Dennis Hofheinz and Dominique Unruh. Towards key-dependent message security in the standard model. In *EUROCRYPT '08*, volume 4965 of LNCS, pages 108–126, 2008.

[HW09] Susan Hohenberger and Brent Waters. Short and stateless signatures from the RSA assumption. In *Advances in Cryptology - CRYPTO 2009*, pages 654–670, 2009.

[KB16] Taechan Kim and Razvan Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In *Advances in Cryptology - CRYPTO 2016*, pages 543–571, 2016.

[KMT19] Fuyuki Kitagawa, Takahiro Matsuda, and Keisuke Tanaka. Cca security and trapdoor functions via key-dependent-message security. In *Crypto '19 (To Appear)*, 2019.

[KNYY19] Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Designated verifier/prover and preprocessing nizks from diffie-hellman assumptions. In *Advances in Cryptology - EUROCRYPT 2019*, pages 622–651, 2019.

[KW19] Venkata Koppula and Brent Waters. Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. In *Advances in Cryptology - CRYPTO 2019*, 2019.

[LQR+19] Alex Lombardi, Willy Quach, Ron D. Rothblum, Daniel Wichs, and David J. Wu. New constructions of reusable designated-verifier nizks. In *Advances in Cryptology - EUROCRYPT 2019*, 2019.

[MSS16] Alfred Menezes, Palash Sarkar, and Shashank Singh. Challenges with assessing the impact of NFS advances on the security of pairing-based cryptography. In *Paradigms in Cryptology - Mycrypt 2016. Malicious and Exploratory Cryptology*, pages 83–108, 2016.

[New80] Donald J Newman. Simple analytic proof of the prime number theorem. *The American Mathematical Monthly*, 87(9):693–696, 1980.

[Ngu05] Lan Nguyen. Accumulators from bilinear pairings and applications. In *Topics in Cryptology - CT-RSA 2005*, pages 275–292, 2005.

[QRW19] Willy Quach, Ron D. Rothblum, and Daniel Wichs. Reusable designated-verifier nizks for all NP from CDH. In *Advances in Cryptology - EUROCRYPT 2019*, pages 593–621, 2019.

[Sha83] Adi Shamir. On the generation of cryptographically strong pseudorandom sequences. *ACM Trans. Comput. Syst.*, 1(1):38–44, 1983.

[Sop10] Ivan Soprounov. A short proof of the prime number theorem for arithmetic progressions. *preprint*, 2010.

[STY00] Tomas Sander, Amnon Ta-Shma, and Moti Yung. Blind, auditable membership proofs. In *Financial Cryptography, 4th International Conference, FC 2000 Anguilla, British West Indies, February 20-24, 2000, Proceedings*, pages 53–71, 2000.

[Tao09] Terence Tao. The prime number theorem in arithmetic progressions, and dueling conspiracies. Terry Tao Blog Post, 24 September, 2009. https://terrytao.wordpress.com/2009/09/24/the-prime-number-theorem-in-arithmetic-progressions-and-dueling-conspiracies/.

[Zag97] Don Zagier. Newman's short proof of the prime number theorem. *The American mathematical monthly*, 104(8):705–708, 1997.

[Zha16] Mark Zhandry. The magic of elfs. In *Advances in Cryptology - CRYPTO 2016*, pages 479–508, 2016.

A Hinting PRG from One Way Function with Encryption

In this section, we generically construct (n, ℓ)-hinting PRG for any polynomials $n(\cdot), \ell(\cdot)$ from $(n-1, n, \ell)$-smooth recyclable OWFE, an extractor and a standard pseudorandom generator. Let $\mathcal{OWFE} = (K, f, E_1, E_2, D)$ be any $(n-1, n, \ell)$-smooth recyclable OWFE with randomness space $\mathcal{R}$, and let co-domain of f be $\mathcal{C}$. Let the min-entropy of distribution $\{f(x) : x \leftarrow \{0,1\}^n\}$ be k. As f is one-way, we know that $k = \Omega(\log \lambda)$. Let $\mathsf{Ext} : \mathcal{C} \times \mathcal{S} \to \mathcal{W}$ be a (k, ϵ) extractor, where ϵ is negligibly small in security parameter. Let $\mathsf{PRG} : \mathcal{W} \to \{0,1\}^\ell$ be a pseudorandom generator. We construct Hinting PRG with $(\mathsf{Setup}, \mathsf{Eval})$ as follows.

$\mathsf{Setup}(1^\lambda)$: First sample $\mathsf{pp}' \leftarrow K(1^\lambda)$. Then sample $\{\rho_{i,b}\}_{i \in [n], b \in \{0,1\}}$ uniformly at random from $\mathcal{R}$ and compute $\mathsf{ct}_{i,b} = E_1(\mathsf{pp}', (i,b); \rho_{i,b})$ for $i \in [n], b \in \{0,1\}$. Sample an extractor seed $\mathfrak{s} \leftarrow \mathcal{S}$. Output public parameters $\mathsf{pp} = (\mathsf{pp}', \{\mathsf{ct}_{i,b}\}_{i,b}, \mathfrak{s})$.

$\mathsf{Eval}(\mathsf{pp}, x, i)$: Parse pp as $(\mathsf{pp}', \{\mathsf{ct}_{i,b}\}_{i,b}, \mathfrak{s})$. If $i = 0$, output $\mathsf{PRG}(\mathsf{Ext}(f(\mathsf{pp}', x), \mathfrak{s}))$. Otherwise, output $D(\mathsf{ct}_{i,x_i}, x)$.

A.1 Security

We now prove that the above scheme is a secure hinting PRG. Formally, we prove

Theorem A.1. *If $\mathcal{OWFE}$ is an $(n-1, n, \ell)$-smooth recyclable OWFE,* Ext *is a strong seeded extractor with appropriate parameters and* PRG *is a secure pseudorandom generator, then the above construction is a secure hinting PRG as per Definition 2.4.*

Proof. We prove the above theorem via a sequence of following hybrids. First, we modify the challenger to use E_2 algorithm to generate HPRG challenge. We then switch the randomly sampled $y_{i,1-x_i}$ values to be sampled in a structured way i.e., $y_{i,1-x_i} = E_2(\mathsf{pp}, (f(\mathsf{pp}, x), i, 1-x_i); \rho_{i,1-x_i})$. We then switch each of the challenge elements to random by using OWFE encryption security.

Hybrid H_0: This game corresponds to the original hinting prg game in which the challenger always chooses $\beta = 0$.

1. The challenger first samples OWFE public parameters $\mathsf{pp}' \leftarrow K(1^\lambda)$ and randomness $\{\rho_{i,b}\}_{i\in[n],b\in\{0,1\}}$. It then computes $\mathsf{ct}_{i,b} = E_1(\mathsf{pp}', (i,b); \rho_{i,b})$ for $i \in [n], b \in \{0,1\}$. The challenger then samples an extractor seed $\mathfrak{s}$.
2. The challenger samples a bit string $x \leftarrow \{0,1\}^n$, computes the challenge $y_0 = \mathsf{PRG}(\mathsf{Ext}(f(\mathsf{pp}', x), \mathfrak{s})), y_{i,x_i} = D(\mathsf{ct}_{i,x_i}, x), y_{i,\overline{x_i}} \leftarrow \{0,1\}^\ell$ for $i \in [n]$.
3. It sends public parameters $\mathsf{pp} = (\mathsf{pp}', \{ct_{i,b}\}_{i,b}, \mathfrak{s})$ and challenge $y = (y_0, \{y_{i,b}\}_{i,b})$ to the adversary.
4. The adversary outputs a bit β'.

Hybrid H_1: This is same as previous hybrid, except that the challenger computes y_{i,x_i} using E_2 algorithm.

2. The challenger samples a bit string $x \leftarrow \{0,1\}^n$, computes $t = f(\mathsf{pp}', x)$ and the challenge $y_0 = \mathsf{PRG}(\mathsf{Ext}(t, \mathfrak{s})), y_{i,x_i} = E_2(\mathsf{pp}', (t, i, x_i); \rho_{i,x_i}), y_{i,\overline{x_i}} \leftarrow \{0,1\}^\ell$ for $i \in [n]$.

We now define a sequence of $n+1$ hybrids. For the sake of simplicity, let Hybrid $H_{2.0}$ be same as Hybrid H_1.

Hybrid $H_{2.j}$ ($j \in [n]$): This is same as previous hybrid, except that the challenger computes $y_{i,\overline{x_i}}$ for $i \le j$ differently.

2. The challenger samples a bit string $x \leftarrow \{0,1\}^n$, computes $t = f(\mathsf{pp}', x)$ and the challenge $y_0 = \mathsf{PRG}(\mathsf{Ext}(t, \mathfrak{s}))$, $y_{i,b} = E_2(\mathsf{pp}', (t, i, b); \rho_{i,b})$ for all (i,b) s.t. $i \le j$ or $b = x_i$ and samples $y_{i,1-x_i} \leftarrow \{0,1\}^\ell$ for all $i > j$.

We now define a sequence of $2n+1$ hybrids. For the sake of simplicity, let Hybrid $H_{3.0.1}$ be same as Hybrid $H_{2.n}$.

Hybrid $H_{3.j.b'}$ ($j \in [n], b' \in \{0,1\}$): In this hybrid, the challenger samples x s.t. $x_j = 1 - b'$. It also samples $y_{i,b}$ uniformly at random for all $(i,b) \prec (j,b')$.

2. The challenger samples a bit string $x \leftarrow \{0,1\}^n$ s.t. $x_j = 1 - b'$, computes $t = f(\mathsf{pp}', x)$ and the challenge $y_0 = \mathsf{PRG}(\mathsf{Ext}(t, \mathfrak{s}))$, computes $y_{i,b} = E_2(\mathsf{pp}', (t, i, b); \rho_{i,b})$ for $(i,b) \succeq (j,b')$ and samples $y_{i,b} \leftarrow \{0,1\}^\ell$ for $(i,b) \prec (j,b')$.

We now define a sequence of $2n$ hybrids.

Hybrid $H_{4.j.b'}$: This hybrid is same as Hybrid $H_{3.j.b'}$, except that the challenger samples $y_{j,b'}$ at random.

2. The challenger samples a bit string $x \leftarrow \{0,1\}^n$ s.t. $x_j = 1 - b'$, computes $t = f(\mathsf{pp}', x)$ and the challenge $y_0 = \mathsf{PRG}(\mathsf{Ext}(t, \mathfrak{s}))$, computes $y_{i,b} = E_2(\mathsf{pp}', (t, i, b); \rho_{i,b})$ for $(i,b) \succ (j,b')$ and samples $y_{i,b} \leftarrow \{0,1\}^\ell$ for $(i,b) \preceq (j,b')$.

Hybrid H_5: This hybrid is same as Hybrid $H_{4.n.1}$, except that the challenger samples $x \leftarrow \{0,1\}^n$ without any restriction.

2. The challenger samples a bit string $x \leftarrow \{0,1\}^n$, computes $t = f(\mathsf{pp}', x)$ and the challenge $y_0 = \mathsf{PRG}(\mathsf{Ext}(t, \mathfrak{s}))$, $y_{i,b} \leftarrow \{0,1\}^\ell$ for all $i \in [n], b \in \{0,1\}$.

Hybrid H_6: This is same as Hybrid $H_{5.n.1}$, except that the challenger samples t uniformly at random.

2. The challenger samples $t \leftarrow \mathcal{W}$, computes $y_0 = \mathsf{PRG}(t)$, $y_{i,b} \leftarrow \{0,1\}^\ell$ for all $i \in [n], b \in \{0,1\}$.

Hybrid H_7: This is same as Hybrid H_6, except that the challenger samples y_0 uniformly at random.

2. The challenger samples $y_0 \leftarrow \{0,1\}^\ell$, $y_{i,b} \leftarrow \{0,1\}^\ell$ for all $i \in [n], b \in \{0,1\}$.

Note that hybrid H_0 is the original hinting PRG game when challenger always chooses $\beta = 0$ and hybrid H_7 is the original hinting PRG game when challenger always chooses $\beta = 1$. We prove that these 2 hybrids are computationally indistinguishable using the following lemmas. For any PPT adversary $\mathcal{A}$, let $p_s^{\mathcal{A}}$ be the probability that $\mathcal{A}$ outputs 1 in Hybrid H_s.

Lemma A.1. *Assuming $\mathcal{OWFE}$ is perfectly correct, for any adversary $\mathcal{A}$, we have $p_0^{\mathcal{A}} = p_1^{\mathcal{A}}$.*

Proof. Assuming $\mathcal{OWFE}$ is perfectly correct, the distribution of the challenger's output is same in hybrids H_0 and H_1. ■

Lemma A.2. *Assuming $\mathcal{OWFE}$ has secure encryption property, for any PPT adversary $\mathcal{A}$, and index $j \in [n]$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have $|p_{2.j}^{\mathcal{A}} - p_{2.j-1}^{\mathcal{A}}| \leq$ negl(λ).*

Proof. Suppose there exists a PPT Adversary $\mathcal{A}$, and an index $j \in [n]$ such that $|p_{2.j}^{\mathcal{A}} - p_{2.j-1}^{\mathcal{A}}|$ is non-negligible. We construct a reduction algorithm $\mathcal{B}$ that breaks OWFE security of encryption.

The reduction algorithm first samples $x \leftarrow \{0,1\}^n$ and sends (x, j) to challenger $\mathcal{C}$. The challenger samples OWFE public parameters $\mathsf{pp}' \leftarrow K(1^\lambda)$, a bit $\alpha \leftarrow \{0,1\}$, randomness ρ and computes $\mathsf{ct}^* = E_1(\mathsf{pp}', j, 1 - x_j; \rho)$. If $\alpha = 1$, it computes $y^* = E_2(\mathsf{pp}', f(\mathsf{pp}', x), j, 1 - x_j; \rho)$. Otherwise, it samples $y^* \leftarrow \{0,1\}^\ell$. The challenger sends $(\mathsf{pp}', \mathsf{ct}^*, y^*)$ to $\mathcal{B}$. $\mathcal{B}$ then samples randomness $\rho_{i,b} \leftarrow \mathcal{R}$ and computes $\mathsf{ct}_{i,b} = E_1(\mathsf{pp}', (i,b); \rho_{i,b})$ for $(i,b) \neq (j, 1 - x_j)$. It then initializes $\mathsf{ct}_{j,\overline{x_j}} = \mathsf{ct}^*$, $y_{j,\overline{x_j}} = y^*$. $\mathcal{B}$ then samples an extractor seed $\mathfrak{s} \leftarrow \mathcal{S}$, computes $t = f(\mathsf{pp}', x)$, $y_0 = \mathsf{PRG}(\mathsf{Ext}(t, \mathfrak{s}))$, and $y_{i,b} = E_2(\mathsf{pp}', (t, i, b); \rho_{i,b})$ for all (i,b) s.t. $i < j \vee b = x_i$. It then samples $y_{i,1-x_i} \leftarrow \{0,1\}^\ell$ for $i > j$ and sends public parameters $\mathsf{pp} = (\mathsf{pp}', \{ct_{i,b}\}_{i,b}, \mathfrak{s})$ and challenge $(y_0, \{y_{i,b}\}_{i,b})$ to the adversary $\mathcal{A}$. The adversary outputs a bit β'. The reduction algorithm $\mathcal{B}$ outputs β' as its guess in OWFE game.

Note that if $\alpha = 0$, $\mathcal{B}$ emulates Hybrid $H_{2.j-1}$ challenger to $\mathcal{A}$. If $\alpha = 1$, $\mathcal{B}$ emulates Hybrid $H_{2.j}$ challenger to $\mathcal{A}$. Moreover, $\mathcal{B}$ acts as a valid adversary in OWFE encryption security game. By our assumption, $|p_{2.j}^{\mathcal{A}} - p_{2.j-1}^{\mathcal{A}}| = |\Pr[\beta' = 1|\alpha = 0] - \Pr[\beta' = 1|\alpha = 1]|$ is non-negligible. Therefore, $\mathcal{B}$ breaks OWFE encryption security. ■

Lemma A.3. *Assuming $(n-1, n)$-smoothness of $\mathcal{OWFE}$ scheme, for any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have $|p_{2.n}^{\mathcal{A}} - p_{3.1.0}^{\mathcal{A}}| \leq$ negl(λ).*

Proof. Suppose there exists a PPT Adversary $\mathcal{A}$ such that $|p_{2.n}^{\mathcal{A}} - p_{3.1.0}^{\mathcal{A}}|$ is non-negligible. We construct a reduction algorithm $\mathcal{B}$ that breaks smoothness property of $\mathcal{OWFE}$.

Let $X_0 = \{0,1\}^n$ and $X_1 = \{x \in \{0,1\}^n : x_1 = 1\}$. Let S_0 and S_1 be uniform distributions on sets X_0 and X_1 respectively. Note that both S_0 and S_1 have min-entropy at least $n - 1$. The reduction algorithm $\mathcal{B}$ sends S_0 and S_1 to the challenger $\mathcal{C}$ of smoothness game. The challenger $\mathcal{C}$ samples a bit $\alpha \leftarrow \{0,1\}$, samples $x \leftarrow S_\alpha$, samples OWFE public parameters $\mathsf{pp}' \leftarrow K(1^\lambda)$, and sends $(\mathsf{pp}', t = f(\mathsf{pp}', x))$ to $\mathcal{B}$. The reduction algorithm $\mathcal{B}$ samples an extractor seed $\mathfrak{s}$, randomness $\rho_{i,b} \leftarrow \mathcal{R}$, computes $\mathsf{ct}_{i,b} = E_1(\mathsf{pp}', (i,b); \rho_{i,b})$, $y_{i,b} = E_2(\mathsf{pp}', (t, i, b); \rho_{i,b})$ for $i \in [n], b \in \{0,1\}$ and sets $y_0 = \mathsf{PRG}(\mathsf{Ext}(t, \mathfrak{s}))$. $\mathcal{B}$ sends public parameters $\mathsf{pp} = (\mathsf{pp}', \{\mathsf{ct}_{i,b}\}_{i,b}, \mathfrak{s})$ and challenge $(y_0, \{y_{i,b}\}_{i,b})$ to $\mathcal{A}$. The adversary outputs a bit β', which $\mathcal{B}$ outputs as its guess in smoothness game.

Note that if $\alpha = 0$, $\mathcal{B}$ emulates Hybrid $H_{2.n}$ challenger to $\mathcal{A}$. If $\alpha = 1$, $\mathcal{B}$ emulates Hybrid $H_{3.1.0}$ challenger to $\mathcal{A}$. Moreover, $\mathcal{B}$ acts as a valid adversary in OWFE encryption security game. By our assumption, $|p_{2.n}^{\mathcal{A}} - p_{3.1.0}^{\mathcal{A}}| = |\Pr[\beta' = 1|\alpha = 0] - \Pr[\beta' = 1|\alpha = 1]|$ is non-negligible. Therefore, $\mathcal{B}$ breaks $(n-1, n)$-smoothness property of $\mathcal{OWFE}$. ■

Lemma A.4. *Assuming $\mathcal{OWFE}$ has secure encryption property, for any PPT adversary $\mathcal{A}$, index $j \in [n]$, bit $b' \in \{0,1\}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have $|p_{3.j.b'}^{\mathcal{A}} - p_{4.j.b'}^{\mathcal{A}}| \leq$ negl(λ).*

Proof. Suppose there exists a PPT Adversary $\mathcal{A}$, an index $j \in [n]$, bit $b' \in \{0,1\}$ such that $|p^{\mathcal{A}}_{3.j.b'} - p^{\mathcal{A}}_{4.j.b'}|$ is non-negligible. We construct a reduction algorithm $\mathcal{B}$ that breaks OWFE security of encryption.

The reduction algorithm first samples $x \leftarrow \{0,1\}^n$ s.t. $x_j = 1-b'$, and sends (x,j) to challenger $\mathcal{C}$. The challenger samples OWFE public parameters $\mathsf{pp}' \leftarrow K(1^\lambda)$, a bit $\alpha \leftarrow \{0,1\}$, randomness ρ and computes $\mathsf{ct}^* = E_1(\mathsf{pp}', j, 1-x_j; \rho)$. If $\alpha = 0$, it computes $y^* = E_2(\mathsf{pp}', f(\mathsf{pp}', x), j, 1-x_j; \rho)$. Otherwise, it samples $y^* \leftarrow \{0,1\}^\ell$. The challenger sends $(\mathsf{pp}', \mathsf{ct}^*, y^*)$ to $\mathcal{B}$. $\mathcal{B}$ then samples randomness $\rho_{i,b} \leftarrow \mathcal{R}$ and computes $\mathsf{ct}_{i,b} = E_1(\mathsf{pp}', (i,b); \rho_{i,b})$ for $(i,b) \neq (j,b')$. It then initializes $\mathsf{ct}_{j,b'} = \mathsf{ct}^*$, $y_{j,b'} = y^*$. $\mathcal{B}$ then computes $t = f(\mathsf{pp}', x)$, $y_0 = \mathsf{PRG}(\mathsf{Ext}(t, \mathfrak{s}))$, and $y_{i,b} = E_2(\mathsf{pp}', (t,i,b); \rho_{i,b})$ for all (i,b) s.t. $(i,b) \succ (j,b')$. It then samples $y_{i,b} \leftarrow \{0,1\}^\ell$ for all $(i,b) \prec (j,b')$ and sends public parameters $\mathsf{pp} = (\mathsf{pp}', \{ct_{i,b}\}_{i,b}, \mathfrak{s})$ and challenge $(y_0, \{y_{i,b}\}_{i,b})$ to the adversary $\mathcal{A}$. The adversary outputs a bit β'. The reduction algorithm $\mathcal{B}$ outputs β' as its guess in OWFE game.

Note that if $\alpha = 0$, $\mathcal{B}$ emulates Hybrid $H_{3.j.b'}$ challenger to $\mathcal{A}$. If $\alpha = 1$, $\mathcal{B}$ emulates Hybrid $H_{3.j.b'}$ challenger to $\mathcal{A}$. Moreover, $\mathcal{B}$ acts as a valid adversary in OWFE encryption security game. By our assumption, $|p^{\mathcal{A}}_{3.j.b'} - p^{\mathcal{A}}_{4.j.b'}| = |\Pr[\beta' = 1|\alpha = 0] - \Pr[\beta' = 1|\alpha = 1]|$ is non-negligible. Therefore, $\mathcal{B}$ breaks OWFE encryption security. ■

Lemma A.5. *Assuming $(n-1,n)$-smoothness of $\mathcal{OWFE}$ scheme, for any PPT adversary $\mathcal{A}$, index $j \in [n]$, bit $b' \in \{0,1\}$, s.t. $(j,b') \prec (n,1)$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have $|p^{\mathcal{A}}_{4.j.b'} - p^{\mathcal{A}}_{3.j+b'.1-b'}| \leq negl(\lambda)$.*

Proof. This proof is similar to proof of Lemma A.3. ■

Lemma A.6. *Assuming $(n-1,n)$-smoothness of $\mathcal{OWFE}$ scheme, for any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have $|p^{\mathcal{A}}_{4.n.1} - p^{\mathcal{A}}_5| \leq negl(\lambda)$.*

Proof. This proof is similar to proof of Lemma A.3. ■

Lemma A.7. *Assuming $\mathcal{OWFE}$ satisfies one-wayness property and* Ext *is a strong seeded extractor with appropriate parameters, for any adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have $|p^{\mathcal{A}}_5 - p^{\mathcal{A}}_6| \leq negl(\lambda)$.*

Proof. Assuming f is one-way, we know that the min-entropy k of the distribution $\{f(x) : x \leftarrow \{0,1\}^n\}$ is $\Omega(\log\lambda)$ bits. If $\mathsf{Ext} : \mathcal{C} \times \mathcal{S} \rightarrow \mathcal{W}$ is a strong seeded (k,ϵ) extractor for a negligibly small ϵ, then the distributions $(\mathfrak{s}, \mathsf{Ext}(f(\mathsf{pp}', x)))$ and $(\mathfrak{s}, u)$, where $\mathfrak{s} \leftarrow \mathcal{S}, \mathsf{pp}' \leftarrow K(1^\lambda), x \leftarrow \{0,1\}^n, w \leftarrow \mathcal{W}$, have a statistical difference of ϵ. Moroever, with an appropriate choice of ϵ, we have $|\mathcal{W}| \geq 2^{\Omega(\log\lambda)}$. ■

Lemma A.8. *Assuming* $\mathsf{PRG} : \mathcal{W} \rightarrow \{0,1\}^\ell$ *is a secure PRG for any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have $|p^{\mathcal{A}}_6 - p^{\mathcal{A}}_7| \leq negl(\lambda)$.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ such that $|p^{\mathcal{A}}_6 - p^{\mathcal{A}}_7|$ is non-negligible. We construct a reduction algorithm $\mathcal{B}$ that breaks PRG security.

The PRG challenger $\mathcal{C}$ samples a bit $\alpha \leftarrow \{0,1\}$. If $\alpha = 0$, it samples $t \leftarrow \mathcal{W}$ and lets $v = \mathsf{PRG}(t)$. Otherwise, it samples $v \leftarrow \{0,1\}^\ell$. The challenger sends v to the reduction algorithm $\mathcal{B}$, which samples HPRG public parameters pp, $y_{i,b} \leftarrow \{0,1\}^\ell$ for $i \in [n], b \in \{0,1\}$, sets $y_0 = v$ and sends pp, challenge $(y_0, \{y_{i,b}\}_{i,b})$ to $\mathcal{A}$. The adversary outputs a bit β'. $\mathcal{B}$ outputs β' as its guess in PRG game.

Note that, $\mathcal{B}$ acts as Hybrid H_6 challenger if $\alpha = 0$, and as Hybrid H_7 challenger if $\alpha = 1$. By our assumption, $|p^{\mathcal{A}}_6 - p^{\mathcal{A}}_7| = |\Pr[\beta' = 1|\alpha = 0] - \Pr[\beta' = 1|\alpha = 1]|$ is non-negligible and $\mathcal{B}$ breaks PRG security. ■

By the above sequence of lemmas and triangle inequality, for any PPT adversary $\mathcal{A}$, there exists a negligible function negl$(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|p^{\mathcal{A}}_0 - p^{\mathcal{A}}_7| \leq \text{negl}(\lambda)$. Therefore, the above construction is a secure hinting prg. ■

B One Way Function with Encryption from q-DDHI Assumption

We now construct (k, n)-OWFE from any n-DDHI hard group generator GGen. Suppose $\mathsf{GGen}(1^\lambda)$ generates a group of order at most 2^m, the below construction requires $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$ for any fixed constant α. This is a variant of the construction from n-DBDHI assumption presented in Section 7. This construction does not use pairings but has longer ciphertext compared to the one presented in Section 7. For the sake of simplicity, we construct a OWFE scheme where the encryption algorithm outputs elements in a group. The construction can be extended to output ℓ-length bit strings by using PRGs and randomness extractors.

$K(1^\lambda)$: Sample a group $\mathcal{G} = (\mathbb{G}, p) \leftarrow \mathsf{GGen}(1^\lambda)$. Sample a generator $g \leftarrow \mathbb{G}_1$ and random values $\alpha, d_0, d_1 \leftarrow \mathbb{Z}_p$. Output the public parameters $(\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$.

$f(\mathsf{pp}, x)$: Parse public parameters pp as $\mathsf{pp} = (\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$. Let the polynomial $(d_0 x + d_1) \cdot \prod_{j=1}^n (\alpha + 2j + x_j) = \sum_{i=0}^n c_i \alpha^i$, where c_i is a function of d_0, d_1, x. Output $\prod_{i=0}^n \left(g^{\alpha^i}\right)^{c_i}$.

$E_1(\mathsf{pp}, (i, b); \rho)$: Let $h = g^{\rho \cdot (\alpha + 2i + b)}$. Compute and output $(h, h^\alpha, h^{\alpha^2}, \cdots, h^{\alpha^{n-1}}, i)$. Note that these values can be computed given $(g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n})$.

$E_2(\mathsf{pp}, (y, i, b); \rho)$: Compute and output y^ρ.

$D(\mathsf{pp}, \mathsf{ct}, x)$: Let $\mathsf{ct} = (h, h^\alpha, h^{\alpha^2}, \cdots, h^{\alpha^{n-1}}, i)$. Consider the polynomial $(d_0 x + d_1) \cdot \prod_{j \in [1,n] \setminus \{i\}} (\alpha + 2j + x_j) = \sum_{j=0}^{n-1} c_j \alpha^j$, where c_j is a function of d_0, d_1, x. Compute and output $\prod_{j=0}^{n-1} \left(h^{\alpha^j}\right)^{c_j}$.

Correctness. For any set of public parameters $\mathsf{pp} = (\mathcal{G}, g, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^n}, d_0, d_1)$, string $x \in \{0,1\}^n$, index $j \in [n]$ and randomness ρ, we have $D(\mathsf{pp}, E_1(\mathsf{pp}, (j, x_j); \rho), x) = g^{\rho (d_0 x + d_1) \prod_i (\alpha + 2i + x_i)} = f(\mathsf{pp}, x)^\rho = E_2(\mathsf{pp}, (f(\mathsf{pp}, x), j, x_j); \rho)$.

B.1 Security

We now prove that the above construction satisfies one-wayness, encryption security and smoothness properties.

One-Wayness. We now prove that the above construction satisfies (k, n)-one-wayness property for any $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$.

Lemma B.1. *Assuming n-DDHI assumption holds (Assumption 2), for any (k, n) source s.t. $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$, the above construction satisfies (k, n)-one-wayness property as per Definition 2.1.*

Proof. This is similar to the proof of Lemma 7.1. Suppose there exists a PPT adversary $\mathcal{A}$ and a (k, n) source S such that $k = \omega(m)$ and $\mathcal{A}$ breaks one-wayness property of the above construction with non-negligible probability. We construct a reduction algorithm $\mathcal{B}$ that wins against n-DDHI challenger $\mathcal{C}$.

The challenger $\mathcal{C}$ first samples a group structure $\mathcal{G} = (\mathbb{G}, p) \leftarrow \mathsf{GGen}(1^\lambda)$, a generator $h \leftarrow \mathbb{G}$, a value $\beta \leftarrow \mathbb{Z}_p^*$ and a bit $\gamma \leftarrow \{0,1\}$. If $\gamma = 0$, it sets $T = h^{1/\beta}$. Otherwise, it samples $T \leftarrow \mathbb{G}$. The challenger then sends $(\mathcal{G}, h, h^\beta, h^{\beta^2}, \cdots, h^{\beta^n}, T)$ to the reduction algorithm $\mathcal{B}$. The reduction algorithm samples a string $x \leftarrow S$ and an index $j \in [n]$. It then implicity sets $\alpha = \beta - 2j - 1 + x_j$. If there exists $i \in [2n+1]$ s.t. $\alpha + i = 0 \bmod p$, then $\mathcal{B}$ aborts and guesses randomly. This happens with negligible probability. The reduction algorithm samples $d_0, d_1 \leftarrow \mathbb{Z}_p$, computes public parameters $\mathsf{pp} = (\mathcal{G}, h, h^\alpha, h^{\alpha^2}, \cdots h^{\alpha^n}, d_0, d_1)$ and sends $\mathsf{pp}, y = f(\mathsf{pp}, x)$ to the adversary $\mathcal{A}$. Note that these values can be computed given $(h, h^\beta, h^{\beta^2}, \cdots, h^{\beta^n})$. The adversary outputs x'. If $f(\mathsf{pp}, x) \neq f(\mathsf{pp}, x')$ or $x_j = x'_j$, then $\mathcal{B}$ aborts and guesses randomly. Otherwise, we know that $h^{(d_0 x + d_1) \cdot \prod_i (\alpha + 2i + x_i)} = h^{(d_0 x + d_1) \prod_i (\alpha + 2i + x'_i)}$. Consider the polynomial

$$\frac{(d_0 x + d_1) \cdot \prod_i (\alpha + 2i + x_i)}{\alpha + 2j + 1 - x_j} = \frac{c}{\beta} + \sum_{i=0}^{n-1} c_i \beta^i$$

where $c, \{c_i\}_i$ are dependent only on x, d_0, d_1. We know that, $h^{\frac{c}{\beta}} \cdot \prod_{i=0}^{n-1} \left(g^{\beta^i}\right)^{c_i} = h^{(d_0 x + d_1) \cdot \prod_{i \neq j}(\alpha + 2i + x'_i)}$. $\mathcal{B}$ checks if $T^c \cdot \prod_{i=0}^{n-1} \left(g^{\beta^i}\right)^{c_i} = h^{(d_0 x + d_1) \cdot \prod_{i \neq j}(\alpha + 2i + x'_i)}$ holds. If the condition holds, $\mathcal{B}$ outputs 0. Otherwise, it outputs 1.

We now analyze the advantage of $\mathcal{B}$ in n-DDHI game. As β is sampled uniformly, α is also uniformly distributed. As p is superpolynomial in λ, $\alpha \in [1, p-2n-2]$ with overwhelming probability and $\mathcal{B}$ simulates one-wayness game challenger to $\mathcal{A}$ in statistically indistinguishable way. By our assumption, $f(\mathsf{pp}, x) \neq f(\mathsf{pp}, x')$ with non-negligible probability. By Lemma C.1, for any (k, n) source S s.t. $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$, the distribution of $f(\mathsf{pp}, S)$ is statistically close to uniform as α is sampled from $[1, p-2n-2]$. As a result, with overwhelming probability there exists many z s.t. $f(\mathsf{pp}, x) = f(\mathsf{pp}, z)$. As j is sampled uniformly from $[n]$, $f(\mathsf{pp}, x) = f(\mathsf{pp}, x'), x_j \neq x'_j$ with non-negligible probability. The final check performed by $\mathcal{B}$ always holds if $T = e(h, h)^{1/\beta}$, and holds with negligible probability if T is sampled randomly. Therefore, $\mathcal{B}$ wins n-DBDHI game with non-negligible probability. ■

Security of Encryption. We now prove that the above construction satisfies encryption security property.

Lemma B.2. *Assuming n-DDHI assumption holds (Assumption 2), the above construction satisfies encryption security property as per Definition 2.2.*

Proof. This is similar to the proof of Lemma 7.2. Suppose there exists a PPT adversary $\mathcal{A}$ that breaks encryption security of the above construction with non-negligible probability. We construct a reduction algorithm $\mathcal{B}$ that wins against n-DDHI challenger $\mathcal{C}$.

The challenger $\mathcal{C}$ first samples a group structure $\mathcal{G} = (\mathbb{G}, p) \leftarrow \mathsf{GGen}(1^\lambda)$, a generator $h \leftarrow \mathbb{G}$, a value $\beta \leftarrow \mathbb{Z}_p^*$ and a bit $\gamma \leftarrow \{0, 1\}$. If $\gamma = 0$, it sets $T = h^{1/\beta}$. Otherwise, it samples $T \leftarrow \mathbb{G}$. The challenger then sends $(\mathcal{G}, h, h^\beta, h^{\beta^2}, \cdots, h^{\beta^n}, T)$ to the reduction algorithm $\mathcal{B}$. The adversary sends a string $x \in \{0, 1\}^n$ and an index j to $\mathcal{B}$. $\mathcal{B}$ samples $d_0, d_1 \leftarrow \mathbb{Z}_p$ and implicitly sets $\alpha = \beta - 2j - 1 + x_j$. It then computes public parameters $\mathsf{pp} = (\mathcal{G}, h, h^\alpha, h^{\alpha^2}, \cdots, h^{\alpha^n}, d_0, d_1)$, samples randomness $\rho \leftarrow \mathbb{Z}_p$ and computes $\mathsf{ct}^* = (h^\rho, h^{\rho \cdot \alpha}, h^{\rho \cdot \alpha^2}, \cdots, h^{\rho \cdot \alpha^{n-1}}, j)$. Consider the polynomial

$$\frac{\rho \cdot (d_0 x + d_1) \cdot \prod_{i=1}^{n}(\alpha + 2i + x_i)}{\alpha + 2j + 1 - x_j} = \frac{c}{\beta} + \sum_{i=0}^{n-1} c_i \beta^i$$

where $c, \{c_i\}_i$ are dependent only on ρ, x, d_0, d_1. The reduction algorithm computes $k^* = T^c \cdot \prod_{i=0}^{n-1} \left(h^{\beta^i}\right)^{c_i}$ and sends $\mathsf{pp}, \mathsf{ct}^*, k^*$ to the adversary. The adversary outputs a bit γ'. $\mathcal{B}$ outputs γ' as its guess in n-DDHI game.

We now analyze the advantage of $\mathcal{B}$ in n-DDHI game. As β is sampled uniformly, α is also uniformly distributed. Let $\rho' = \frac{\rho}{\alpha + 2j + 1 - x_j} \bmod p$. As $\beta \neq 0 \bmod p$ and ρ is uniformly distributed, ρ' is also uniformly distributed in $\mathbb{Z}_p$. If $\gamma = 0$, then $(\mathsf{pp}, \mathsf{ct}^*, k^*)$ is same as $(\mathsf{pp}, E_1(\mathsf{pp}, (j, 1 - x_j); \rho'), E_2(\mathsf{pp}, (f(\mathsf{pp}, x), j, 1 - x_j); \rho')$. If $\gamma = 1$, then k^* is uniformly random. As $\mathcal{A}$ distinguishes these 2 distributions with non-negligible probability, $|\Pr[\gamma' = 1 | \gamma = 0] - \Pr[\gamma' = 1 | \gamma = 1]|$ is non-negligible. Therefore, $\mathcal{B}$ breaks n-DDHI assumption. ■

Smoothness. We now prove that the above construction satisfies (k, n)-smoothness property for any $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$.

Lemma B.3. *The above construction satisfies (k, n)-smoothness property for any $k \geq m + 2\lambda$ and $n \leq k + m - 2\lambda$ as per Definition 2.3.*

Proof. This proof is same as the proof of Lemma 7.3. ■

C Leftover Hash Lemma over $\mathbb{Z}_p$

We now present a lemma that is used in proving various theorems in this work.

Lemma C.1. *For any prime p, any (k,n)-source S s.t. $k \geq \log p + 2\lambda$, $n \leq k + \log p - 2\lambda$, any subset $\mathcal{A} \subseteq [0, p-2n-2]$, the distribution $\Big(K, H(K,S)\Big)$ is statistically indistinguishable from $\Big(K, U\Big)$, where $H : \mathcal{K} \times \{0,1\}^n \to \mathbb{Z}_p$ is a hash function with key space $\mathcal{K} = (\mathbb{Z}_p \times \mathbb{Z}_p \times \mathcal{A})$ and is defined as $H((d_0,d_1,\alpha),x) = (d_0 x + d_1) \cdot \prod_{k=1}^{n} (\alpha + 2k + x_k)$ mod p, hash key $K = (d_0, d_1, \alpha)$ is uniformly sampled from $\mathcal{K}$ and U is the uniform distribution on $\mathbb{Z}_p$.*

Proof. This proof is information-theoretic. We first bound the probability $\Pr_K[H(K,s) = H(K,t)]$ for any $s,t \in \{0,1\}^n$ s.t. $s \neq t$. We then use analysis similar to leftover hash lemma and prove that the distribution $(K, H(K,S))$ is statistically indistinguishable from (K,U). Consider any $s,t \in \{0,1\}^n$ s.t. $s \neq t$.

$$\Pr_K[H(K,s) = H(K,t)] = \sum_{c \in \mathbb{Z}_p} \Pr_K[H(K,s) = H(K,t) = c]$$

$$= \sum_{c \in \mathbb{Z}_p} \Pr_{d_0,d_1,\alpha} \Big[(d_0 s + d_1) = c \cdot \prod_{k=1}^{n} (\alpha + 2k + s_k)^{-1} \bmod p \bigwedge (d_0 t + d_1) = c \cdot \prod_{k=1}^{n} (\alpha + 2k + t_k)^{-1} \bmod p\Big]$$

Note that for any $\alpha \in \mathcal{A}$, $(\alpha + 2k + s_k)^{-1}, (\alpha + 2k + t_k)^{-1}$ mod p is unique for all $k \in [n]$. For any $\alpha \in \mathcal{A}$ and $c \in \mathbb{Z}_p$, let us compute the number of (d_0, d_1) pairs in $\mathbb{Z}_p^2$ that satisfy the above pair of equations. On subtracting the equations, we get

$$d_0(s-t) = c \cdot \Big(\prod_{k=1}^{n} (\alpha + 2k + s_k)^{-1} - \prod_{k=1}^{n} (\alpha + 2k + t_k)^{-1}\Big) \bmod p.$$

Consider the following 2 cases.

- Case 1 ($s - t \neq 0$ mod p): There exists a unique (d_0, d_1) pair satisfying the pair of equations for every $\alpha \in \mathcal{A}$ and $c \in \mathbb{Z}_p$. Therefore,

$$\sum_{c \in \mathbb{Z}_p} \Pr_K[H(K,s) = H(K,t) = c] = \sum_{c \in \mathbb{Z}_p} \frac{1}{p^2} = \frac{1}{p}$$

- Case 2 ($s - t = 0$ mod p): If $c = 0$, for any $\alpha \in \mathcal{A}$, number of $(d_0, d_1) \in \mathbb{Z}_p^2$ satisfying the pair of equations is p. If $c \neq 0$, for any $\alpha \in \mathcal{A}$ such that $f(\alpha) = \prod_{k=1}^{n} (\alpha + 2k + s_k) - \prod_{k=1}^{n} (\alpha + 2k + t_k) = 0$ mod p, number of $(d_0, d_1) \in \mathbb{Z}_p^2$ satisfying the pair of equations is p. For any α s.t. $f(\alpha) \neq 0$ mod p, no $(d_0, d_1) \in \mathbb{Z}_p^2$ satisfying the pair of equations. By lagrange's theorem, $f(\alpha) = 0$ mod p has at most n solutions.

$$\begin{aligned}
\sum_{c \in \mathbb{Z}_p} \Pr_K[H(K,s) = H(K,t) = c] &\leq \Pr_K[H(K,s) = H(K,t) = 0] + \sum_{c \neq 0} \Pr_K[H(K,s) = H(K,t) = c] \\
&\leq \frac{p}{p^2} + \sum_{c \neq 0} \frac{p}{p^2} \cdot \Pr_{\alpha} \Big[\prod_{k=1}^{n} (\alpha + 2k + s_k) - \prod_{k=1}^{n} (\alpha + 2k + t_k) = 0 \bmod p\Big] \\
&\leq \frac{1}{p} + (p-1) \cdot \frac{1}{p} \cdot \frac{n}{p - 2n - 1} \leq \frac{2n+1}{p} \qquad \text{(Assuming } p - 2n - 1 \geq p/2\text{)}
\end{aligned}$$

We now bound the statistical distance between distributions $\mathcal{D}_1 = (K, H(K,S))$ and $\mathcal{D}_2 = (K,U)$. For any distribution D, let $\mathsf{CP}(D)$ be collision probability on D.

$$\begin{aligned}
\mathsf{CP}(D_1) &= \Pr_{\substack{K_1,K_2,\\ s,t\leftarrow S}}[(K_1, H(K_1,s)) = (K_2, H(K_2,t))] = \Pr[K_1 = K_2] \cdot \Pr_{\substack{K,\\ s,t\leftarrow S}}[H(K,s) = H(K,t)] \\
&= \frac{1}{|\mathcal{K}|} \cdot \Big(\Pr_{s,t}[s=t] + \Pr_{s,t}[s \neq t \bmod p] \Pr_{H}[H(s) = H(t) | s \neq t \bmod p] \\
&\qquad + \Pr_{s,t}[s = t \bmod p, s \neq t] \Pr_{H}[H(s) = H(t) | s = t \bmod p, s \neq t] \Big) \\
&\le \frac{1}{|\mathcal{K}|} \cdot \Big(\frac{1}{2^k} + 1 \cdot \frac{1}{p} + \frac{1}{2^k} \cdot \left\lfloor \frac{2^n}{p} \right\rfloor \cdot \frac{2n+1}{p} \Big) \le \frac{1}{|\mathcal{K}|} \cdot \Big(\frac{1}{2^k} + \frac{1}{p} + \frac{2^{n-k} \cdot (2n+1)}{p^2} \Big)
\end{aligned}$$

We know that statistical difference between D_1 and D_2 is given by

$$\begin{aligned}
\mathsf{SD}(D_1, D_2) &\le \sqrt{|\mathcal{K}| \cdot p} \sqrt{\mathsf{CP}(D_1) - \mathsf{CP}(D_2)} \\
&\le \sqrt{|\mathcal{K}| \cdot p} \sqrt{\frac{1}{|\mathcal{K}|} \Big(\frac{1}{2^k} + \frac{1}{p} + \frac{2^{n-k} \cdot (2n+1)}{p^2} \Big) - \frac{1}{|\mathcal{K}| \cdot p}} \\
&= \sqrt{\frac{p}{2^k} + \frac{2^{n-k} \cdot (2n+1)}{p}} = \mathrm{negl}(\lambda) \qquad (\text{As } k \ge \log p + 2\lambda \text{ and } n - k \le \log p - 2\lambda)
\end{aligned}$$

■