Algebraic Frameworks for Cryptographic Primitives

by

Navid Alamati

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2020

Doctoral Committee:

Associate Professor Chris Peikert, Chair
Professor Jeffrey Lagarias
Professor Quentin Stout
Professor Martin Strauss

"It must be admitted that the use of geometric intuition has no logical necessity in mathematics, and is often left out of the formal presentation of results. If one had to construct a mathematical brain, one would probably use resources more efficiently than creating a visual system. But the system is there already, it is used to great advantage by human mathematicians, and it gives a special flavor to human mathematics."

David Ruelle
[Conversations on Mathematics with a Visitor from Outer Space, 1998]

Navid Alamati

alamati@umich.edu
ORCID iD: 0000-0001-8621-7486
2020

Acknowledgments

Undoubtedly, I was more than fortunate to be advised by Chris Peikert. He has guided me during the past five years with unwavering patience and unparalleled percipience. No words can adequately describe my appreciation for him. I thank the other members of the committee—Jeffrey Lagarias, Quentin Stout, and Martin Strauss.

I was lucky to be interned twice at Fujitsu Labs of America, during which I have had the chance of working with wonderful researchers—Arnab Roy, Hart Montgomery, and Sikhar Patranabis. I am grateful to Hart and Sikhar for the innumerable hours of discussion and the countless moments of wonder through which I tried to do research in cryptography. I am also thankful to Luca De Feo for being a great companion for an attempt on isogenies for dummies, one of whom has written this thesis.

Last but not least, I would like to thank the (virtual) accompany of many people around the world, all of whom can be brought under the umbrella of *friends*.

Table of Contents

List of Figures

List of Abbreviations

wPRF Weak Pseudorandom Function

wUF Weak Unpredictable Function

CRHF Collision-Resistant Hash Function

EGA Effective Group Action

HOWF Homomorphic One-Way Function

HPS Hash Proof System

IHwPRF Input-Homomorphic weak Pseudorandom Function

IHwUF Input-Homomorphic weak Unpredictable Function

KHwPRF Key-Homomorphic weak Pseudorandom Function

MPC Multiparty Computation

OT Oblivious Transfer

PIR Private Information Retrieval

PKE Public-Key Encryption

PRF Pseudorandom Function

REGA Restricted Effective Group Action

SSP-OT Statistically Sender-Private Oblivious Transfer

TDF Trapdoor Function

UF Unpredictable Function

Abstract

A fundamental goal in theoretical cryptography is to identify the conceptually simplest abstractions that generically imply a collection of other cryptographic primitives. For symmetric-key primitives, this goal has been accomplished by showing that one-way functions are necessary and sufficient to realize primitives ranging from symmetric-key encryption to digital signatures. By contrast, for asymmetric primitives, we have no (known) unifying simple abstraction even for a few of its most basic objects. Moreover, even for public-key encryption (PKE) alone, we have no unifying abstraction that all known constructions follow. The fact that almost all known PKE constructions exploit some algebraic structure suggests considering abstractions that have some basic algebraic properties, irrespective of their concrete instantiation.

We make progress on the aforementioned fundamental goal by identifying simple and useful cryptographic abstractions and showing that they imply a variety of asymmetric primitives. Our general approach is to augment symmetric abstractions with algebraic structure that turns out to be sufficient for PKE and much more, thus yielding a "bridge" between symmetric and asymmetric primitives. We introduce two algebraic frameworks that capture almost all concrete instantiations of (asymmetric) cryptographic primitives, and we also demonstrate their applicability by showing their cryptographic implications. Therefore, rather than manually building different cryptosystems from a new assumption, one only needs to build one (or more) of our simple structured primitives, and a whole host of cryptosystems immediately follows.

CHAPTER 1

Introduction

A symmetric-key encryption scheme requires two or more parties to share a secret key through some secret channel prior to encryption and decryption. By contrast, in a public-key encryption (PKE) scheme, the public keys (which are needed for encryption) can be distributed publicly, including to the attacker. Although the history of symmetric-key cryptography goes back millennia, public-key cryptography is a relatively recent development, dating to the mid-20th century. In a celebrated work, Impagliazzo [Imp95] proposed five "worlds" and their implications for algorithms and cryptography, which range from *Algorithmica*—where "efficient" algorithms for all (worst-case) problems in NP exist and cryptography is essentially nonexistent—to *Cryptomania*, a world in which public-key cryptography exists. Only two of Impagliazzo's worlds allow for cryptography: *Minicrypt*, where symmetric cryptographic primitives exist but public-key cryptography does not, and the aforementioned *Cryptomania*.

It turns out that Minicrypt is a fairly simple world, and all Minicrypt primitives can be based on a simple abstraction called a *one-way function* (OWF). An OWF is an efficiently computable function that it is hard to invert on the image of a random input. A number of famous works have shown how to build the most commonly studied and used Minicrypt primitives from one-way functions in a generic manner. For instance, one-way functions imply pseudorandom generators [BM82, HILL99], which in turn can be used to build pseudorandom functions [GGM84]. From these primitives, it has long been known how to generically build symmetric-key encryption schemes and digital signature schemes [Rom90]. As Barak [Bar17] points out, "it seems that you cannot throw a rock without hitting a one-way function," and we can base candidate OWFs on a variety of problems from different domains such as algebra, logic, and combinatorics.

By contrast, Cryptomania has no unifying abstraction even for a few of its most basic objects. Moreover, even for PKE alone, we have no unifying abstraction that all known constructions follow. To get PKE we have to rely on a limited class of two "strains" of problem types [Bar17], namely

"algebraic" and "noisy linear" problems, which all relate to algebraic structures. Even within each strain there is no unifying way of viewing the PKE constructions. For instance, in the ElGamal PKE scheme [ElG84] the decryption algorithm performs multiplication/exponentiation, whereas in the Goldwasser-Micali PKE scheme [GM82] the decryption algorithm tests for quadratic residuosity.

A fundamental goal in theoretical cryptography is to identify the conceptually simplest objects that generically imply a collection of others. Ideally, we want the situation of Cryptomania to be more like Minicrypt, with many objects implied by (if not equivalent to) a common simple abstraction or a small set thereof. This raises a fundamental question in the complexity of public-key cryptography: do there exist such objects for Cryptomania? The fact that almost all known PKE constructions exploit some common algebraic structure suggests considering abstractions that have some basic algebraic properties, irrespective of their concrete instantiation. Hence, the question essentially is what should the particular primitives look like (in a huge design space), such that:

- they are easily instantiated from different concrete assumptions (such as the Diffie-Hellman assumptions in appropriately chosen groups, or the Quadratic Residuosity assumption);
- they generically imply (almost) all of the objects that have previously been obtained from these concrete assumptions;
- each of these primitives is conceptually and syntactically simple—in particular, simpler than Cryptomania objects that they imply.

1.1 A Bridge from Minicrypt to Cryptomania[1]

We make progress on the above-described goal by identifying three simple cryptographic objects and showing their applications. Our approach is to augment Minicrypt abstractions with simple algebraic structure that turns out to be sufficient for PKE and much more, thus yielding a "bridge" between Minicrypt and Cryptomania. We focus on the well-studied Minicrypt primitives one-way function (OWF), weak unpredictable function (wUF), and weak pseudorandom function (wPRF).

A keyed function family is a function $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ such that $\mathcal{K}$ is the key space and $\mathcal{X}, \mathcal{Y}$ are input and output spaces, respectively. Typically, the sets $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{K}$ are parameterized by the security parameter λ, which is a parameter used to quantify resource requirements of a cryptographic protocol. We often use the notation $F_k(x)$ to denote $F(k, x)$. If $G : \mathcal{X} \to \mathcal{Y}$ is a function, let $G^{\$}$ denote a randomized oracle that, when invoked, samples $x \leftarrow \mathcal{X}$ uniformly and outputs $(x, G(x))$.

[1]The results of this section are taken from the paper [AMPR19].

A weak unpredictable function (wUF) family is an efficiently computable (keyed) function family F such that for all probabilistic polynomial-time adversaries $\mathcal{A}$ we have

$$\Pr[\mathcal{A}^{F_k^{\$}}(x^*) = F_k(x^*)] \leq \mathrm{negl}(\lambda),$$

where $\mathrm{negl}(\lambda)$ denotes some negligible function of λ and $k \leftarrow \mathcal{K}$, $x^* \leftarrow \mathcal{X}$ are uniformly sampled elements from $\mathcal{K}, \mathcal{X}$, respectively. Basically, the security requirement is that given access to polynomially many (random) input-output pairs of the form $(x_i, F_k(x_i))$, no attacker can predict $F_k(x^*)$ with non-negligible probability for a fresh uniformly chosen $x^* \leftarrow \mathcal{X}$.

A weak pseudorandom function (wPRF) family is an efficiently computable (keyed) function family F such that for all probabilistic polynomial-time adversaries $\mathcal{A}$ we have

$$\left|\Pr[\mathcal{A}^{F_k^{\$}} = 1] - \Pr[\mathcal{A}^{U^{\$}} = 1]\right| \leq \mathrm{negl}(\lambda),$$

where $k \leftarrow \mathcal{K}$, and $U : \mathcal{X} \to \mathcal{Y}$ is a truly random function. Roughly speaking, the security requirement is that given access to polynomially many (random) input-output pairs of the form (x_i, y_i), no attacker can distinguish between the real experiment where $y_i = F_k(x_i)$ and the ideal experiment where $y_i = U(x_i)$ for a truly random function U.

To define our structured primitives, we augment OWF/wUF/wPRF with a group homomorphism. An input-homomorphic weak UF/PRF (IHwUF/IHwPRF) F is a weak UF/PRF with a group homomorphism over the input and output space. That is, $(\mathcal{X}, \oplus)$ and $(\mathcal{Y}, \otimes)$ are groups with efficiently computable group operations such that for every $k \in \mathcal{K}$ the function $F_k : \mathcal{X} \to \mathcal{Y}$ is a group homomorphism, i.e., we have

$$F\left(k, x_1 \oplus x_2\right) = F\left(k, x_1\right) \otimes F\left(k, x_2\right).$$

A Homomorphic One-Way Function (HOWF) is a one-way function $f : \mathcal{X} \to \mathcal{Y}$ such that $\mathcal{X}$ and $\mathcal{Y}$ are groups with efficiently computable group operations and f is a group homomorphism.

We also consider bounded homomorphisms, where we have a bounded number of invocations for which the homomorphism property holds. This restriction lets us work with lattice-based and other "noisy" cryptographic assumptions.

We present a framework for building cryptographic primitives from Minicrypt primitives with algebraic structure, and we show that many cryptographic primitives can be realized (in a generic manner) from HOWF/IHwUF/IHwPRF. We refer the reader to Figure 1.1 for an overview of our framework. To name a few, we show that

- HOWF implies collision-resistant hash function (CRHF), which is a compressing function such that it is hard (for all polynomially bounded attackers) to find two distinct inputs that hash to the same value. Consequently, HOWF gets us outside Minicrypt (because CRHF is not known to follow from ordinary OWF [Sim98]), but not quite to Cryptomania.

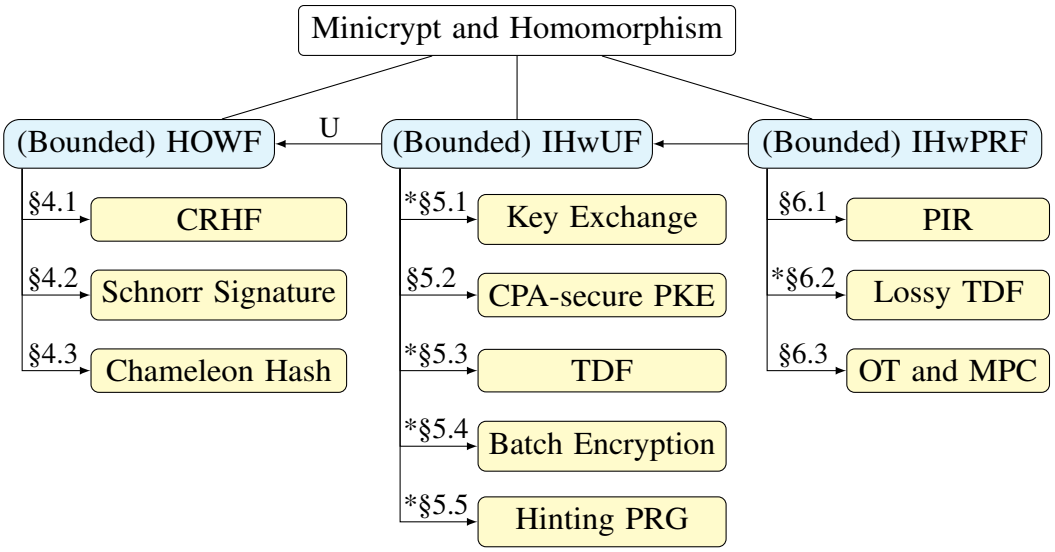
- IHwUF implies PKE, along with a variety of rich Cryptomania objects that are considered quite a bit more powerful that ordinary PKE. For instance, IHwUF implies trapdoor function (TDF), which is an OWF with some extra trapdoor (which can be generated together with the description of the function) that makes inversion easy. IHwUF also implies identity-based encryption, which is a richer form of encryption in which any string (such as an email address) can be used as a public key.

- IHwPRF implies private information retrieval (PIR), which is a useful primitive that allows a user to retrieve an item from a database without revealing which item is retrieved. A nontrivial PIR requires that the total communication complexity of the protocol should be sublinear in the size of database. IHwPRF also implies the powerful notion of lossy trapdoor function, which is known to be sufficient for realizing many cryptographic primitives, ranging from chosen ciphertext-secure cryptosystem to oblivious transfer [PW08].

Instantiations from concrete assumptions. We show that "mainstream" concrete cryptographic assumptions such as Decisional Diffie-Hellman (DDH) and Learning With Errors (LWE) naturally imply (bounded) HOWF/IHwUF/IHwPRF. We also show that a (bounded) group-homomorphic PKE implies a (bounded) IHwPRF. This allows instantiating these primitives from any concrete assumption that implies a (bounded) homomorphic PKE (e.g., Quadratic Residuosity assumption). Unfortunately, there is a caveat to this: the transformation from homomorphic PKE to IHwPRF comes with a disadvantage that the input space may *depend* on the key.[1] The reader may refer to Figure 1.2 for an overview of instantiations from concrete assumptions.

Building cryptosystems from new assumptions. One of the benefits of our framework is the implications for new assumptions. Rather than manually building different cryptosystems from a new assumption, one only needs to build one (or more) of our simple structured primitives, and a whole host of cryptosystems immediately follows. Ideally, it will be easy to show the existence of many cryptographic primitives from new assumptions using our results.

[1]This property is necessary to realize certain cryptographic primitives from IHwUF or IHwPRF. We refer to Section 3.4 for a discussion on this property and the details of the instantiations.



U: Unbounded homomorphism with abelian groups.
*: The input space is *independent* of the key.

Figure 1.1: Cryptographic primitives from HOWF/IHwUF/IHwPRF.

1.2 Homomorphism Over the Key Space[1]

So far, we only considered realizing strong asymmetric primitives from *input-homomorphic* weak PRF. However, there is also a primitive that is homomorphic with respect to the key (rather that the input), known as key-homomorphic weak PRF (KHwPRF). Roughly speaking, a key-homomorphic (weak) PRF family is a (weak) PRF family with the following homomorphism property over the key and output space:

$$F\left(k_1 \oplus k_2, x\right) = F\left(k_1, x\right) \otimes F\left(k_2, x\right).$$

Key-homomorphic (weak) pseudorandom functions were first implicitly shown in [NPR99] in the random oracle model and then formally defined and constructed in the standard model with slightly relaxed approximate homomorphism property in [BLMR13]. Key-homomorphic (weak) PRF has been used to build primitives like distributed PRF [NPR99, BLMR13], updatable encryption [EPRS17, LT18], and PRF with security against related-key attacks [LMR14].

Following the work of Boneh *et al.* [BLMR13], there have been a few works constructing improved variants of approximate KHPRF [BP14, BV15, BFP+15]. However, all of the known

[1]The results of this section are taken from the paper [AMP19].

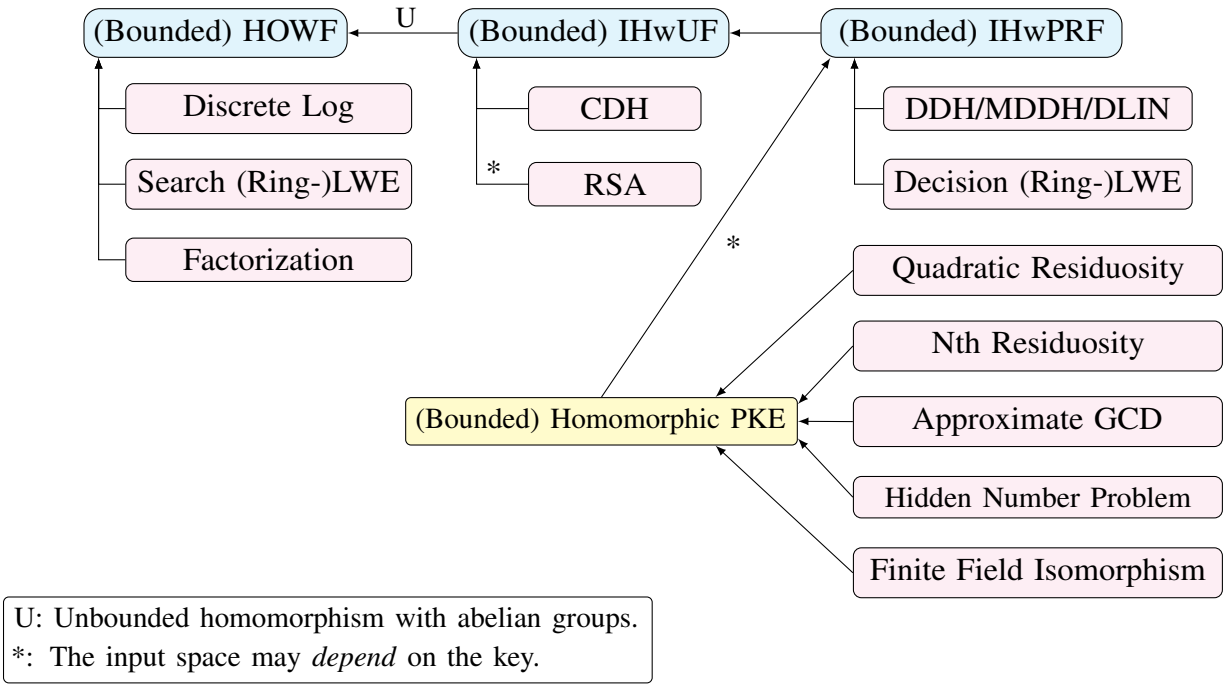


Figure 1.2: Instantiations from concrete assumptions

constructions of KHPRF still require strong "PKE assumptions." For instance, we only know how to build (exact) key-homomorphic PRF in the standard model from multilinear maps or related assumptions [BLMR13]. Even constructions in the random oracle model require public-key assumptions like DDH [NPR99].

The assumptions required for the aforementioned constructions of KHPRF are seemingly heavyweight for an ostensibly symmetric-key primitive that is typically targeted for applications in the symmetric-key setting. This leads us to a natural question: can we construct more efficient key-homomorphic PRFs, or is there some fundamental lower bound limiting their efficiency? Boneh *et al.* [BLMR13] optimistically state, "Another interesting area of research is to construct key-homomorphic PRFs whose performance is comparable to real-world block ciphers such as AES," but so far there are no known realizations of such a KHPRF.

We prove that any key-homomorphic *weak* PRF (KHwPRF) $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with an abelian output group $\mathcal{Y}$ implies noninteractive key exchange. In fact, we show that a KHwPRF with an abelian output group implies a much stronger primitive, namely *input-homomorphic* wPRF. By the results of previous section [AMPR19], IHwPRF implies a large number of public-key

primitives, including identity-based encryption (IBE) [DG17b, BLSV18], private information retrieval (PIR) [KO97], and lossy trapdoor function (LTDF) [PW08]. In essence, our results indicate that it is seemingly unlikely that KHwPRF, and hence KHPRF, with abelian output groups are implied by symmetric-key primitives [IR89, GHMM18]. Our results also hold in the bounded setting with abelian output groups (encompassing nearly all applications of *almost* KHPRF from lattice-based assumptions). To our knowledge, all existing constructions of KHwPRF (and almost KHwPRF) have abelian output groups.

In addition, we show that any exact (not bounded) homomorphic one-way function (HOWF) with abelian input and output groups can be broken in polynomial time using a quantum computer. This immediately rules out the existence of abelian, exact KHwPRF (and hence KHPRF) in the quantum setting. In other words, over abelian groups, KHwPRF (and KHPRF) with bounded homomorphism are the best that we can hope for in the quantum world. We refer to Section 4.4 for more details.

Finally, we demonstrate a construction of Naor-Reingold style PRF [NR97] from any KHwPRF. As we explain in Section 7.5, this allows us to construct parallel and potentially efficient PRF from any KHwPRF. To the best of our knowledge, it was not known how to construct a Naor-Reingold style PRF from a *generic* primitive.

1.3 A Framework Based on Group Actions[1]

In the previous sections, we discussed the power of basic symmetric primitives endowed with some algebraic structure, namely group homomorphism. However, there are certain cryptographic constructions that are not captured in the homomorphism-based framework, most notably the isogeny-based cryptography. The study of isogeny-based cryptography was initiated in 1997 by Couveignes [Cou06], but began in earnest in the late 2000s with several new ideas around collision resistant hashing [CLG09], key exchange [RS06, Sto10], signatures [Sto09] and key escrow [Tes06]. Isogeny-based cryptography became much more popular after the introduction of the SIDH key exchange scheme [JD11, DJP14], the first practical post-quantum scheme based on isogenies, and a precursor to the NIST competition candidate SIKE [AKC$^+$17].

One of the most recent additions to the isogeny portfolio is CSIDH [CLM$^+$18], an efficient variant of the original key-exchange proposal of Couveignes, Rostovtsev, and Stolbunov. CSIDH spurred a fair amount of new research in isogeny-based schemes, notably signatures [DG19, BKV19]. Indeed, among all isogeny-based assumptions, CSIDH, its predecessors, and its derivatives

[1]The results of this section are taken from the paper [ADMP20].

are the only ones amenable to group actions.

Isogenies. In a nutshell, an isogeny is a morphism of elliptic curves, i.e., a map from a curve to another curve that preserves the group structure. The central objects of study in isogeny-based cryptography are *isogeny graphs*, i.e., graphs whose vertices represent elliptic curves, and whose edges represent isogenies between them. There is a large variety of isogeny graphs, depending on which kinds of curves and isogenies are chosen. One such choice would be *complex multiplication graphs*, which arise from so-called *horizontal* isogenies of complex mutliplication elliptic curves; indeed, these graphs are isomorphic to Cayley graphs of *quadratic imaginary class groups*, and thus present a natural group action.

Constructions from isogenies. Known constructions from isogeny-based assumptions amenable to group actions include noninteractive key exchange [CLM+18], interactive zero-knowledge protocols and signatures [DG19, BKV19], multi-round UC-secure oblivious transfer against passive corruptions [dOPS18], and threshold signatures [DM20].

In addition, known constructions from isogeny-based assumptions not amenable to group actions include PKE [JD11, AKC+17], ephemeral key exchange [JD11], interactive zero-knowledge protocols and signatures [DJP14, YAJ+17, GPS17], collision-resistant hash function [CLG09], multi-round UC-secure oblivious transfer against passive corruptions [BOB18, dOPS18, Vit19], and verifiable delay functions [DMPS19].

Cryptographic group actions. In order to simplify the presentation and understanding of certain isogeny-based constructions, some prior works have chosen to use *group actions* as an abstraction for them, including the first presentations [Cou06]. Informally, a group action is a mapping of the form $\star : G \times X \to X$, where G is a group and X is a set, such that for any $g_1, g_2 \in G$ and any $x \in X$, we have

$$g_1 \star (g_2 \star x) = (g_1 g_2) \star x.$$

From a cryptographic point of view, we can endow group actions with different hardness properties. For instance, a *one-way* group action [BY91] is endowed with the following property: given randomly chosen set elements $x_1, x_2 \in X$, it is hard to find a group element $g \in G$ such that $g \star x_1 = x_2$ (assuming such a g exists). Similarly, one could define a *weakly pseudorandom* group action with following property: for a randomly chosen secret group element $g \in G$, an adversary that sees many tuples of the form $(x_i, g \star x_i)$ cannot distinguish them from tuples of the form (x_i, u_i)

where each x_i and u_i are sampled uniformly and independently from X.[1] We refer to group actions endowed with such hardness properties as *cryptographic group actions*.

As an example, we note that a simple cryptographic group action is implied by the DDH assumption. If we set $X = \mathbb{H}$ (where $\mathbb{H}$ is some group of prime order p) and $G = \mathbb{Z}_p^*$, then the mapping $z \star h \mapsto h^z$ where

$$\star : \mathbb{Z}_p^* \times \mathbb{H} \to \mathbb{H}$$

is a weakly pseudorandom group action assuming that the DDH assumption holds over $\mathbb{H}$. We note that here the "set" $\mathbb{H}$ is actually structured. However, there exist candidate quantum-resistant cryptographic group actions where the set may not be a group.

Cryptographic group actions have received less attention compared to traditional group-based assumptions. Nonetheless, there has been a small number of works studying various candidate cryptographic group actions [GS10, JQSY19] and their hardness properties [BY91, GPSV18]. In terms of public-key primitives, these works have demonstrated that cryptographic group actions endowed with some hardness properties imply PKE and non-interactive key exchange (NIKE).

In terms of cryptographic capabilities, group-theoretic assumptions have been studied extensively over the past couple of decades. At present, we have a reasonably comprehensive understanding of what is (and is not) constructible from the most commonly encountered group-theoretic assumptions such as DLOG, CDH, and DDH[2] (barring a few breakthrough results using novel non-black-box techniques, e.g., [DG17b]). The cryptographic capabilities of these assumptions have also been explained from the point of view of their underlying algebraic structure [AMPR19]. On the other hand, our understanding of the cryptographic capabilities of group actions is still somewhat limited.

A natural question to ask is what primitives can we build from cryptographic group actions? We believe that it is important to understand the cryptographic capabilities of group actions given that they capture the algebraic structure underlying some candidate post-quantum cryptographic assumptions, namely isogeny-based cryptography amenable to group actions.

Cryptographic group actions and isogenies. One of the key objects associated with an elliptic curve is its *endomorphism ring*. In the cases that interest us here, this ring is known to be isomorphic to an imaginary quadratic order $\mathcal{O}$, i.e., a 2-dimensional $\mathbb{Z}$-lattice and a subring of an imaginary quadratic number field $\mathbb{Q}(\sqrt{D})$. An elliptic curve with endomorphism ring isomorphic to a given $\mathcal{O}$ is said to have *complex multiplication (CM) by* $\mathcal{O}$.

[1] We note that sampling directly from the uniform distribution over the set X may not be possible in certain cases. We elaborate more on this later.

[2] Discrete Logarithm, Computational Diffie-Hellman, and Decisional Diffie-Hellman assumptions.

The celebrated theory of complex multiplication establishes a correspondence between the *ideal classes* of $\mathcal{O}$ and the isogenies between elliptic curves with CM by $\mathcal{O}$. More precisely, it defines a regular abelian group action

$$\mathrm{Cl}(\mathcal{O}) \times \mathcal{E}_k(\mathcal{O}) \to \mathcal{E}_k(\mathcal{O})$$

of the *class group* $\mathrm{Cl}(\mathcal{O})$ on the set $\mathcal{E}_k(\mathcal{O})$ of elliptic curves, defined over a field k, with complex multiplication by $\mathcal{O}$. Moreover, each element of $\mathrm{Cl}(\mathcal{O})$ corresponds to a unique class of isogenies, which can be leveraged to evaluate the group action. We refer to [De 17] for more details.

Unfortunately, the correspondence between isogenies and the CM group action becomes less than ideal when we start contemplating algorithmic properties. Indeed, a natural requirement for a cryptographic group action is that given *any* group element $g \in G$ and a set element $x \in X$, computing $g \star x$ can be done efficiently. However this does not hold for the CM group action, which can be evaluated efficiently only for a small subset of group elements.

The usual workaround adopted in isogeny-based cryptography is to represent elements of $\mathrm{Cl}(\mathcal{O})$ as $\mathbb{Z}$-linear combinations of a fixed set of "low norm" generators $\mathfrak{g}_i$ for which evaluating the group action is efficient, i.e., as $\mathfrak{a} = \prod_{i=1}^{\ell} \mathfrak{g}_i^{a_i}$. Then, evaluating the action is efficient as long as the exponents a_i are polynomial in the security parameter. This trick is not devoid of consequences: group elements do not have a unique representation, sampling uniformly in the group may not be possible in general, and even testing equality becomes tricky. We will capture the limitations of this framework in our definition of a *Restricted Effective Group Action (REGA)*.

To illustrate the limitations of REGA, we refer to SeaSign [DG19], which is the Fiat-Shamir transform of an interactive authentication protocol based on CSIDH. To prove the knowledge of a secret $s \in G$ s.t. $y = s \star x$, the basic idea is to first commit to $r \star x$ for some random r, and then reveal $s^{-b}r$ depending on a bit b sent by the challenger. While it is straightforward to prove that this protocol is zero-knowledge when the elements of G have unique representation and are sampled uniformly, the proof breaks down for CSIDH if we do not have a unique representation of $\mathrm{Cl}(\mathcal{O})$, which happens for some choices of parameters. To fix this issue, SeaSign uses a rejection sampling technique [Lyu09], which considerably increases parameters and signature/verification time.

An alternative fix is to compute the group structure of $\mathrm{Cl}(\mathcal{O})$, in the form of a *relation lattice* of the low norm generators. This restores the ability to represent uniquely and to sample uniformly the elements of the group. This is the approach taken by the isogeny-based signature CSI-FiSh [BKV19], which pre-computes the group structure.

While the approach taken by CSI-FiSh to build a full-fledged cryptographic group action extends the capabilities of isogeny-based cryptography, recent results [Pei20, BS20] showed quantum attacks against CSIDH for certain choice of parameters. Unfortunately, computing the group structure of a

larger class group seems out of reach today, due to the subexponential complexity of the classical algorithms. This limitation will go away once quantum computers become powerful enough to apply Shor's algorithms to this group order computation, but until then we believe that REGA can be a fundamental tool to construct post-quantum cryptographic protocols based on isogenies.

Bilinear maps gained popularity in cryptography partly because works such as [BF01, GPS08] presented them in a generic, easy-to-use manner that effectively abstracted the mathematical details underlying Weil or Tate pairings. Similarly, an easy-to-use abstraction for isogeny-based assumptions might make them more accessible to cryptographers.

We improve the state of the art of cryptographic group actions as follows:

- We formally define different notions of cryptographic group actions endowed with natural hardness properties such as *one-wayness*, *weak unpredictability*, and *weak pseudorandomness*. We then show how certain isogeny-based assumptions can be modeled using our definitions.
- We show several applications of cryptographic group actions (based on our definitions above) which were not previously known from isogeny-based assumptions. These include smooth projective hash functions, dual-mode PKE, and two-message statistically sender-private OT.
- We introduce a new assumption over cryptographic group actions called the *linear hidden shift* (LHS) assumption. We present some discussions on the security of the LHS assumption and show that it implies symmetric KDM-CPA secure encryption, which in conjunction with PKE implies many primitives that were not previously known from isogeny-based assumptions.

(Restricted) Effective Group Action. We introduce new definitions for group actions endowed with hardness properties. As we have alluded earlier, not all isogeny-based assumptions can be modeled by a cryptographic group action. So, building upon the definitions of Brassard and Yung [BY91] and Couveignes [Cou06] as starting points, we create new definitions that allow us to model a larger class of isogeny-based assumptions. Our first new definition is that of an *effective* group action (EGA). This models the standard notion of cryptographic group actions. In Section 8.1 we present the formal definitions for effective group actions and the associated axioms of mathematical structure. While our definitions bear some resemblance to existing works, they are more amenable to cryptographic constructions in the post-quantum setting. Much of the early work on cryptographic group actions [BY91, Cou06] either predates the major advances in quantum cryptanalysis like Shor's algorithm [Sho97] or did not focus on post-quantum applications. We note that CSI-FiSh [BKV19] can be modeled as an effective group action defined above (plausibly as a weakly pseudorandom effective group action).

Our definition of effective group action does not capture isogeny-based assumptions such as CSIDH [CLM+18], where we cannot compute the group action efficiently for all $g \in G$. To address this, we introduce the notion of a *restricted* effective group action (REGA). The basic idea is the following: in an REGA the group action is efficiently computable for some small subset of G. Note that we can still "simulate" the effect of a general group action by computing the group action on a sequence of different elements from this subset.

New constructions. We show new constructions from our group-action based primitives, which can be concretely instantiated from some isogeny-based assumptions. We refer to Figure 1.3 for an overview of our results.

- **Statistically Sender-Private OT (SSP-OT):** We show how to build two-message statistically sender-private OT [NP01] in the *plain model* from any weakly pseudorandom EGA/REGA. SSP-OT has many cryptographic applications, such as non-malleable commitments [KS17], two-round witness indistinguishable proofs with private-coin verifier [JKKR17, BGI+17, KKS18], and three-message statistical receiver-private OT in the plain model [GJJM20]. To our knowledge, these primitives were not previously known from isogeny-based assumptions.

- **Dual-mode PKE:** We demonstrate a construction of dual-mode PKE [PVW08] from any weakly pseudorandom EGA/REGA. Dual-mode PKE implies UC-secure[1] round-optimal OT protocols in the common reference string model against malicious parties. Such OT protocols are in turn sufficient to construct UC-secure round-optimal multi-party computation (MPC) protocols for general functionalities in the same security model [GS18] . Our construction implies the first round-optimal OT and MPC protocols from isogeny-based assumptions. Previously known constructions of OT from isogenies [BOB18, dOPS18, Vit19] were neither round-optimal nor UC secure against malicious parties.

- **Hash Proof System (HPS):** We demonstrate a construction of HPS [CS02] from any weakly pseudorandom EGA/REGA. Hash proof system has many applications such as chosen-ciphertext secure (CCA-secure) PKE [CS02], password authenticated key-exchange (PAKE) [GL03], and privacy-preserving protocols [BPV12]. To our knowledge, this is the first hash proof system from isogeny-based assumptions. In particular, our result implies the first CCA-secure encryption scheme in the standard model from isogenies. Previously known CCA-secure encryption schemes from isogenies [CLM+18] were in the random oracle model.

[1] Universal Composability (UC) is a frameowork for the analysis of cryptographic protocols that guarantees security under a general composition operation [Can01].

Linear Hidden Shift assumption. We introduce a new assumption over group actions that we call the *Linear Hidden Shift* (LHS) assumption and provide some evidence for its security. We describe the assumption informally below.

Let G be a multiplicative group. For a vector of group elements $\mathbf{g} \in G^n$ and a binary vector $\mathbf{s} \in \{0,1\}^n$, let $\langle \mathbf{g}, \mathbf{s} \rangle$ denote the subset product $\prod_{i=1}^{n} g_i^{s_i}$. Informally, the LHS assumption states that for any m that is polynomial in the security parameter, it holds that

$$\{(x_i, \mathbf{g}_i, (\langle \mathbf{g}_i, \mathbf{s} \rangle) \star x_i)\}_{i \in [m]} \quad \stackrel{c}{\approx} \quad \{(x_i, \mathbf{g}_i, u_i)\}_{i \in [m]},$$

where $\mathbf{g}_i \leftarrow G^n$, $\mathbf{s} \leftarrow \{0,1\}^n$, $x_i \leftarrow X$, and $u_i \leftarrow X$ (all sampled independently).

The LHS assumption is sufficient to realize symmetric-key KDM-CPA secure encryption, and enables us to realize many primitives such as TDF and designated-verifier NIZK, which were previously not known from isogeny-based assumptions. We believe that the LHS assumption is of independent interest and may have other cryptographic applications.

On the LHS assumption. We show some evidence to support the claim that LHS assumption holds over any weakly pseudorandom EGA/REGA. We first show a *search to decision reduction*: namely, that the decision variant of the LHS assumption mentioned above is *equivalent* to its search variant, which states that no PPT adversary can recover $\mathbf{s}$. Next, we show that in certain settings an additive variant of the LHS assumption is equivalent to the weak pseudorandom EGA if $G = \mathbb{Z}_N^*$ and the elements are sampled from a structured distribution. Based on this evidence, it appears likely that the LHS assumption holds with respect to some of the known group-action based isogenies.

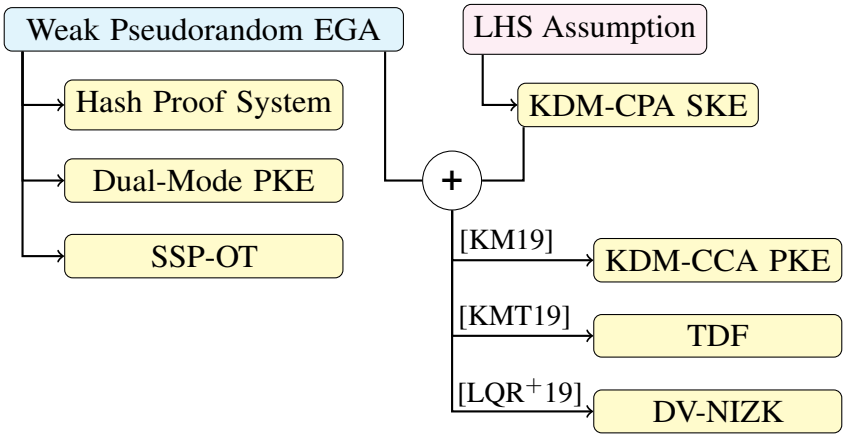


Figure 1.3: Overview of our results and implications

1.4 Organization

The preliminaries and detailed constructions (and proofs) are organized as follows:

- Chapter 2 presents preliminary background material.
- Chapter 3 formally defines (bounded) HOWF/IHwUF/IHwPRF/KHwPRF, describes a general protocol to build primitives from IHwUF/IHwPRF, and shows instantiations from concrete assumptions.
- Chapter 4 describes how to construct some primitives from (bounded) HOWF, and shows the impossibility of HOWF with exact homomorphism in the quantum setting.
- Chapter 5/6 shows constructions of various primitives from (bounded) IHwUF/IHwPRF, and describes how to instantiate them from the general protocol.
- Chapter 7 shows constructions of various primitives (including Naor-Reingold style PRF) from (bounded) KHwPRF. It also presents a discussion on the output group of KHwPRF.
- Chapter 8 introduces our group action-based framework and describes constructions of different primitives from a weakly pseudorandom EGA/REGA. It also presents the LHS assumption and describes a construction of symmetric-key KDM-CPA encryption scheme.

CHAPTER 2

Preliminaries

2.1 Notation

For any positive integer n, we use $[n]$ to denote the set $\{1, \ldots, n\}$. We use λ for the security parameter. We use the symbols $\oplus$, $\otimes$, and $\odot$ as group operations defined in the context. For two equal-length strings s_1 and s_2 we denote their bitwise XOR as $\mathrm{XOR}(s_1, s_2)$. For a finite set S, we use $s \leftarrow S$ to sample uniformly from the set S. We use the notations $\overset{s}{\approx}$ and $\overset{c}{\approx}$ to denote statistical and computational indistinguishability, respectively.

Let $\mathbf{Y} = \{y_{ij}\} \in \mathcal{Y}^{m \times n}$ be an $m \times n$ matrix of group elements. Let $\mathbf{s} = (s_1, \ldots, s_n) \in \{0,1\}^n$ be an arbitrary binary vector. We denote by $\mathbf{Ys} \in \mathcal{Y}^m$ the vector of group elements

$$\Big(\bigotimes_{j:s_j=1} y_{1,j}, \ldots, \bigotimes_{j:s_j=1} y_{m,j} \Big).$$

We naturally extend this notation to matrices of group elements. Let $\mathbf{S} = [\mathbf{s}_1 \mid \ldots \mid \mathbf{s}_\ell] \in \{0,1\}^{n \times \ell}$ be an arbitrary binary matrix whose columns are $\mathbf{s}_1, \ldots, \mathbf{s}_\ell$. We denote by $\mathbf{YS} \in \mathcal{Y}^{m \times \ell}$ the matrix of group elements $[\mathbf{Ys}_1 \mid \ldots \mid \mathbf{Ys}_\ell]$.

2.2 Cryptographic Primitives

If $\{\mathcal{X}_\lambda^{(i)}\}_{i \in [\ell]}$ be an ensemble of distributions (where each distribution is parameterized by λ) such that $\ell \in \mathrm{poly}(\lambda)$ and $\mathcal{X}_\lambda^{(i)} \overset{c}{\approx} \mathcal{X}_\lambda^{(i+1)}$ for $i \in [n-1]$, then $\mathcal{X}_\lambda^{(1)} \overset{c}{\approx} \mathcal{X}_\lambda^{(n)}$.

A keyed function family is a function $F : K \times X \to Y$ such that K is the key space and X, Y are input and output spaces, respectively. We often use the notation $F_k(x)$ to denote $F(k, x)$. If $G : X \to Y$ is a function, let $G^{\$}$ denote a randomized oracle that, when invoked, samples $x \leftarrow X$ uniformly and outputs $(x, G(x))$.

Weak Unpredictable Function. A weak unpredictable function (wUF) family is an efficiently computable (keyed) function family F (where K, X, Y are indexed by the security parameter λ) such that for all probabilistic polynomial-time adversaries $\mathcal{A}$ we have

$$\Pr[\mathcal{A}^{F_k^{\$}}(x^*) = F_k(x^*)] \leq \mathrm{negl}(\lambda),$$

where $k \leftarrow K$, $x^* \leftarrow X$ are uniformly sampled elements from K, X, respectively. Basically, the security requirement is that given access to polynomially many (random) input-output pairs of the form $(x_i, F_k(x_i))$, no attacker can predict $F_k(x^*)$ with non-negligible probability for a fresh uniformly chosen $x^* \leftarrow X$.

Unpredictable Function. An unpredictable function (UF) family is an efficiently computable (keyed) function family F such that for all probabilistic polynomial-time adversaries $\mathcal{A}$ we have

$$\Pr[\mathcal{A}^{F(k,\cdot)} = (x^*, F(k, x^*))] \leq \mathrm{negl}(\lambda),$$

where $k \leftarrow K$ and x^* is any element that has *not* been queried by $\mathcal{A}$. We remark that in case of UF, the inputs are *adaptively* chosen, and the adversary does not need to prepare its queries beforehand.

Weak Pseudorandom Function. A weak pseudorandom function (wPRF) family is an efficiently computable (keyed) function family F such that for all probabilistic polynomial-time adversaries $\mathcal{A}$ we have

$$\left|\Pr[\mathcal{A}^{F_k^{\$}} = 1] - \Pr[\mathcal{A}^{U^{\$}} = 1]\right| \leq \mathrm{negl}(\lambda),$$

where $k \leftarrow K$, and $U : X \rightarrow Y$ is a truly random function. Roughly speaking, the security requirement is that given access to polynomially many (random) input-output pairs of the form (x_i, y_i), no attacker can distinguish between the real experiment where $y_i = F_k(x_i)$ and the ideal experiment where $y_i = U(x_i)$ for a truly random function U.

Pseudorandom Function. A pseudorandom function (PRF) family is an efficiently computable (keyed) function family F such that for all probabilistic polynomial-time adversaries $\mathcal{A}$ we have

$$\left|\Pr[\mathcal{A}^{F_k} = 1] - \Pr[\mathcal{A}^{U} = 1]\right| \leq \mathrm{negl}(\lambda),$$

where $k \leftarrow K$, and $U : X \rightarrow Y$ is a truly random function.

Collision-Resistant Hash Function. Let $\{H : \{0,1\}^{\ell(\lambda)} \to \{0,1\}^{m(\lambda)}\}_{H \in \mathcal{H}}$ be family of (keyed) hash functions. We say $\mathcal{H}$ is a collision-resistant hash function (CRHF) family if $m(\lambda) < \ell(\lambda)$ and for any PPT adversary $\mathcal{A}$

$$\Pr_{H \leftarrow \mathcal{H}}[(x, x') \leftarrow \mathcal{A}(H) \text{ s.t. } x \neq x' \wedge H(x) = H(x')] \leq \mathrm{negl}(\lambda).$$

Trapdoor Function. A tuple of algorithms (G, F, F^{-1}) is said to be a trapdoor function (TDF) family if it satisfies the following properties:

- $G(1^\lambda)$ outputs $(\mathsf{pp}, \mathsf{t})$ where $F_{\mathsf{pp}}(\cdot)$ computes an injective function whose input space is $\{0,1\}^n$, and $F_{\mathsf{t}}^{-1}(\cdot)$ computes the inverse of $F_{\mathsf{pp}}(\cdot)$.
- For any adversary $\mathcal{A}$ we have

$$\Pr_{\substack{\mathsf{pp} \leftarrow G(1^\lambda) \\ x \leftarrow \{0,1\}^n}}[x \leftarrow \mathcal{A}(\mathsf{pp}, F_{\mathsf{pp}}(x))] \leq \mathrm{negl}(\lambda).$$

We refer the reader to [Gol01, Gol04, KL14] for a comprehensive treatment of well-known cryptographic primitives.

2.3 Min-entropy and Leftover Hash Lemma

For a discrete random variable Z with sample space Ω, its *min-entropy* is defined as

$$H_\infty(Z) = \min_{\omega \in \Omega}\{-\log \Pr[Z = \omega]\}.$$

We will use the following lemma, which is a simplified version of the Leftover Hash Lemma (LHL). We refer the reader to [HILL99] for a proof of LHL.

Lemma 2.3.1. *Let $\{H_s : \mathcal{Z} \to Y\}_{s \in \mathcal{S}}$ be a family of pairwise independent hash functions, and let Z and S be discrete random variables over $\mathcal{Z}$ and $\mathcal{S}$, respectively. If $H_\infty(X) > \log|Y| + 2\log(\varepsilon^{-1})$ we have*

$$\Delta[(S, H_S(X)), (S, U)] \leq \varepsilon,$$

where U denotes the uniform distribution over the set Y.

We will also use the following corollary of the leftover hash lemma.

Lemma 2.3.2. *Let G be a (multiplicative) finite group, and let n be a positive integer such that $n > \log|G| + \omega(\log(\lambda))$. If $\mathbf{g} \leftarrow G^n$ and $\mathbf{s} \leftarrow \{0,1\}^n$, we have*

$$(\mathbf{g}, \prod_{i=1}^{n} g_i^{s_i}) \overset{s}{\approx} (\mathbf{g}, u),$$

where $u \leftarrow G$ is a uniformly chosen element from G.

CHAPTER 3

A Framework Based on Homomorphism

3.1 Building Blocks

Definition 3.1.1. (Homomorphic One-Way Function.) A one-way function $f : \mathcal{X} \to \mathcal{Y}$ is called a homomorphic one-way function (HOWF) if the following conditions hold:

1. $(\mathcal{X}, \oplus)$ and $(\mathcal{Y}, \otimes)$ are both efficiently samplable groups.
2. The group operations $\oplus$ and $\otimes$, and inverse operation in each group are efficiently computable.
3. For every $x_1, x_2 \in \mathcal{X}$, we have

$$f(x_1 \oplus x_2) = f(x_1) \otimes f(x_2).$$

We also consider a notion of *bounded* homomorphism, in the sense that the homomorphism is preserved for an a priori bounded number of group operations in the input group of one-way function. We formally describe this notion as γ-bounded homomorphism, where the parameter γ reflects the maximum number of group operations that the homomorphism can tolerate.

Definition 3.1.2. (γ-Bounded Homomorphic OWF.) A one-way function $f : \mathcal{X} \to \mathcal{Y}$ is called a γ-bounded homomorphic one-way function (γ-bounded HOWF) if the following conditions hold:

1. $(\mathcal{X}, \oplus)$ and $(\mathcal{Y}, \otimes)$ are both efficiently samplable groups.
2. The group operations $\oplus$ and $\otimes$, and inverse operation in each group are efficiently computable.
3. If $(x_1, \ldots, x_L) \leftarrow \mathcal{X}^L$ and $L \leq \gamma$, it holds that

$$\Pr\left[f\Big(\bigoplus_{j\in[L]} x_j\Big) = \bigotimes_{j\in[L]} f(x_j)\right] \geq 1 - \mathrm{negl}(\lambda).$$

Definition 3.1.3. (Input-Homomorphic Weak UF.) A function family $\{F(k,\cdot) : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ is called an Input-Homomorphic Weak UF (IHwUF) family if the following conditions hold:

1. $\{F(k,\cdot) : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ is a *weak UF* family.
2. $(\mathcal{X}, \oplus)$ and $(\mathcal{Y}, \otimes)$ are both efficiently samplable groups.
3. The group operations $\oplus$ and $\otimes$, and inverse operation in each group are efficiently computable.
4. For every $k \in \mathcal{K}$ and for every $x_1, x_2 \in \mathcal{X}$, we have

$$F\left(k, x_1 \oplus x_2\right) = F\left(k, x_1\right) \otimes F\left(k, x_2\right).$$

Definition 3.1.4. (Input-Homomorphic Weak PRF.) A function family $\{F(k,\cdot) : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ is called an Input-Homomorphic Weak PRF (IHwPRF) family if it satisfies the aforementioned requirements and additionally, F is a *weak PRF* family.

We also consider a notion of *bounded* homomorphism, in the sense that input homomorphism is preserved only for an a priori bounded number of group operations in the input group of the weak UF/PRF. We formally describe this notion as γ-bounded homomorphism, where the parameter γ reflects the maximum number of group operations that the homomorphism can tolerate.

Definition 3.1.5. (γ-Bounded Input-Homomorphic Weak UF.) A family $\{F(k,\cdot) : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ is called a γ-bounded IHwUF family if there exists a universal mapping $\mathcal{R} : \mathcal{Y} \to \mathcal{Z}$ such that:

1. $\{F(k,\cdot) : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ and $\{\mathcal{R}\left(F(k,\cdot)\right) : \mathcal{X} \to \mathcal{Z}\}_{k\in\mathcal{K}}$ are weak UF families.
2. $(\mathcal{X}, \oplus)$ and $(\mathcal{Y}, \otimes)$ are both efficiently samplable groups.
3. The group operations $\oplus$ and $\otimes$, and inverse operation in each group are efficiently computable.
4. If $(x_1, \ldots, x_L) \leftarrow \mathcal{X}^L$ and $L \leq \gamma$, it holds that

$$\Pr\left[\mathcal{R}\Big(F\big(k, \bigoplus_{j\in[L]} x_j\big)\Big) = \mathcal{R}\Big(\bigotimes_{j\in[L]} F\left(k, x_j\right)\Big)\right] \geq 1 - \mathrm{negl}(\lambda).$$

Definition 3.1.6. (γ-Bounded Input-Homomorphic Weak PRF.) A family $\{F(k,\cdot) : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ is called a γ-bounded IHwPRF family if it satisfies the aforementioned requirements and additionally, F is a *weak PRF* family.

Definition 3.1.7. (Key-Homomorphic Functions.) A function family $\{F(k,\cdot) : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ is key-homomorphic if the following conditions hold:

- $(\mathcal{K}, \oplus)$ and $(\mathcal{Y}, \otimes)$ are efficiently samplable groups, and the group operations and the inverse operation in each group are efficiently computable.
- For any pair of keys $k_1, k_2 \in \mathcal{K}$ and any input $x \in \mathcal{X}$, we have

$$F(k_1, x) \otimes F(k_2, x) = F(k_1 \oplus k_2, x).$$

A key-homomorphic weak PRF (KHwPRF) family is a weak PRF family that is key homomorphic. Similarly, a key-homomorphic PRF (KHPRF) family is a PRF family that is also key homomorphic.

Definition 3.1.8. A weak pseudorandom function family $\{F(k,\cdot) : \mathcal{X} \to \mathcal{Y}\}_{k\in\mathcal{K}}$ is a γ-bounded KHwPRF family if there exists (efficiently computable) universal mappings $\mathcal{R}_{\mathsf{in}} : \mathcal{Y} \to \mathcal{Z}_{\mathsf{in}}$ and $\mathcal{R}_{\mathsf{out}} : \mathcal{Z}_{\mathsf{in}} \to \mathcal{Z}_{\mathsf{out}}$ such that

- $(\mathcal{K}, \oplus)$, $(\mathcal{Y}, \otimes)$, and $(\mathcal{Z}_{\mathsf{in}}, \odot)$ are efficiently samplable groups, and the group operations and the inverse operation in each group are efficiently computable.
- For a randomly chosen key vector $(k_1, \ldots, k_L) \leftarrow \mathcal{K}^L$ such that $L \leq \gamma$, and a randomly chosen input $x \leftarrow \mathcal{X}$, the following holds with overwhelming probability:

$$\mathcal{R}_{\mathsf{in}}\Big(F\Big(\bigoplus_{j\in[L]} k_j, x\Big)\Big) = \mathcal{R}_{\mathsf{in}}\Big(\bigotimes_{j\in[L]} F(k_j, x)\Big),$$

$$\mathcal{R}_{\mathsf{out}}\Big(\bigodot_{j\in[L]} \mathcal{R}_{\mathsf{in}}\Big(F(k_j, x)\Big)\Big) = \mathcal{R}_{\mathsf{out}}\Big(\mathcal{R}_{\mathsf{in}}\Big(\bigotimes_{j\in[L]} F(k_j, x)\Big)\Big).$$

3.2 A Family of Collision-Resistant One-Way Functions

The starting point our framework is a family of CR-OWFs from IHwUFs (and hence, IHwPRFs). Informally, given an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, subset-sum on uniformly random vectors $\mathbf{x} \in \mathcal{X}^n$ is both one-way and collision-resistant when n is sufficiently large. This family has a few interesting features. Firstly, the evaluation procedure involves n homomorphic operations, where n is apriori bounded, meaning that the family is equivalently implied by any γ-bounded IHwUF for $\gamma \geq n$. Secondly, evaluation correctness, one-wayness and collision-resistance hold even if the input and output group of the IHwUF are *nonabelian*. Thirdly, the resulting function family does *not* use the

key space of F explicitly. As we will see later, this feature will be helpful to construct asymmetric cryptographic primitives.

Construction. Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an IHwUF. Fix $n > \log |\mathcal{X}| + \omega(\log(\lambda))$ and sample $2n$ uniformly random group elements from $\mathcal{X}$ as

$$\mathbf{x} = \{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}.$$

Define the family of functions $\mathrm{OWF}_{\mathbf{x}} : \{0,1\}^n \to \mathcal{X}$ as

$$\mathrm{OWF}_{\mathbf{x}}(\mathbf{r} = (r_1, \ldots, r_n)) = \bigoplus_{j\in[n]} x_{j,r_j}.$$

One-Wayness. We first prove the one-wayness of the aforementioned family of functions. Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an IHwUF. Define the experiment $\mathsf{Expt}^{\mathrm{OWF-IHwUF}}$ as follows.

1. The challenger uniformly samples group elements $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ and a binary string $\mathbf{r} = (r_1, \cdots, r_n) \leftarrow \{0,1\}^n$.
2. The challenger computes $x^* = \bigoplus_{j\in[n]} x_{j,r_j}$ and sends the tuple $\left(\{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, x^*\right)$ to the adversary $\mathcal{A}$.
3. Eventually, the adversary $\mathcal{A}$ outputs a binary string $\mathbf{r}' = (r'_1, \ldots, r'_n) \in \{0,1\}^n$.

For any PPT adversary $\mathcal{A}$ we define $\mathbf{Adv}^{\mathrm{OWF-IHwUF}}(\mathcal{A})$ to be the probability of $x^* = \bigoplus_{j\in[n]} x_{j,r'_j}$.

Lemma 3.2.1. *For all PPT adversaries $\mathcal{A}$, we have $\mathbf{Adv}^{\mathrm{OWF-IHwUF}}(\mathcal{A}) = \mathrm{negl}(\lambda)$.*

Proof. Suppose that there exists a PPT adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\mathrm{OWF-IHwUF}}(\mathcal{A})$ is non-negligible. We construct a PPT algorithm $\mathcal{B}$ such that $\mathcal{B}$ breaks the weak unpredictability of F. $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ queries its oracle $2n$ times and receives $\{x_{j,b}, y_{j,b} = F(k, x_{j,b})\}_{j\in[n],b\in\{0,1\}}$. The algorithm $\mathcal{B}$ also gets a (uniformly random) challenge $x^* \in \mathcal{X}$.
2. $\mathcal{B}$ forwards the tuple $\left(\{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, x^*\right)$ to the adversary $\mathcal{A}$.

3. Eventually, $\mathcal{A}$ outputs a binary string $\mathbf{r} = (r_1, \cdots, r_n) \in \{0,1\}^n$.

4. If $x^* = \bigoplus_{j\in[n]} x_{j,r_j}$, $\mathcal{B}$ outputs $y^* = \bigotimes_{j\in[n]} y_{j,r_j}$. Otherwise $\mathcal{B}$ outputs a random $y^* \leftarrow \mathcal{Y}$.

By the leftover hash lemma, we know that $(\mathbf{x}, \bigoplus_{j\in[n]} x_{j,r_j})$ is statistically indistinguishable from $(\mathbf{x}, x^*)$. Hence $\mathcal{B}$ correctly simulates the one-wayness game for $\mathcal{A}$. By input-homomorphism of F, we have

$$F(k, x^*) = F\Big(k, \bigoplus_{j\in[n]} x_{j,r_j}\Big) = \bigotimes_{j\in[n]} y_{j,r_j} = y^*.$$

It follows that $\mathbf{Adv}^{\text{IHwUF}}(\mathcal{B})$ is negligibly different from $\mathbf{Adv}^{\text{OWF}-\text{IHwUF}}(\mathcal{A})$, as desired.

Collision Resistance. We prove that the aforementioned OWF family is also collision-resistant. Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an IHwUF. For a fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, define the experiment $\mathsf{Expt}^{\text{CRHF}-\text{IHwUF}}$ as follows.

1. The challenger samples $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ and sends them to the adversary $\mathcal{A}$.
2. $\mathcal{A}$ outputs $\mathbf{r} = (r_1, \cdots, r_n) \in \{0,1\}^n$ and $\mathbf{r}' = (r'_1, \cdots, r'_n) \in \{0,1\}^n$.

For any PPT adversary $\mathcal{A}$ we define $\mathbf{Adv}^{\text{CRHF}-\text{IHwUF}}(\mathcal{A})$ to be the probability of the event that

$$\bigoplus_{j\in[n]} x_{j,r_j} = \bigoplus_{j\in[n]} x_{j,r'_j}.$$

Lemma 3.2.2. *For all PPT adversaries $\mathcal{A}$, we have $\mathbf{Adv}^{\text{CRHF}-\text{IHwUF}}(\mathcal{A}) = \text{negl}(\lambda)$.*

Proof. Suppose that there exists a PPT adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\text{CRHF}-\text{IHwUF}}(\mathcal{A})$ is non-negligible. We construct a PPT algorithm $\mathcal{B}$ such that $\mathcal{B}$ breaks the weak unpredictability of F as follows:

1. $\mathcal{B}$ queries its oracle $2n$ times and receives $\{x_{j,b}, y_{j,b} = F(k, x_{j,b})\}_{j\in[n],b\in\{0,1\}}$. The algorithm $\mathcal{B}$ also gets a (uniformly random) challenge $x^* \in \mathcal{X}$.
2. $\mathcal{B}$ uniformly randomly picks $i \leftarrow [n]$ and $b^* \leftarrow \{0,1\}$, and sets $x_{i,b^*} := x^*$. It then forwards $\{x_{j,b}\}_{j\in[n],b\in\{0,1\}}$ to the adversary $\mathcal{A}$.
3. $\mathcal{A}$ outputs $\mathbf{r} = (r_1, \cdots, r_n) \in \{0,1\}^n$ and $\mathbf{r}' = (r'_1, \cdots, r'_n) \in \{0,1\}^n$.

4. $\mathcal{B}$ proceeds as follows:

   - If $\bigoplus_{j\in[n]} x_{j,r_j} \neq \bigoplus_{j\in[n]} x_{j,r'_j}$ or $r_i = r'_i$, $\mathcal{B}$ outputs a random $y^* \leftarrow \mathcal{Y}$.
   - Otherwise, assume wlog that $r_i = b^*$. The following must hold:

     $$x^* = \Big(\bigoplus_{j\in[i-1]} x_{j,r_j}\Big)^{-1} \oplus \Big(\bigoplus_{j\in[n]} x_{j,r'_j}\Big) \oplus \Big(\bigoplus_{j\in[i+1,n]} x_{j,r_j}\Big)^{-1},$$

     where the right-hand side is independent of x^*. $\mathcal{B}$ now outputs y^* as

     $$y^* = \Big(\bigotimes_{j\in[i-1]} y_{j,r_j}\Big)^{-1} \otimes \Big(\bigotimes_{j\in[n]} y_{j,r'_j}\Big) \otimes \Big(\bigotimes_{j\in[i+1,n]} y_{j,r_j}\Big)^{-1}.$$

     By the input-homomorphism of F, we have $F(k, x^*) = y^*$.

Observe that if $\mathcal{A}$ outputs a valid collision $(\mathbf{r}, \mathbf{r}')$, the probability that $\mathbf{r}$ and $\mathbf{r}'$ differ in the ith bit for a randomly chosen $i \leftarrow [n]$ is at least $1/n$. It follows that

$$\mathbf{Adv}^{\text{IHwUF}}(\mathcal{B}) \geq \left(\frac{\mathbf{Adv}^{\text{CRHF}-\text{IHwUF}}(\mathcal{A})}{n}\right),$$

which is non-negligible, as desired.

Note 3.2.3. The aforementioned OWF/CRHF family can be instantiated using a γ-bounded IHwUF family, subject to the restriction that $n \leq \gamma$. The proofs of one-wayness and collision resistance also follow similarly.

3.3 The Framework

In this section, we present a framework for designing (asymmetric) cryptographic primitives from a combination of wUF/wPRF evaluations and subset-sum/subset-product operations. The protocol consists of four phases: initialization, pre-evaluation, evaluation and post-evaluation. The evaluation phase is based on wUF/wPRF evaluations, while the pre-evaluation and post-evaluation phases are based on subset-sum/subset-product operations over group elements. For the sake of clarity, we exemplify each phase of the framework using an equivalent framework in the discrete-log setting. More specifically, we show how our framework subsumes a generic methodology of constructing cryptographic primitives from the CDH/DDH assumptions over prime order groups.

1. **Initialization:**

   - Publish $\left(\mathcal{G}, g, N^*, \bar{N}, \widehat{N}\right)$, where $\mathcal{G}$ describes a cyclic group $(\mathbb{G}, \cdot)$ of prime order p with uniform generator g, and $N^* = N^*(\lambda)$, $\bar{N} = \bar{N}(\lambda)$ and $\widehat{N} = \widehat{N}(\lambda)$ are fixed functions.
   - Let $\mathcal{F}$ be the description of an IHwUF/IHwPRF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ over groups $(\mathcal{X}, \oplus)$ and $(\mathcal{Y}, \otimes)$. Fix $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, $N^* = N^*(\lambda)$, $\bar{N} = \bar{N}(\lambda)$ and $\widehat{N} = \widehat{N}(\lambda)$, and publish $\left(\mathcal{F}, \mathrm{X}, n, N^*, \bar{N}, \widehat{N}\right)$, where X is a tuple of $2n$ uniform elements (called base elements) from $\mathcal{X}$ as

$$\mathrm{X} = \{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n], b\in\{0,1\}}.$$

2. **Pre-Evaluation:**

   - Given the generator g, sample N^* random elements from $\mathbb{Z}_q$ as

$$\{\alpha_{n^*} \leftarrow \mathbb{Z}_q\}_{n^*\in[N^*]},$$

     and output the tuple

$$\mathrm{A} = \{g^*_{n^*} = g^{\alpha_{n^*}}\}_{n^*\in[N^*]}.$$

   - Given the tuple of base elements X, sample N^* random strings as

$$\{\mathbf{s}_{n^*} = (s_{n^*,1} \ldots, s_{n^*,n}) \leftarrow \{0,1\}^n\}_{n^*\in[N^*]},$$

     and output the tuple

$$\mathrm{X}^* = \left\{x^*_{n^*} = \bigoplus_{j=1}^{n} x_{j,s_{n^*,j}}\right\}_{n^*\in[N^*]}.$$

3. **Evaluation:**

   - Given the generator g and the pre-evaluation output A, sample $\beta_1, \ldots, \beta_{\bar{N}} \leftarrow \mathbb{Z}_q$ and output B, B^* where

$$\mathrm{B} = \left\{h_{\bar{n}} = g^{\beta_{\bar{n}}}\right\}_{\bar{n}\in[\bar{N}]}, \quad \mathrm{B}^* = \left\{h^*_{\bar{n},n^*} = \left(g^*_{n^*}\right)^{\beta_{\bar{n}}}\right\}_{\bar{n}\in[\bar{N}], n^*\in[N^*]}.$$

- Given base elements X and the pre-evaluation output X^*, sample $k_1, \ldots, k_{\bar{N}} \leftarrow \mathcal{K}$ and output the tuple $\left(\mathrm{Y}, \mathrm{Y}^*\right)$, where

$$\mathrm{Y} = \left\{ y_{j,b}^{(\bar{n})} = F\left(k_{\bar{n}}, x_{j,b}\right) \right\}_{\bar{n}\in[\bar{N}], j\in[n], b\in\{0,1\}}, \mathrm{Y}^* = \left\{ y_{\bar{n},n^*}^* = F\left(k_{\bar{n}}, x_{n^*}^*\right) \right\}_{\bar{n}\in[\bar{N}], n^*\in[N^*]}.$$

4. **Post-Evaluation:**

- Given the generator g and the evaluation outputs $(h_1, \ldots, h_{\bar{N}})$, sample $\widehat{N}$ random elements from $\mathbb{Z}_q$ as

$$\left\{\gamma_{\hat{n}} \leftarrow \mathbb{Z}_q\right\}_{\hat{n}\in[\widehat{N}]},$$

and output

$$\mathrm{C} = \left\{ \hat{g}_{\hat{n}} = g^{\gamma_{\hat{n}}}, \hat{h}_{\bar{n},\hat{n}} = h_{\bar{n}}^{\gamma_{\hat{n}}} \right\}_{\bar{n}\in[\bar{N}], \hat{n}\in[\widehat{N}]}.$$

- Given X and the evaluation output $\mathrm{Y} = \left\{ y_{j,b}^{(1)}, \ldots, y_{j,b}^{(\bar{N})} \right\}_{j\in[n], b\in\{0,1\}}$, sample $\widehat{N}$ random binary vectors as

$$\left\{\mathbf{r}_{\hat{n}} = (r_{\hat{n},1} \ldots, r_{\hat{n},n}) \leftarrow \{0,1\}^n\right\}_{\hat{n}\in[\widehat{N}]},$$

and output $(\hat{\mathrm{X}}, \hat{\mathrm{Y}})$, where

$$\hat{\mathrm{X}} = \left\{ \hat{x}_{\hat{n}} = \bigoplus_{j=1}^{n} x_{j,r_{\hat{n},j}} \right\}_{\hat{n}\in[\widehat{N}]}, \quad \hat{\mathrm{Y}} = \left\{ \hat{y}_{\bar{n},\hat{n}} = \bigotimes_{j=1}^{n} y_{j,r_{\hat{n},j}}^{(\bar{n})} \right\}_{\bar{n}\in[\bar{N}], \hat{n}\in[\widehat{N}]}.$$

Functionality. The following claim is a direct consequence of the input homomorphism of F.

Claim 3.3.1. *For each $n^* \in [N^*], \hat{n} \in [\hat{N}], \bar{n} \in [\bar{N}]$, we have:*

$$\begin{aligned} y_{\bar{n},n^*}^* &= F\left(k_{\bar{n}}, x_{n^*}^*\right) = \bigotimes_{j\in[n]} y_{j,s_{n^*,j}}^{(\bar{n})}, \\ \hat{y}_{\bar{n},\hat{n}} &= \bigotimes_{j\in[n]} y_{j,s_{\hat{n},j}}^{(\bar{n})} = F\left(k_{\bar{n}}, \hat{x}_{\hat{n}}\right). \end{aligned}$$

Remark 3.3.2. If F is a γ-bounded IHwUF/IHwPRF for $\gamma > n$, the relations above are modified as follows:

$$\begin{aligned}\mathcal{R}\left(y^*_{\bar{n},n^*}\right) &= \mathcal{R}\left(F\left(k_{\bar{n}}, x^*_{n^*}\right)\right) = \mathcal{R}\Big(\bigotimes_{j\in[n]} y^{(\bar{n})}_{j,s_{n^*,j}}\Big),\\ \mathcal{R}\left(\hat{y}_{\bar{n},\hat{n}}\right) &= \mathcal{R}\Big(\bigotimes_{j\in[n]} y^{(\bar{n})}_{j,s_{\hat{n},j}}\Big) = \mathcal{R}\left(F\left(k_{\bar{n}}, \hat{x}_{\hat{n}}\right)\right).\end{aligned}$$

Security (IHwPRF). The following theorem captures the security guarantees provided by the general framework when instantiated using an IHwPRF.

Theorem 3.3.3. *Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an* IHwPRF. *Then for any $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, for all functions $N^* = N^*(\lambda)$, $\bar{N} = \bar{N}(\lambda)$ and $\widehat{N} = \widehat{N}(\lambda)$, and for any PPT adversary $\mathcal{A}$, it holds that*

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}, \mathrm{Y}^*, \hat{\mathrm{Y}}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathrm{U}_{\mathrm{X}}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{U}_{\mathrm{Y}}, \mathrm{U}_{\mathrm{Y}^*}, \mathrm{U}_{\hat{\mathrm{Y}}}\right) = 1\right]\right| \leq \mathrm{negl}(\lambda),$$

where

- *The tuple $(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}, \mathrm{Y}^*, \hat{\mathrm{Y}})$ is as defined in the protocol above,*
- *U_{X},$\mathrm{U}_{\mathrm{X}^*}$, and $\mathrm{U}_{\hat{\mathrm{X}}}$ are tuples of $2n$, N^*, and $\widehat{N}$ uniformly sampled group elements from $\mathcal{X}$,*
- *U_{Y}, $\mathrm{U}_{\mathrm{Y}^*}$, and $\mathrm{U}_{\hat{\mathrm{Y}}}$ are tuples of $(2n \cdot \bar{N})$, $(N^* \cdot \bar{N})$, and $(\widehat{N} \cdot \bar{N})$ uniform elements from $\mathcal{Y}$.*

Proof. The proof proceeds in two stages. The first stage applies the leftover hash lemma, while the second stage relies on the weak pseudorandomness of F.

Applying LHL. Let $\mathrm{U}_{\mathrm{X}^*} = \{u^*_1, \ldots, u^*_{N^*}\}$ be a tuple of N^* uniformly sampled group elements in $\mathcal{X}$, and let

$$\widetilde{\mathrm{U}}_{\mathrm{Y}^*} = \left\{y^*_{1,n^*}, \ldots, y^*_{\bar{N},n^*}\right\}_{n^*\in[N^*]},$$

where $y^*_{\bar{n},n^*} = F(k_{\bar{n}}, u^*_{n^*})$ for each $\bar{n} \in [\bar{N}], n^* \in [N^*]$, and $k_1, \ldots, k_{\bar{N}} \in \mathcal{K}$ are the same PRF keys as used in the "real" protocol. We use the following lemma.

Lemma 3.3.4. *For any $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, for all functions $N^* = N^*(\lambda)$, $\bar{N} = \bar{N}(\lambda)$ and $\widehat{N} = \widehat{N}(\lambda)$, and for any PPT adversary $\mathcal{A}$, we have*

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}, \mathrm{Y}^*, \hat{\mathrm{Y}}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \hat{\mathrm{X}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \hat{\mathrm{Y}}\right) = 1\right]\right| \leq \mathrm{negl}(\lambda),$$

Proof. The proof of this lemma follows from the leftover hash lemma. Recall that in the "real" experiment for each $n^* \in [N^*]$ the binary string $\mathrm{s}_{n^*} \in \{0,1\}^n$ used to generate $x^*_{n^*}$ is uniformly and independently sampled. By the leftover hash lemma, it follows that $(\mathrm{X}^*, \mathrm{Y}^*)$ and $\left(\mathrm{U}_{\mathrm{X}^*}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}\right)$ are statistically indistinguishable. This completes the proof of Lemma 3.3.4. □

Similarly, let $\mathrm{U}_{\hat{\mathrm{X}}} = \left\{\hat{u}_1, \ldots, \hat{u}_{\widehat{N}}\right\}$ be a tuple of $\widehat{N}$ uniformly sampled group elements in $\mathcal{X}$, and let

$$\widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}} = \left\{\hat{y}_{1,n^*}, \ldots, \hat{y}_{\bar{N},\hat{n}}\right\}_{\hat{n}\in[\widehat{N}]},$$

where $\hat{y}_{\bar{n},n^*} = F(k_{\bar{n}}, \hat{u}_{n^*})$ for each $\bar{n} \in [\bar{N}]$ and each $\hat{n} \in [\widehat{N}]$, and $k_1, \ldots, k_{\bar{N}} \in \mathcal{K}$ are the same PRF keys as used in the "real" protocol. We also use the following lemma.

Lemma 3.3.5. *For any $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, for all functions $N^* = N^*(\lambda)$, $\bar{N} = \bar{N}(\lambda)$ and $\widehat{N} = \widehat{N}(\lambda)$, and for any PPT adversary $\mathcal{A}$, we have*

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \hat{\mathrm{X}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \hat{\mathrm{Y}}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}\right) = 1\right]\right| \leq \mathrm{negl}(\lambda).$$

Pseudorandomness. We use the weak pseudorandomness of F to prove the following lemma.

Lemma 3.3.6. *For any $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, for all functions $N^* = N^*(\lambda)$, $\bar{N} = \bar{N}(\lambda)$ and $\widehat{N} = \widehat{N}(\lambda)$, and for any PPT adversary $\mathcal{A}$, we have*

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{U}_{\mathrm{Y}}, \mathrm{U}_{\mathrm{Y}^*}, \mathrm{U}_{\hat{\mathrm{Y}}}\right) = 1\right]\right| \leq \mathrm{negl}(\lambda),$$

where U_{Y}, $\mathrm{U}_{\mathrm{Y}^*}$ and $\mathrm{U}_{\hat{\mathrm{Y}}}$ denote tuples of $(2n \cdot \bar{N})$, $(N^* \cdot \bar{N})$ and $(\widehat{N} \cdot \bar{N})$ uniform elements in $\mathcal{Y}$, respectively, while $\widetilde{\mathrm{U}}_{\mathrm{Y}^*}$ and $\widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}$ are as described in Lemmas 3.3.4 and 3.3.5, respectively. We prove this lemma through a sequence of hybrid arguments. Define the collection $\{\mathrm{Y}^{(\bar{n})}\}_{\bar{n}\in[0,\bar{N}]}$ as follows:

- $\mathrm{Y}^{(0)}$ is identical to Y as output by the "real" protocol.
- $\mathrm{Y}^{(\bar{N})}$ is identical to U_{Y}.
- For each $\bar{n} \in [\bar{N}]$, $\mathrm{Y}^{(\bar{n})}$ is identical to $\mathrm{Y}^{(\bar{n}-1)}$ except that the subtuple of elements $\{y^{(\bar{n})}_{j,b}\}_{j\in[n],b\in\{0,1\}}$ is replaced by $2n$ elements sampled uniformly at random from $\mathcal{Y}$.

Similarly, define the collection $\{\widetilde{\mathrm{U}}^{(\bar{n})}_{\mathrm{Y}^*}\}_{\bar{n}\in[0,\bar{N}]}$ as follows.

- $\widetilde{\mathrm{U}}^{(0)}_{\mathrm{Y}^*}$ is identical to $\widetilde{\mathrm{U}}_{\mathrm{Y}^*}$ as described in Lemma 3.3.4.

- $\widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{N})}$ is identical to $\mathrm{U}_{\mathrm{Y}^*}$.
- For each $\bar{n} \in [\bar{N}]$, $\widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n})}$ is identical to $\widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n}-1)}$ except that the subtuple of elements $\{y^*_{\bar{n},n^*}\}_{n^*\in[N^*]}$ is replaced by N^* elements sampled uniformly at random from $\mathcal{Y}$.

Finally, define the collection $\{\widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n})}\}_{\bar{n}\in[0,\bar{N}]}$ as follows.

- $\widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(0)}$ is identical to $\widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}$ as described in Lemma 3.3.5.
- $\widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{N})}$ is identical to $\mathrm{U}_{\hat{\mathrm{Y}}}$.
- For each $\bar{n} \in [\bar{N}]$, $\widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n})}$ is identical to $\widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n}-1)}$ except that the subtuple of elements $\{\hat{y}_{\bar{n},\hat{n}}\}_{\hat{n}\in[\widehat{N}]}$ is replaced by $\widehat{N}$ elements sampled uniformly at random from $\mathcal{Y}$.

We first prove the following auxiliary lemma.

Lemma 3.3.7. *For any $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, for all functions $N^* = N^*(\lambda)$, $\bar{N} = \bar{N}(\lambda)$ and $\widehat{N} = \widehat{N}(\lambda)$, and for any PPT adversary $\mathcal{A}$,*

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}^{(\bar{n}-1)}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n}-1)}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n}-1)}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}^{(\bar{n})}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n})}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n})}\right) = 1\right]\right|$$

is negligible in the security parameter λ.

Proof. To prove this lemma, suppose that there exists a PPT adversary $\mathcal{A}$ such that

$$\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}^{(\bar{n}-1)}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n}-1)}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n}-1)}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}^{(\bar{n})}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n})}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n})}\right) = 1\right]\right|$$

is non-negligible for some $\bar{n} \in [\bar{N}]$. We construct a PPT algorithm $\mathcal{B}$ that breaks the weak pseudorandomness of F. $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ queries its oracle $(2n + N^* + \widehat{N})$ times and receives the following tuples

$$\left\{\{x_{j,b}, y_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{x^*_{n^*}, y^*_{n^*}\}_{n^*\in[N^*]}, \{\hat{x}_{\hat{n}}, \hat{y}_{\hat{n}}\}_{\hat{n}\in[N^*]}\right\}.$$

2. It then samples $(\bar{N} - \bar{n})$ keys as $k_{\bar{n}+1}, \ldots, k_{\bar{N}} \leftarrow \mathcal{K}$ and sets the following for each $n' \in [\bar{N}]$:

$$
\begin{aligned}
y_{j,b}^{(n')} &= \begin{cases} y_{j,b} & \text{if } n' = \bar{n}, \\ F(k_{n'}, x_{j,b}) & \text{if } n' > \bar{n}, \\ y \leftarrow \mathcal{Y} & \text{otherwise.} \end{cases} \qquad \text{for } j \in [n], b \in \{0,1\} \\
y_{n',n^*}^{*} &= \begin{cases} y_{n^*}^{*} & \text{if } n' = \bar{n}, \\ F(k_{n'}, x_{n^*}^{*}) & \text{if } n' > \bar{n}, \\ y \leftarrow \mathcal{Y} & \text{otherwise.} \end{cases} \qquad \text{for } n^* \in [N^*] \\
\hat{y}_{n',\hat{n}} &= \begin{cases} \hat{y}_{\hat{n}} & \text{if } n' = \bar{n}, \\ F(k_{n'}, \hat{x}_{\hat{n}}) & \text{if } n' > \bar{n}, \\ y \leftarrow \mathcal{Y} & \text{otherwise.} \end{cases} \qquad \text{for } \hat{n} \in [\widehat{N}]
\end{aligned}
$$

3. $\mathcal{B}$ then sets the following:

$$
\begin{aligned}
\mathrm{X} &= \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \quad \mathrm{U}_{\mathrm{X}^*} = \{x_{n^*}^{*}\}_{n^*\in[N^*]}, \quad \mathrm{U}_{\hat{\mathrm{X}}} = \{\hat{x}_{\hat{n}}\}_{\hat{n}\in[N^*]}, \\
\mathrm{Y}' &= \left\{y_{j,b}^{(1)}, \ldots, y_{j,b}^{(\bar{N})}\right\}_{j\in[n],b\in\{0,1\}}, \\
\widetilde{\mathrm{U}}'_{\mathrm{Y}^*} &= \left\{y_{1,n^*}^{*}, \ldots, y_{\bar{N},n^*}^{*}\right\}_{n^*\in[N^*]}, \quad \widetilde{\mathrm{U}}'_{\hat{\mathrm{Y}}} = \{\hat{y}_{1,\hat{n}}, \ldots, \hat{y}_{\bar{N},\hat{n}}\}_{\hat{n}\in[N^*]},
\end{aligned}
$$

   and sends the tuple $\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}', \widetilde{\mathrm{U}}'_{\mathrm{Y}^*}, \widetilde{\mathrm{U}}'_{\hat{\mathrm{Y}}}\right)$ to $\mathcal{A}$.

4. Eventually, $\mathcal{A}$ outputs a bit b. $\mathcal{B}$ outputs the same bit b.

Observe the following:

- When $\mathcal{B}$ interacts with the weak PRF, the distribution of $\left(\mathrm{Y}', \widetilde{\mathrm{U}}'_{\mathrm{Y}^*}, \widetilde{\mathrm{U}}'_{\hat{\mathrm{Y}}}\right)$ is identical to the distribution of $\left(\mathrm{Y}^{(\bar{n}-1)}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n}-1)}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n}-1)}\right)$.
- When $\mathcal{B}$ interacts with a random function, the distribution of $\left(\mathrm{Y}', \widetilde{\mathrm{U}}'_{\mathrm{Y}^*}, \widetilde{\mathrm{U}}'_{\hat{\mathrm{Y}}}\right)$ is identical to the distribution of $\left(\mathrm{Y}^{(\bar{n})}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n})}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n})}\right)$.

It follows that $\mathbf{Adv}^{\mathrm{IHwUF}}(\mathcal{B})$ is negligibly different from the advantage of $\mathcal{A}$, which completes the proof of Lemma 3.3.7.

To prove Lemma 3.3.6, observe that

$$\begin{aligned}
&\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}\right)=1\right]-\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{U}_{\mathrm{Y}}, \mathrm{U}_{\mathrm{Y}^*}, \mathrm{U}_{\hat{\mathrm{Y}}}\right)=1\right]\right| \\
\leq & \sum_{\bar{n}=1}^{\bar{N}}\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}^{(\bar{n}-1)}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n}-1)}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n}-1)}\right)=1\right]-\right. \\
&\left.\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}^{(\bar{n})}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}^{(\bar{n})}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}^{(\bar{n})}\right)=1\right]\right| \leq \operatorname{negl}(\lambda).
\end{aligned}$$

This completes the proof of Lemma 3.3.6. □

Putting everything together. Using Lemmas 3.3.4, 3.3.5, and 3.3.6 we have

$$\begin{aligned}
&\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}, \mathrm{Y}^*, \hat{\mathrm{Y}}\right)=1\right]-\Pr\left[\mathcal{A}\left(\mathrm{U}_{\mathrm{X}}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{U}_{\mathrm{Y}}, \mathrm{U}_{\mathrm{Y}^*}, \mathrm{U}_{\hat{\mathrm{Y}}}\right)=1\right]\right| \\
\leq & \left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}, \mathrm{Y}^*, \hat{\mathrm{Y}}\right)=1\right]-\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \hat{\mathrm{X}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \hat{\mathrm{Y}}\right)=1\right]\right|+ \\
&\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \hat{\mathrm{X}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \hat{\mathrm{Y}}\right)=1\right]-\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}\right)=1\right]\right|+ \\
&\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{Y}, \widetilde{\mathrm{U}}_{\mathrm{Y}^*}, \widetilde{\mathrm{U}}_{\hat{\mathrm{Y}}}\right)=1\right]-\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{U}_{\mathrm{Y}}, \mathrm{U}_{\mathrm{Y}^*}, \mathrm{U}_{\hat{\mathrm{Y}}}\right)=1\right]\right|+ \\
&\left|\Pr\left[\mathcal{A}\left(\mathrm{X}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{U}_{\mathrm{Y}}, \mathrm{U}_{\mathrm{Y}^*}, \mathrm{U}_{\hat{\mathrm{Y}}}\right)=1\right]-\Pr\left[\mathcal{A}\left(\mathrm{U}_{\mathrm{X}}, \mathrm{U}_{\mathrm{X}^*}, \mathrm{U}_{\hat{\mathrm{X}}}, \mathrm{U}_{\mathrm{Y}}, \mathrm{U}_{\mathrm{Y}^*}, \mathrm{U}_{\hat{\mathrm{Y}}}\right)=1\right]\right| \\
\leq & \operatorname{negl}(\lambda),
\end{aligned}$$

since X and U_{X} are both distributed uniformly over $\mathcal{X}^{2n}$. This completes the proof of Theorem 3.3.3.

Security (IHwUF). The following theorem captures the security guarantees provided by the general framework when instantiated using an IHwUF.

Theorem 3.3.8. *Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an* IHwUF. *For any $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, for all functions $N^* = N^*(\lambda)$, $\bar{N} = \bar{N}(\lambda)$ and $\widehat{N} = \widehat{N}(\lambda)$, for any arbitrary $(n^*, \hat{n}, \bar{n}) \in [N^*] \times [\widehat{N}] \times [\bar{N}]$, and for any PPT adversary $\mathcal{A} = (\mathcal{A}^*, \hat{\mathcal{A}})$, we have*

$$\Pr\left[\mathcal{A}^*\big(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}\big) = y^*_{\bar{n},n^*}\right] + \Pr\left[\hat{\mathcal{A}}\big(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}\big) = \hat{y}_{\bar{n},\hat{n}}\right] \leq \operatorname{negl}(\lambda),$$

where the tuple $(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}, y^*_{\bar{n},n^*}, \hat{y}_{\bar{n},\hat{n}})$ for $(n^*, \hat{n}, \bar{n}) \in [N^*] \times [\widehat{N}] \times [\bar{N}]$ is as defined in the protocol above.

Proof. Suppose that there exists a PPT adversary $\mathcal{A} = (\mathcal{A}^*, \hat{\mathcal{A}})$ such that

$$\Pr\left[\mathcal{A}^*\big(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}\big) = y^*_{\bar{n},n^*}\right] + \Pr\left[\hat{\mathcal{A}}\big(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}\big) = \hat{y}_{\bar{n},\hat{n}}\right]$$

is non-negligible for some $(n^*, \hat{n}, \bar{n}) \in [N^*] \times [\widehat{N}] \times [\bar{N}]$.

- Assume that there exists $(n^*, \bar{n}) \in [N^*] \times [\bar{N}]$ such that $\Pr\left[\mathcal{A}^*(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}) = y^*_{\bar{n},n^*}\right]$ is non-negligible. We construct a PPT algorithm $\mathcal{B}$ that breaks the weak unpredictability of F. $\mathcal{B}$ proceeds as follows:

  1. $\mathcal{B}$ queries its oracle $2n$ times and receives the tuple $\{x_{j,b}, y_{j,b}\}_{j\in[n],b\in\{0,1\}}$, and a challenge $x^* \in \mathcal{X}$.
  2. It then samples $(\bar{N}-1)$ PRF as $k_1, \ldots, k_{\bar{n}-1}, k_{\bar{n}+1}, \ldots, k_{\bar{N}} \leftarrow \mathcal{K}$ and sets the following for each $n' \in [\bar{N}]$, $j \in [n]$, $b \in \{0,1\}$:

$$y^{(n')}_{j,b} \quad = \quad \begin{cases} y_{j,b} & \text{if } n' = \bar{n} \\ F(k_{n'}, x_{j,b}) & \text{otherwise.} \end{cases}$$

  3. $\mathcal{B}$ then samples (N^*-1) random group elements as

$$x^*_1, \ldots, x^*_{n^*-1}, x^*_{n^*+1}, x^*_{N^*} \leftarrow \mathcal{X}$$

     and sets $x^*_{n^*} = x^*$, where x^* is the input challenge to $\mathcal{B}$.
  4. $\mathcal{B}$ additionally samples $\hat{N}$ group elements $\{\hat{x}_{\hat{n}}\}_{\hat{n}\in\hat{N}}$ and sends the tuple $\big(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}\big)$ to $\mathcal{A}$, where

$$\mathrm{X} = \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}\,, \quad \mathrm{X}^* = (x^*_1, \ldots, x^*_{N^*})\,, \quad \hat{\mathrm{X}} = (\hat{x}_1, \ldots, \hat{x}_{\hat{N}})\,,$$

$$\mathrm{Y} = \left\{y^{(1)}_{j,b}, \ldots, y^{(\bar{N})}_{j,b}\right\}_{j\in[n],b\in\{0,1\}}.$$

  5. Eventually, $\mathcal{A}$ outputs $y^* \in \mathcal{Y}$. $\mathcal{B}$ outputs the same y^*.

By the leftover hash lemma, the distributions of X^* and $\hat{\mathrm{X}}$ are statistically indistinguishable from those of the real protocol. It follows that $\mathbf{Adv}^{\mathrm{IHwUF}}(\mathcal{B})$ is negligibly different from the advantage of $\mathcal{A}^*$.

- A similar argument shows that if there exists a pair of indices $(\hat{n}, \bar{n}) \in [\hat{N}] \times [\bar{N}]$ such that $\Pr\left[\hat{\mathcal{A}}(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}) = \hat{y}_{\bar{n},\hat{n}}\right]$ is non-negligible, then there exists an attacker against the weak unpredictability of F with non-negligible advantage.

Combining the aforementioned items, for any $(n^*, \hat{n}, \bar{n}) \in [N^*] \times [\widehat{N}] \times [\bar{N}]$, and for any PPT adversary $\mathcal{A} = (\mathcal{A}^*, \hat{\mathcal{A}})$, we have

$$\Pr\left[\mathcal{A}^*\big(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}\big) = y^*_{\bar{n},n^*}\right] + \Pr\left[\hat{\mathcal{A}}\big(\mathrm{X}, \mathrm{X}^*, \hat{\mathrm{X}}, \mathrm{Y}\big) = \hat{y}_{\bar{n},\hat{n}}\right] \leq \mathrm{negl}(\lambda),$$

which completes the proof of Theorem 3.3.8.

Lemma 3.3.9. *Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an* IHwUF. *For any $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, for any $\mathbf{s} \in \{0,1\}^n$ and any (fixed) $i \in [n]$, if $x_{j,b} \leftarrow \mathcal{X}$ and $k \leftarrow \mathcal{K}$ we have*

$$\big(\mathbf{s}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{F(k,x_{j,b})\}_{j\in[n],b\in\{0,1\}} \setminus \{F(k,x_{i,1-s_i})\}, \text{HardCore}(F(k, \bigoplus_{j\in[n]} x_{j,s_j}))\big) \overset{c}{\approx}$$
$$\big(\mathbf{s}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{F(k,x_{j,b})\}_{j\in[n],b\in\{0,1\}} \setminus \{F(k,x_{i,1-s_i})\}, \beta\big),$$

where $\beta \leftarrow \{0,1\}$ is a uniform bit.

Proof. Since $F(k, x_{i,1-s_i})$ is not given, by the input homomorphism of F, it is enough to show the following statement

$$\big(\{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{F(k,x_{j,b})\}_{j\in[n],b\in\{0,1\}} \setminus F(k,x_{i,b^*}), \text{HardCore}(F(k,x_{i,b^*}))\big) \overset{c}{\approx}$$
$$\big(\{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{F(k,x_{j,b})\}_{j\in[n],b\in\{0,1\}} \setminus F(k,x_{i,b^*}), \beta\big),$$

which follows from the weak unpredictability of F. □

Corollary 3.3.10. *Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an* IHwUF. *For any $n > \log|\mathcal{X}| + \omega(\log(\lambda))$ and any (fixed) $i \in [n]$, if $x_{j,b} \leftarrow \mathcal{X}$, $\mathbf{s} \leftarrow \{0,1\}^n$, and $k \leftarrow \mathcal{K}$ we have*

$$\big(\{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{F(k,x_{j,b})\}_{j\in[n],b\in\{0,1\}} \setminus \{F(k,x_{i,1-s_i})\}, \text{HardCore}(F(k, \bigoplus_{j\in[n]} x_{j,s_j}))\big) \overset{c}{\approx}$$
$$\big(\{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{F(k,x_{j,b})\}_{j\in[n],b\in\{0,1\}} \setminus \{F(k,x_{i,1-s_i})\}, \beta\big),$$

where $\beta \leftarrow \{0,1\}$ is a uniform bit.

Proof. The proof is similar to the proof of Lemma 3.3.9. □

3.4 Instantiations from Cryptographic Assumptions

In this section, we provide instantiations of IHwUFs/IHwPRFs from concrete assumptions. We start with Diffie-Hellman assumption as an easy example. We then provide an example of IHwPRF where the input space *depends* on the secret key. After that we provide an example of IHwPRF where a bounded number of homomorphisms is allowed. Finally, we prove that any (group-)homomorphic PKE implies an IHwPRF family, providing constructions of IHwPRF family from different concrete assumptions.

CDH/DDH. Given a group $(\mathbb{G}, \cdot)$ of order q with generator g, define the function $F : \mathbb{Z}_q \times \mathbb{G} \to \mathbb{G}$ as $F(k, h) = h^k$. We prove that F is an IHwPRF assuming DDH assumption. It is easy to see that for any $h_1, h_2 \in \mathbb{G}$ we have $F(k, h_1 \cdot h_2) = F(k, h_1) \cdot F(k, h_2)$. Let $\mathcal{A}$ be an attacker against the weak pseudorandomness of F, and let n be the number of queries of the attacker. It is enough to show that

$$((g^{x_1}, g^{kx_1}), (g^{x_2}, g^{kx_2}), \ldots, (g^{x_n}, g^{kx_n})) \overset{c}{\approx} ((g^{x_1}, g^{r_1}), (g^{x_2}, g^{r_2}), \ldots, (g^{x_n}, g^{r_n})),$$

where $k, x_i, r_i \leftarrow \mathbb{Z}_q$ are uniform and independent for $i \in [n]$, and (g^{x_i}, g^{kx_i}) is the answer to the ith query.

Given a DDH-challenge tuple (g^k, g^{x^*}, y) where y is either g^{kx^*} or a random element of $\mathbb{G}$, the reduction first samples n pairs of random elements: $\{(s_i, t_i) \leftarrow \mathbb{Z}_q^2\}_{i \in [n]}$. It then outputs the following tuple:

$$\left((g^{s_1}(g^{x^*})^{t_1}, (g^k)^{s_1} y^{t_1}), \ldots, (g^{s_n}(g^{x^*})^{t_n}, (g^k)^{s_n} y^{t_n})\right).$$

Observe that when $y = g^{kx^*}$ the tuple above is identical to the "real" game where all queries answered as PRF output, and y random corresponds to the "ideal" game where all queries answered as a random function. Therefore, an attacker against the weak pseudorandomness of F with advantage ε implies a DDH distinguisher with advantage ε. A similar argument shows that F is an IHwUF under CDH assumption.

Matrix-DDH. The DDH problem has been generalized to several different algebraic problems, like Decisional Linear (DLIN [BBS04]) and k-linear [HK07, Sha07]. Escala et al [EHK$^+$13] generalized all these assumptions into one framework called the matrix-DDH assumptions. Given a cyclic group $(\mathbb{G}, \cdot)$ of order q with generator g and a $\mathbb{Z}_q$-matrix $\mathbf{A}$, they adopted the notation $[\mathbf{A}]$ to denote the component-wise exponentiation $g^{\mathbf{A}}$. The matrix-DDH problem is parameterized

by a distribution $\mathcal{D}_{\ell,k}$ on $\mathbb{Z}_q^{\ell\times k}$ matrices with $(\ell > k)$, where the top $k \times k$ matrix, denoted $\bar{\mathbf{A}}$, is overwhelmingly invertible. The remaining $(\ell - k) \times k$ bottom matrix is denoted $\underline{\mathbf{A}}$. The $\mathcal{D}_{\ell,k}$ matrix-DDH assumption states that with $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$ and $(\mathbf{w}, \mathbf{r}) \leftarrow \mathbb{Z}_q^k \times \mathbb{Z}_q^{\ell-k}$:

$$\left([\bar{\mathbf{A}}\mathbf{w}], [\underline{\mathbf{A}}\mathbf{w}]\right) \overset{c}{\approx} \left([\bar{\mathbf{A}}\mathbf{w}], [\mathbf{r}]\right).$$

We now show that any $\mathcal{D}_{\ell,k}$-matrix-DDH assumption can give us an IHwPRF. The key sampling algorithm is as follows: first sample $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$ and then compute key $\mathbf{K} = \bar{\mathbf{A}}^{-1}\underline{\mathbf{A}}$. Now we define the function $F : \mathbb{Z}_q^{k\times(\ell-k)} \times \mathbb{G}^k \to \mathbb{G}^{\ell-k}$ as $F(\mathbf{K}, [\mathbf{x}]) = \mathbf{K}[\mathbf{x}] = [\mathbf{K}\mathbf{x}]$. We prove that F is an IHwPRF assuming the $\mathcal{D}_{\ell,k}$-Matrix-DDH assumption. It is easy to see that for any $[\mathbf{x}_1], [\mathbf{x}_2] \in \mathbb{G}^k$ we have

$$F(\mathbf{K}, [\mathbf{x}_1 + \mathbf{x}_2]) = F(\mathbf{K}, [\mathbf{x}_1]) + F(\mathbf{K}, [\mathbf{x}_2]).$$

Let $\mathcal{A}$ be an attacker against the weak pseudorandomness of F, and let n be the number of queries of the attacker. It is enough to show that

$$(([\mathbf{x}_1], [\mathbf{K}\mathbf{x}_1]), \ldots, ([\mathbf{x}_n], [\mathbf{K}\mathbf{x}_n])) \overset{c}{\approx} (([\mathbf{x}_1], [\mathbf{r}_1]), \ldots, ([\mathbf{x}_n], [\mathbf{r}_n])),$$

where $\mathbf{x}_i, \mathbf{r}_i$ are uniform and independent for $i \in [n]$, and $[\mathbf{x}_i], [\mathbf{K}\mathbf{x}_i]$ is the answer to the i-th query.

It was shown in [EHK+13] that by the random self-reducibility of matrix-DDH samples, the advantage of any attacker in distinguishing the above distributions is bounded by a multiplicative factor of $(\ell - k)$ over a single sample matrix-DDH advantage. Therefore, an attacker against the weak pseudorandomness of F with advantage ε implies a Matrix-DDH distinguisher with advantage $\varepsilon/(\ell - k)$. A similar argument shows that F is an IHwUF under the computational analog of matrix-DDH assumption.

Quadratic Residuosity. Let $N = pq$ be a composite modulus where p and q are randomly generated equal-size primes, and let $\mathcal{J}_N^{+1}$ be the set of all elements in $\mathbb{Z}_N^*$ with Jacboi symbol 1. Define $F(k = (p, q), x \in \mathcal{J}_N^{+1})$ as follows:

$$F(k, x) = \begin{cases} 0 & \text{if } x \in \mathcal{QR}_N, \\ 1 & \text{if } x \notin \mathcal{QR}_N. \end{cases}$$

First, given the factorization of N one can efficiently determine whether an element $x \in \mathcal{J}_N^{+1}$ is

a quadratic residue. Moreover, Observe that for any $x_1, x_2 \in \mathcal{J}_N^{+1}$ we have

$$F(k, x_1 x_2) = F(k, x_1) + F(k, x_2)$$

where $x_1 x_2$ is the product of x_1 and x_2 in $\mathbb{Z}_N^*$, and $+$ is addition modulo 2.[1] A simple hybrid argument similar to the case of DDH construction implies the weak pseudorandomness of F. It follows that F is an IHwPRF under QR assumption.

Remark 3.4.1. We note that although F is an IHwPRF (and hence IHwUF), it has a limitation that the input space *depends* on the key. In some applications, it is necessary to know the input space *before* generating the key.

By a similar argument it is also possible to construct an IHwUF from RSA assumption. However, similar to the case of QR, the input space (implicitly) depends on the choice of the key.

LWE. We now sketch a construction of IHwPRF from the LWE assumption [Reg05]. Let n, q be the parameters of the LWE assumption where n is the dimension of the secret and q is the modulus. Also, let χ denote the (Gaussian-like) noise distribution. Let GSamp be a Gaussian sampler algorithm that on a uniformly chosen input $\mathbf{u} \leftarrow \{0,1\}^\ell$, outputs a sample according to χ. First, we use a weak PRF $F_N : \mathcal{K}_N \times \mathbb{Z}_q^n \to \{0,1\}^\ell$ to generate the randomness for GSamp algorithm.[2] We define the bounded IHwPRF[3] $F : \mathcal{K} \times \mathbb{Z}_q^n \to \mathbb{Z}_q$ as

$$F((k, \mathbf{s}), \mathbf{a}) = \langle \mathbf{s}, \mathbf{a} \rangle + \mathsf{GSamp}(F_N(k, \mathbf{a}))$$

$$\mathcal{R}(b \in \mathbb{Z}_q) = \begin{cases} 0 & |b| \leq q/4 \\ 1 & |b| > q/4 \end{cases}$$

where $\mathcal{K} = \mathcal{K}_N \times \mathbb{Z}_q^n, k \leftarrow \mathcal{K}_N$ and $\mathbf{s} \leftarrow \mathbb{Z}_q^n$. The weak pseudorandomness of F follows from a simple hybrid argument, and we omit the details here. As for bounded homomorphism, observe that if q is sufficiently large (superpolynomial in n), for any $\gamma = \mathrm{poly}(n) \leq q/n$ we have

$$\mathcal{R}\Big(F\Big((k, \mathbf{s}), \sum_{i \in [\gamma]} \mathbf{a}_i\Big)\Big) = \mathcal{R}\Big(\sum_{i \in [\gamma]} F((k, \mathbf{s}), \mathbf{a}_i)\Big).$$

[1] Recall that for any $x_1, x_2 \in \mathcal{J}_N^{+1}$, the product $x_1 x_2$ is a quadratic non-residue if and only if exactly one of them is a quadratic non-residue.

[2] Note that we do not need any homomorphism for F_N, and it is just used to generated the noise for LWE samples.

[3] See Definition 3.1.6 for a formal definition of bounded IHwPRF.

Remark 3.4.2. It is easy to see that if q is polynomial in n, the probability that the equality above does not hold is bounded by $1/\operatorname{poly}(n)$. We remark that for almost all of the applications in this thesis (except the case of non-interactive key exchange) one can use polynomial modulus simply by repeating cryptographic protocol with independent randomness.[1] One can analogously construct a bounded IHwPRF family based on the Ring LWE assumption [LPR10].

DCR/FFI/AGCD/HNP. We now show that an IHwUF/IHwPRF is implied by any assumption that yields a (group-)homomorphic PKE. Informally, the decryption algorithm of any homomorphic PKE can be viewed as an IHwPRF, where the ciphertext space and the message space are the input space and the output space of the IHwPRF, respectively. We stress that here we use a slight generalization of the definition of weak pseudorandomness where the input/key is sampled according to some efficiently samplable distribution over the input/key space and these distributions are not necessarily uniform. However, most of the instantiations from concrete assumptions results in a uniform distribution over the key/input space.

Lemma 3.4.3. *Let* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be a CPA-secure homomorphic PKE. Let* $\mathcal{K}$*,* $(\mathcal{M}, \otimes)$*, and* $(\mathcal{C}, \oplus)$ *be the key space, message space, and ciphertext space of* Π*, respectively. The function family* F *defined as*

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^\lambda), \quad F(\mathsf{sk} \in \mathcal{K}, c \in \mathcal{C}_{\mathsf{pk}}) = \mathsf{Dec}(\mathsf{sk}, c) = m \in \mathcal{M},$$

is an IHwPRF family, where $\mathcal{C}_{\mathsf{pk}}$ *denotes to set of all valid ciphertexts under the public key* pk*.*

Proof. Observe that by homomorphism of Π, for any $c_1, c_2 \in \mathcal{C}_{\mathsf{pk}}$ we have

$$F(\mathsf{sk}, c_1 \oplus c_2) = F(\mathsf{sk}, c_1) \otimes F(\mathsf{sk}, c_2),$$

which implies the (γ-bounded) homomorphism of F. Now we show the weak pseudorandomness of F. We define a distribution $\mathcal{D}$ over $\mathcal{C}_{\mathsf{pk}}$ as follows. To sample according to $\mathcal{D}$, first generate a uniform $m \leftarrow \mathcal{M}$ and let $c = \mathsf{Enc}(\mathsf{pk}, m)$ be the encryption of m using a fresh randomness. Let $\mathcal{A}$ be an adversary against the weak pseduorandomness of F, and let n be the number of queries made by $\mathcal{A}$. We define $n + 1$ hybrids as follows. Let H_j be a hybrid that the first $j - 1$ queries of the adversary are answered as $(c_i \leftarrow \mathcal{D}, F(\mathsf{sk}, c_i))$ for $i \in [j - 1]$, and the remaining queries are answered as (c_i, m_i) where m_i is generated randomly and independent of c_i. It is enough to

[1] As an example, for the case of PKE, the encryptor publishes polynomially many encryptions of the same message, and the decryptor can recover the message with probability $1 - \operatorname{negl}(\lambda)$ simply by taking a majority over the decrypted messages.

show that for each $i \in [n]$ the hybrids H_{i-1} and H_i are computationally indistinguishable. To do so, given an attacker $\mathcal{A}$ that distinguishes H_{i-1} and H_i, we build an attacker $\mathcal{B}$ that breaks the semantic security of Π. The attacker $\mathcal{B}$ answers jth query of $\mathcal{A}$ as follows:

- If $j \in [i-1]$, $\mathcal{B}$ samples $m \leftarrow M$ and computes $c \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$. It then sends (c, m) to $\mathcal{A}$.
- If $i = j$, $\mathcal{B}$ samples two uniform messages $m^{(0)}, m^{(1)} \leftarrow \mathcal{M}$ and sends them to its challenger. Upon receiving c^* (challenge ciphertext), $\mathcal{B}$ sends $(c^*, m^{(1)})$ to $\mathcal{A}$.
- If $i + 1 \leq j \leq n$, $\mathcal{B}$ samples $m \leftarrow M$ and computes $c \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$. It then sends (c, r) to $\mathcal{A}$ where r is sampled independently and uniformly over $\mathcal{M}$.

If $\mathcal{A}$ outputs 1, $\mathcal{B}$ also outputs 1. Otherwise, $\mathcal{B}$ outputs 0. Since both of $m^{(0)}$ and $m^{(1)}$ are generated uniformly at random, c^* is distributed according to $\mathcal{D}$. If c^* is an encryption of $m^{(1)}$ we have $F(\mathsf{sk}, c^*) = m^{(1)}$ and hence the reduction maps encryption of $m^{(1)}$ to a valid weak PRF output. On the other hand, if c^* is an encryption of $m^{(0)}$, then c^* is independent of $m^{(0)}$ and hence the reduction maps encryption of $m^{(0)}$ to a random pair $(c^*, m^{(0)})$ where c^* is distributed according to $\mathcal{D}$ and $m^{(0)}$ is uniform. Therefore, the advantage of $\mathcal{B}$ in the CPA security game is equal to the advantage of $\mathcal{A}$ in distinguishing H_{i-1} and H_i. □

Observe that a similar proof also shows that a γ-bounded homomorphic PKE implies a bounded IHwPRF. Therefore, Lemma 3.4.3 immediately yields an IHwPRF from the Decisional Composite Residuosity [Pai99] assumption. It also yields constructions of (γ-bounded) IHwPRF family from several assumptions, e.g., Hidden Number Problem [BV96], Approximate GCD [How01], and Finite Field Isomorphism [DHP+18].

Instantiations of HOWF from Cryptographic Assumptions. It is easy to see that the following assumptions yields HOWF family. Specifically:

- The function family defined by $f_g(x) = g^x$ is an HOWF family based on discrete log assumption since $f_g(x_1 + x_2) = g^{x_1} \cdot g^{x_2} = f_g(x_1) \cdot f_g(x_2)$.
- Let $N = pq$ be an RSA modulus where p and q are equal-size prime numbers. The function family defined by $f_N(x) = x^2$ is an HOWF family based on factorization assumption since $f_N(x_1 x_2) = f_N(x_1) \cdot f_N(x_2)$.
- Let $N = pq$ be an RSA modulus as above, and let $e \leftarrow \mathbb{Z}^*_{\varphi(N)}$. The function family defined by $f_{N,e}(x) = x^e$ is an HOWF family based on RSA assumption since it holds that $f_{N,e}(x_1 x_2) = f_{N,e}(x_1) \cdot f_{N,e}(x_2)$.

It is also easy to describe instantiations of (γ-bounded) HOWF family from assumptions other than the few ones mentioned above. See instantiations of IHwUF/IHwPRF for more details.

CHAPTER 4

Primitives from HOWF

In this chapter, we present constructions of various cryptographic primitives from (bounded) HOWF, including collision-resistant hash function (CRHF), Schnorr-style signature and chameleon hash function. Collision-resistant hash function (and Schnorr-like protocols) from structured primitives have been around for many years. Ogata and Kurosawa [OK93] demonstrated that *homomorphic one-way permutations* imply claw-free permutations. Ishai *et al.* [IKO05] constructed CRHF from homomorphic encryption and homomorphic one-way commitments. Maurer [Mau09] showed Schnorr-style zero-knowledge proof of knowledge protocols from (unbounded) HOWF.

4.1 Collision-Resistant Hash Function

In this section, we show that any HOWF implies a collision-resistant hash function family. Given any HOWF $f : \mathcal{X} \to \mathcal{Y}$, let $\mathbf{x} = \{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ be a vector of $2n$ uniform elements for some fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$. Define $\mathbf{y}$ as

$$\mathbf{y} = \{y_{j,b} = f(x_{j,b})\}_{j\in[n],b\in\{0,1\}}.$$

Now, define the function family $\mathcal{H}_{\mathbf{y}} : \{0,1\}^n \to \mathcal{Y}$ as

$$\mathcal{H}_{\mathbf{y}}\left(\mathbf{r} = (r_1, \cdots, r_n)\right) = \bigotimes_{j\in[n]} y_{j,r_j}.$$

Collision resistance. We now show that the function family $\mathcal{H}_{\mathbf{y}}$ is collision resistant. Define the experiment $\mathsf{Expt}^{\text{CRHF}-\text{HOWF}}$ as follows.

1. The challenger uniformly samples $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$.
2. The challenger sets $y_{j,b} = f(x_{j,b})$ for $j \in [n], b \in \{0,1\}$ and sends $\{y_{j,b}\}_{j\in[n],b\in\{0,1\}}$ to the adversary $\mathcal{A}$.
3. The adversary $\mathcal{A}$ outputs two strings $\mathbf{r} \in \{0,1\}^n$ and $\mathbf{r}' = \{0,1\}^n$.

For any PPT adversary $\mathcal{A}$ we define $\mathbf{Adv}^{\text{CRHF}-\text{HOWF}}(\mathcal{A})$ to be the probability of the event that

$$\bigotimes_{j\in[n]} y_{j,r_j} = \bigotimes_{j\in[n]} y_{j,r'_j}.$$

Lemma 4.1.1. *For all PPT adversaries $\mathcal{A}$, we have $\mathbf{Adv}^{\text{CRHF}-\text{HOWF}}(\mathcal{A}) = \text{negl}(\lambda)$.*

Proof. Let $\mathcal{A}$ be a PPT adversary such that $\mathbf{Adv}^{\text{CRHF}-\text{HOWF}}(\mathcal{A})$ is non-negligible. We construct a PPT algorithm $\mathcal{B}$ that breaks the one-wayness of f. $\mathcal{B}$ proceeds as follows:

1. $\mathcal{B}$ receives a challenge query $y^* \in \mathcal{Y}$ such that $y^* = f(x^*)$ for some (randomly chosen) $x^* \leftarrow \mathcal{X}$.
2. $\mathcal{B}$ samples $2n$ uniformly random elements from $\mathcal{X}$ as $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ and sets $y_{j,b}$ as
$$y_{j,b} = f(x_{j,b}) \quad \text{for } j \in [n], b \in \{0,1\}.$$
3. $\mathcal{B}$ uniformly samples $i \leftarrow [n]$ and $b^* \leftarrow \{0,1\}$, and resets $y_{i,b^*} := y^*$. It then forwards $\{y_{j,b}\}_{j\in[n],b\in\{0,1\}}$ to the adversary $\mathcal{A}$.
4. $\mathcal{A}$ outputs $\mathbf{r} \in \{0,1\}^n$ and $\mathbf{r}' \in \{0,1\}^n$.
5. $\mathcal{B}$ proceeds as follows:
   - If $\bigotimes_{j\in[n]} y_{j,r_j} \neq \bigotimes_{j\in[n]} y_{j,r'_j}$ or $r_i = r'_i$, it outputs a uniformly random $x^* \leftarrow \mathcal{X}$.
   - Otherwise, assume wlog that $r_i = b^*$. Then, the following must hold
$$y^* = \Big(\bigotimes_{j\in[i-1]} y_{j,r_j}\Big)^{-1} \otimes \Big(\bigotimes_{j\in[n]} y_{j,r'_j}\Big) \otimes \Big(\bigotimes_{j\in[i+1,n]} y_{j,r_j}\Big)^{-1},$$

where the right-hand side is independent of y^*. Finally, $\mathcal{B}$ outputs x^* as

$$x^* = \Big(\bigoplus_{j \in [i-1]} x_{j,r_j} \Big)^{-1} \oplus \Big(\bigoplus_{j \in [n]} x_{j,r'_j} \Big) \oplus \Big(\bigoplus_{j \in [i+1,n]} x_{j,r_j} \Big)^{-1}.$$

By the input homomorphism of f, we have $f(x^*) = y^*$.

Observe that if $\mathcal{A}$ outputs a valid collision $(\mathbf{r}, \mathbf{r}')$, the probability that $\mathbf{r}$ and $\mathbf{r}'$ differ in the ith bit for a randomly chosen $i \leftarrow [n]$ is at least $1/n$. It follows that

$$\mathbf{Adv}^{\text{HOWF}}(\mathcal{B}) \geq \left(\frac{\mathbf{Adv}^{\text{CRHF}-\text{HOWF}}(\mathcal{A})}{n} \right),$$

which is non-negligible, as desired.

Note 4.1.2. The aforementioned CRHF family may be equivalently instantiated from any γ-bounded HOWF family, subject to the restriction that $n \leq \gamma$.

4.2 Schnorr Signature

We show how to construct a Schnorr-style signature scheme from any HOWF family. The signature scheme is existentially unforgeable against adaptive chosen-message attacks in the programmable random oracle model.

- $\mathsf{Setup}(1^\lambda)$: Let $f : \mathcal{X} \to \mathcal{Y}$ be an HOWF, and let $H : \mathcal{Y} \times \{0,1\}^* \to \{0,1\}^n$ be a hash function (modeled as a random oracle in the security proof). The setup algorithm fixes some integer $n > \log |\mathcal{X}| + \omega(\log(\lambda))$ and outputs the public parameter pp as

$$\mathsf{pp} = (f, n, H).$$

- $\mathsf{Gen}(\mathsf{pp})$: The key generation algorithm samples $2n$ elements as $\{x_{j,b} \leftarrow \mathcal{X}\}_{j \in [n], b \in \{0,1\}}$ and computes $y_{j,b} = f(x_{j,b})$ for $j \in [n]$ and $b \in \{0,1\}$. It outputs the signing key sk and the verification key vk as

$$\mathsf{sk} = \{x_{j,b}\}_{j \in [n], b \in \{0,1\}}, \quad \mathsf{vk} = \{y_{j,b}\}_{j \in [n], b \in \{0,1\}}.$$

- $\mathsf{Sign}(\mathsf{sk}, \mathsf{m})$: Given the signing key $\mathsf{sk} = \{x_{j,b}\}_{j \in [n], b \in \{0,1\}}$ and a message $\mathsf{m} \in \{0,1\}^*$, the algorithm samples $x \leftarrow \mathcal{X}$ and sets $y = f(x)$. It then sets the vector $\mathbf{r} \in \{0,1\}^n$ as

$\mathbf{r} = H(y, \mathsf{m})$. Finally, it outputs the signature $\sigma = (\mathbf{r}, \hat{x}, y) \in \{0,1\}^n \times \mathcal{X} \times \mathcal{Y}$, where

$$\hat{x} = x \oplus \Big(\bigoplus_{j \in [n]} x_{j,r_j} \Big)^{-1}.$$

- $\mathsf{Ver}(\mathsf{vk}, \mathsf{m}, \sigma)$: Given the verification key $\mathsf{vk} = \{y_{j,b}\}_{j \in [n], b \in \{0,1\}}$, a message $\mathsf{m} \in \{0,1\}^*$, and a signature $\sigma = (\mathbf{r}, \hat{x}, y)$, the verification algorithm checks if the following holds:

$$\mathbf{r} = H(y, \mathsf{m}), \quad y = f(\hat{x}) \otimes \Big(\bigotimes_{j \in [n]} y_{j,r_j} \Big).$$

  If yes, it accepts the signature. Otherwise, it rejects.

Correctness follows from the homomorphism of f. In order to prove existential unforgeability under an adaptively chosen-message attack in the programmable random oracle model, we resort to the forking lemma [PS00]. We first prove the following lemma.

Lemma 4.2.1. *If H is modeled as a random oracle, there exists a PPT simulator $\mathcal{S}$ that produces (with non-negligible probability) a signature $\tilde{\sigma}$ on any arbitrary message m without the knowledge of the signing key sk such that the distribution of $\tilde{\sigma}$ is statistically indistinguishable from that of $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$.*

Proof. The simulator $\mathcal{S}$ receives the verification key $\mathsf{vk} = \{y_{j,b}\}_{j \in [n], b \in \{0,1\}}$ and proceeds as follows:

- The simulator $\mathcal{S}$ uniformly samples $\mathbf{r} = (r'_1, \ldots, r'_n) \leftarrow \{0,1\}^n$ and $\tilde{x} \leftarrow \mathcal{X}$.
- It sets $y' = f(\tilde{x}) \otimes \big(\bigotimes_{j \in [n]} y_{j,r'_j} \big)$ and returns the signature $\tilde{\sigma} = (\mathbf{r}', \tilde{x}, y')$.

Observe that in the simulation, we must have $y' = f(x')$, where

$$x' = \tilde{x} \oplus \Big(\bigoplus_{j \in [n]} x_{j,r'_j} \Big).$$

Since $\tilde{x}$ is uniform in $\mathcal{X}$, so is x'. Hence, the distribution of $(\tilde{x}, y' = f(x'))$ in the simulation is statistically indistinguishable from that of $(\hat{x}, y^* = f(x^*))$ in the real signing algorithm. Finally, under the assumption that H is a random oracle, the distribution of the string $\mathbf{r}'$ in the simulation is also statistically indistinguishable from that of the string $\mathbf{r} = H(y^*, \mathsf{m})$ in the real signing algorithm. This completes the proof of Lemma 4.2.1. □

Let $\mathcal{A}$ be a PPT adversary that performs an existential forgery attack against the aforementioned signature scheme with probability ε, while making $Q_1 = Q_1(\lambda)$ signing queries and $Q_2 = Q_2(\lambda)$ random oracle queries, such that $\varepsilon \geq 10(Q_1+1)(Q_1+Q_2)/2^\lambda$. Let (m, σ) be the resulting forgery, where $\sigma = (\mathbf{r}, \hat{x}, y^*)$. As shown by Pointcheval and Stern in [PS00], Lemma 4.2.1 implies the following forking lemma.

Lemma 4.2.2. *A polynomial-time replay of the adversary $\mathcal{A}$ where its interactions with the signing oracle are replaced by interactions with the simulator $\mathcal{S}$ as described above, produces two valid message-signature pairs (with non-negligible probability)*

$$(\mathsf{m}, \sigma = (\mathbf{r}, \hat{x}, y)), \quad (\mathsf{m}, \sigma' = (\mathbf{r}', \tilde{x}, y')),$$

such that $\sigma \neq \sigma'$.

Finally, given a PPT adversary that forges a pair of non-identical signatures on the same message with non-negligible probability, one can construct a PPT adversary that outputs a collision on the CRHF family described in Section 4.1 with the same probability. This completes the proof of existential unforgeability for the signature scheme.

Note 4.2.3. The aforementioned signature scheme has an a priori bounded number of homomorphic operations, which allows it to be instantiated using a γ-bounded HOWF family, subject to the restriction that $n + 1 \leq \gamma$.

4.3 Chameleon Hash Function

We now show how to construct a chameleon hash function family from any HOWF. The formal definition of chameleon hash functions is presented below.

Definition 4.3.1. (Chameleon Hash Function.) A chameleon hash function family is defined as a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{CHash}, \mathsf{TrpCollision})$ described below.

- $\mathsf{Setup}(1^\lambda)$: Given λ, it outputs the public parameter pp and a trapdoor t.
- $\mathsf{CHash}\,(\mathsf{pp}, \mathbf{s}; r)$: Given pp, a string $\mathbf{s} \in \{0,1\}^n$ (where $n = n(\lambda)$ is included in pp) and randomness r, it outputs a hash h.
- $\mathsf{TrpCollision}\,(\mathsf{t}, (\mathbf{s}, r)\,, \mathbf{s}')$: Given the trapdoor t, a string $\mathbf{s} \in \{0,1\}^n$, some randomness r, and a string $\mathbf{s}' \in \{0,1\}^n$, it outputs some randomness r'.

The following properties must be satisfied:

- **Uniformity:** If $(\mathsf{pp}, \mathsf{t}) \leftarrow \mathsf{Setup}(1^\lambda)$, then for all string pairs $\mathbf{s}, \mathbf{s}' \in \{0,1\}^n$ and uniformly sampled randomness pairs (r, r'), the two distributions $\mathsf{CHash}(\mathsf{pp}, \mathbf{s}; r)$ and $\mathsf{CHash}(\mathsf{pp}, \mathbf{s}'; r')$ are statistically indistinguishable.

- **Collision resistance:** For any PPT adversary $\mathcal{A}$ if $(\mathsf{pp}, \mathsf{t}) \leftarrow \mathsf{Setup}(1^\lambda)$, it holds that

$$\Pr[\mathbf{s} \neq \mathbf{s}' \wedge \mathsf{CHash}(\mathsf{pp}, \mathbf{s}; r) = \mathsf{CHash}(\mathsf{pp}, \mathbf{s}'; r')] \leq \mathrm{negl}(\lambda),$$

  where $((\mathbf{s}, r), (\mathbf{s}', r')) \leftarrow \mathcal{A}(\mathsf{pp})$.

- **Trapdoor collisions:** If $(\mathsf{pp}, \mathsf{t}) \leftarrow \mathsf{Setup}(1^\lambda)$, then for all $\mathbf{s}, \mathbf{s}' \in \{0,1\}^n$ and randomness r, it holds that

$$\mathsf{CHash}(\mathsf{pp}, \mathbf{s}; r) = \mathsf{CHash}(\mathsf{pp}, \mathbf{s}'; r'),$$

  where $r' = \mathsf{TrpCollision}(\mathsf{pp}, (\mathbf{s}, r), \mathbf{s}')$.

Construction from HOWF. We construct a chameleon hash function family from any HOWF as follows.

- $\mathsf{Setup}\left(1^\lambda\right)$: Given an HOWF $f : \mathcal{X} \to \mathcal{Y}$, the setup algorithm samples $2n$ elements from $\mathcal{X}$ as $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ for some fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$. It outputs the public parameter pp and the trapdoor t as:

$$\mathsf{pp} = \left(f, \{y_{j,b}\}_{j\in[n],b\in\{0,1\}}\right), \quad \mathsf{t} = \{x_{j,b}\}_{j\in[n],b\in\{0,1\}},$$

  where $y_{j,b} = f(x_{j,b})$ for $j \in [n]$ and $b \in \{0,1\}$.

- $\mathsf{CHash}(\mathsf{pp}, \mathbf{s}; r)$: Given the public parameter $\mathsf{pp} = \left(f, \{y_{j,b}\}_{j\in[n],b\in\{0,1\}}\right)$, a binary vector $\mathbf{s} \in \{0,1\}^n$, and randomness $r \leftarrow \mathcal{X}$, the hashing algorithm outputs the hash h as

$$h = \Big(\bigotimes_{j\in[n]} y_{j,s_j}\Big) \otimes f(r).$$

- $\mathsf{TrpCollision}(\mathsf{t}, (\mathbf{s}, r), \mathbf{s}')$: Given the trapdoor $\mathsf{t} = \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}$, a string $\mathbf{s} \in \{0,1\}^n$, some randomness $r \in \mathcal{X}$, and another string $\mathbf{s}' = (s'_1, \ldots, s'_n) \in \{0,1\}^n$, the equivocation

algorithm outputs $r' \in \mathcal{X}$ as

$$r' = \Big(\bigoplus_{j \in [n]} x_{j,s'_j} \Big)^{-1} \oplus \Big(\bigoplus_{j \in [n]} x_{j,s_j} \Big) \oplus r.$$

We now argue that the aforementioned construction satisfies the desired properties of a chameleon hash function.

- **Uniformity:** Let $\mathbf{s} = (s_1, \ldots, s_n) \in \{0,1\}^n$ and $\mathbf{s}' = (s'_1, \ldots, s'_n) \in \{0,1\}^n$ be arbitrary binary strings, and let $r, r' \leftarrow \mathcal{X}$ be uniformly random elements in $\mathcal{X}$. Let

  $$x^* = \Big(\bigoplus_{j \in [n]} x_{j,s_j} \Big) \oplus r, \quad x' = \Big(\bigoplus_{j \in [n]} x_{j,s'_j} \Big) \oplus r'.$$

  It is easy to see that both x^* and x' are uniformly distributed over $\mathcal{X}$ so long as r and r' are uniform and independent. This in turn implies that the distributions

  $$\mathsf{CHash}\left(\mathsf{pp}, \mathbf{s}; r\right) = f\left(x^*\right), \quad \mathsf{CHash}\left(\mathsf{pp}, \mathbf{s}'; r'\right) = f\left(x'\right),$$

  are both statistically indistinguishable from the distribution $\{f(x)\}_{x \leftarrow \mathcal{X}}$. This completes the proof of uniformity.

- **Collision resistance.** Suppose that there exists a PPT adversary $\mathcal{A}$ that produces with non-negligible probability ε a tuple $((\mathbf{s}, r), (\mathbf{s}', r'))$ such that $(\mathbf{s}, r) \neq (\mathbf{s}', r')$ and

  $$\mathsf{CHash}\left(\mathsf{pp}, \mathbf{s}; r\right) = \mathsf{CHash}\left(\mathsf{pp}, \mathbf{s}'; r'\right).$$

  An argument similar to the one used in proof of Theorem 4.1.1 can be used to demonstrate the existence of a a PPT algorithm $\mathcal{B}$ that breaks the one-wayness of f with non-negligible probability.

- **Trapdoor Collisions:** It is straightforward to verify that trapdoor collisions produced by the scheme are valid.

Note 4.3.2. The aforementioned chameleon hash construction has an a priori bounded number of homomorphic operations, which allows it to be instantiated similarly using a γ-bounded HOWF family, subject to the restriction that $n + 1 \leq \gamma$.

Implications. The following are some implications of chameleon hash functions:

- Chameleon hash function implies (in a black-box manner) noninteractive statistically hiding and computationally binding trapdoor commitment scheme [KR00], which in turn implies resettable zero-knowledge proof for NP [CGGM00].
- Chameleon hash function implies *chameleon signature* [KR00], which is a noninteractive undeniable signature that guarantees both non-repudiation and non-transferability.

4.4 On the Existence of HOWF in Quantum Setting

We show that any homomorphic one-way function (HOWF) with exact homomorphism over abelian groups can be broken using a quantum algorithm. Since exact KHwPRF over abelian groups trivially imply unbounded HOWF, it follows that there is no secure construction of an unbounded KHwPRF in quantum world. As a result, a secure KHwPRF either needs to have an approximate homomorphism, or the homomorphism should hold over a nonabelian group.

At a high level, given any abelian group with certain conditions there are known quantum algorithms to determine the structure of the group. That is, given an abelian group $\mathcal{G}$, there is an efficient quantum algorithm to find (an efficiently computable) isomorphism $\psi : \mathcal{G} \to \mathbb{Z}_{q_1} \oplus \cdots \oplus \mathbb{Z}_{q_m}$. We apply this to both the input and output group of a candidate HOWF f. Then we show a simple classic algorithm that given these isomorphisms over the input and output group of f, one can simply break one-wayness of f.

Theorem 4.4.1. *Let $f : \mathcal{X} \to \mathcal{Y}$ be a (classic) HOWF such that $\mathcal{X}$ and $\mathcal{Y}$ are abelian groups, and there exists an efficient algorithm to find a generating set for $\mathcal{Y}$. There exists a polynomial quantum algorithm that breaks the one-wayness of f with non-negligible advantage.*[1]

First we recall the following fact from algebra. A proof can be found in any standard textbook.

Theorem 4.4.2. *Any finite abelian group is isomorphic to a direct sum of cyclic groups, and each cyclic group has a prime power order.*

We rely on the following quantum algorithm (see [CM01] and Section 6.2 of [Chi17] for more details).

[1]Note that a set of uniform elements with size $3\log|\mathcal{Y}|$ forms a generating set with an overwhelming probability.

Theorem 4.4.3. *Let $\mathcal{G}$ be a finite abelian group such that (1) each element of $\mathcal{G}$ has a unique decoding, (2) there is an efficient algorithm to do group operations on the elements of $\mathcal{G}$, and (3) there is an efficient algorithm to find a generating set for $\mathcal{G}$. There is a polynomial time quantum algorithm such that decomposes the group $\mathcal{G}$ as*

$$\mathcal{G} = \langle g_1 \rangle \oplus \cdots \oplus \langle g_M \rangle,$$

in terms of the generators $g_1, \ldots, g_M$, and for every $m, m' \in [M]$ such that $m \neq m'$ we have $\langle g_m \rangle \cap \langle g_{m'} \rangle = \{e\}$, where e is the identity element of $\mathcal{G}$. Moreover, the isomorphism

$$\psi : \mathcal{G} \to \mathbb{Z}_{|\langle g_1 \rangle|} \oplus \cdots \oplus \mathbb{Z}_{|\langle g_M \rangle|},$$

(in both ways) can be computed efficiently.

Now we are ready to proceed to the proof of Theorem 4.4.1. Let $f : \mathcal{X} \to \mathcal{Y}$ be an unbounded HOWF such that $\mathcal{X}$ and $\mathcal{Y}$ are abelian groups. Given a challenge $y^* \in \mathcal{Y}$ such that $y^* := f(x^*)$ for some uniform $x^* \leftarrow \mathcal{X}$, we want to find a preimage x such that $f(x) = y^*$. Let

$$\tilde{\mathcal{X}} := \mathbb{Z}_{p_1} \oplus \cdots \oplus \mathbb{Z}_{p_M} \quad , \quad \tilde{\mathcal{Y}} := \mathbb{Z}_{q_1} \oplus \cdots \oplus \mathbb{Z}_{q_N}$$

be the decomposition of groups $\mathcal{X}$ and $\mathcal{Y}$, respectively, where p_i (respectively, q_j) is a prime power for $m \in [M]$ (respectively, $n \in [N]$). We fix some arbitrary order for the cyclic groups, and we call $\tilde{\mathcal{X}}$ an *explicit representation* of $\mathcal{X}$. Using Theorem 4.4.3, we can efficiently compute the isomorphisms $\psi_{\mathcal{X}}, \psi_{\mathcal{Y}}$ (and their inverses) for any element in the domain of the isomorphisms where

$$\psi_{\mathcal{X}} : \mathcal{X} \to \tilde{\mathcal{X}} \quad , \quad \psi_{\mathcal{Y}} : \mathcal{Y} \to \tilde{\mathcal{Y}}.$$

Define $\tilde{f} : \tilde{\mathcal{X}} \to \tilde{\mathcal{Y}}$ as the analog of f over the explicit representations of $\mathcal{X}$ and $\mathcal{Y}$, i.e., define

$$\tilde{f}(\tilde{x}) = \psi_{\mathcal{Y}}(f(\psi_{\mathcal{X}}^{-1}(\tilde{x}))).$$

It is not hard to see that $f(x) = y$ is equivalent to $\tilde{f}(\psi_{\mathcal{X}}(x)) = \psi_{\mathcal{Y}}(y)$. Because the isomorphisms $\psi_{\mathcal{X}}, \psi_{\mathcal{Y}}$ and their inverses are efficiently computable, it is enough to show an attack against the one-wayness of $\tilde{f}$.

For each $n \in [N]$, we define $\mathbf{e}_n \in \mathbb{Z}_{p_1} \oplus \cdots \oplus \mathbb{Z}_{p_N}$ to be the (unit) vector whose nth component is 1, and all other components are 0.[1] For an element $\tilde{y} \in \tilde{\mathcal{Y}}$, let $[\tilde{y}]_m \in \mathbb{Z}_{q_m}$ be the mth component

[1]Notice that each component may live in a different cyclic group.

of $\tilde{y}$. We compute the index set I_m for each $m \in M$ as

$$I_m = \{n \in [N] \mid [\tilde{f}(\mathbf{e}_n)]_m \neq 0\}.$$

All index sets $\{I_m\}_{m\in[M]}$ can be computed efficiently since both N and M are polynomially bounded. Define a vector of variables $\mathbf{z} = (z_1, \ldots, z_N) \in \mathbb{Z}^N$, and for each $m \in [M]$, consider the following system of modular equations where $\{z_i\}_{i\in I_m}$ are the (unknown) variables:

$$S_m : \quad \sum_{i\in I_m} z_i[\tilde{f}(\mathbf{e}_i)]_m \equiv [\tilde{y}]_m \pmod{q_m}.$$

Consider the following observations:

- Without loss of generality we can assume that for two distinct $m, m' \in [M]$, we have $\gcd(q_m, q'_m) = 1$. If $q_m = q'_m$, we can simply merge S_m and $S_{m'}$. If $q_m < q_{m'}$ and $\gcd(q_m, q'_m) > 1$, we can "lift" the equation in $S_{m'}$ simply by multiplying the both sides by $q^{m'}/q^m$ and adding the resulting equation to $S_{m'}$. We refer to this part as "merging" step.

- Observe that for any two *prime powers* p and q, if there is a non-trivial homomorphism from $\mathbb{Z}_p$ to $\mathbb{Z}_q$ then either $p \mid q$ or $q \mid p$. Therefore, if z_n appears in S_m (or equivalently $n \in I_m$), we either have $p_n \mid q_m$ or $q_m \mid p_n$.

Let $\overline{M} \subseteq M$ be the set of indices after the "merging" step. Using the previous observations, it follows that

- For any two distinct $\overline{m}_1, \overline{m}_2 \in \overline{M}$, we have $\gcd(q_{\overline{m}_1}, q_{\overline{m}_2}) = 1$.

- For any $n \in N$, there is at most one $\overline{m} \in \overline{M}$ such that the variable z_n appears in $S_{\overline{m}}$.

Each system of equation(s) $S_{\overline{m}}$ can be seen as a system of linear equation(s) over the group $\mathbb{Z}_{q_{\overline{m}}}$, and it can solved using the known algorithms for solving linear equations over finite abelian groups, e.g., [GR02]. One can equivalently interpret each $S_{\overline{m}}$ as a system of equations over the finite *ring* $\mathbb{Z}_{q_{\overline{m}}}$ (since $q_{\overline{m}}$ is not necessarily prime).

By solving each system $S_{\overline{m}}$, we can determine the vector $\mathbf{z} \in \mathbb{Z}^N$. Finally, we output $\tilde{x}$ as a preimage of $\tilde{y}$ where

$$\tilde{x} = (z_1 \bmod p_1, \ldots, z_N \bmod p_n).$$

By construction, we know that the vector $\mathbf{z}$ satisfies all system of equation(s) $\{S_m\}_{m\in M}$. It follows that $\tilde{f}(\tilde{x}) = \tilde{y}$, as required.

CHAPTER 5

Primitives from IHwUF

We present constructions of various cryptographic primitives from (bounded) IHwUF. We also show how to instantiate these constructions using the general protocol of Chapter 3.

5.1 Noninteractive Key Exchange

We present a noninteractive key exchange protocol between non-uniform PPT algorithms $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ and $\mathcal{B} = (\mathcal{B}_0, \mathcal{B}_1)$. It allows exchange of a single bit and is obtained from an IHwUF in a black-box manner (note that $\mathcal{A}_b$ and $\mathcal{B}_b$ operate in parallel for $b \in \{0,1\}$).

- $\mathsf{Setup}(1^\lambda)$: Given the security parameter λ, the setup algorithm creates a description $\mathcal{F}_{\text{IHwUF}}$ for an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$. It uniformly samples $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ for a fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, and outputs the public parameter pp as

$$\mathsf{pp} = \left(\mathcal{F}_{\text{IHwUF}}\ ,\ \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}\right).$$

- $\mathcal{A}_0\,(\mathsf{pp})$: On input $\mathsf{pp} = \left(\mathcal{F}_{\text{IHwUF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}\right)$, the algorithm $\mathcal{A}_0$ first samples a random $\mathbf{s} = (s_1, \ldots, s_n) \leftarrow \{0,1\}^n$ and then outputs $(\mathsf{st}_{\mathcal{A}}, x^*_{\mathcal{A}})$, where

$$\mathsf{st}_{\mathcal{A}} = \mathbf{s} \quad , \quad x^*_{\mathcal{A}} = \bigoplus_{j\in[n]} x_{j,s_j}.$$

- $\mathcal{B}_0\,(\mathsf{pp})$: On input $\mathsf{pp} = \left(\mathcal{F}_{\text{IHwUF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}\right)$, the algorithm $\mathcal{B}_0$ first samples a key $k \leftarrow \mathcal{K}$ and computes $y_{j,b} = F\,(k, x_{j,b})$ for each $j \in [n]$ and $b \in \{0,1\}$. It then outputs $(\mathsf{st}_{\mathcal{B}}, \mathbf{y}_{\mathcal{B}})$, where

$$\mathsf{st}_{\mathcal{B}} = k \quad , \quad \mathbf{y}_{\mathcal{B}} = \left(\{y_{j,b}\}_{j\in[n],b\in\{0,1\}}\right).$$

- $\mathcal{A}_1\left(\mathsf{pp}, \mathsf{st}_{\mathcal{A}}, \mathbf{y}_{\mathcal{B}}\right)$: On input $\mathsf{st}_{\mathcal{A}} = \mathrm{s}$ and $\mathbf{y}_{\mathcal{B}}$, the algorithm $\mathcal{A}_1$ computes the final bit as

$$\mathsf{k}^* = \text{HardCore}\Big(\bigotimes_{j\in[n]} y_{j,s_j}\Big).$$

- $\mathcal{B}_1\left(\mathsf{pp}, \mathsf{st}_{\mathcal{B}}, x^*_{\mathcal{A}}\right)$: On input $\mathsf{st}_{\mathcal{B}} = k$ and $x^*_{\mathcal{A}}$, the algorithm $\mathcal{B}_1$ computes the final bit as

$$\mathsf{k}^* = \text{HardCore}\left(F\left(k, x^*_{\mathcal{A}}\right)\right).$$

Instantiation from the general protocol. The aforementioned NIKE scheme can be instantiated from the general protocol described in Section 3.3 as follows:

- **Initialization.** Instantiate the protocol using an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with description $\mathcal{F}_{\text{IHwUF}}$, a fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, $N^* = 1$, $\bar{N} = 1$, and $\hat{N} = 0$. Set $\mathsf{pp} = \mathrm{X}$.
- **Pre-Evaluation:** Let $\mathrm{X}^* = \{x^*\}$ be the output of the pre-evaluation phase. Set $x^*_{\mathcal{A}} = x^*$.
- **Evaluation:** Let Y and $\mathrm{Y}^* = \{y^*\}$ be the outputs of the evaluation phase. Set $\mathbf{y}_{\mathcal{B}} = \mathrm{Y}$ and $\mathsf{k}^* = \text{HardCore}\left(y^*\right)$.

Correctness and security of the scheme follow from Claim 3.3.1 and Theorem 3.3.8, respectively.

Note 5.1.1. The aforementioned construction has an a priori bounded number of homomorphic operations, which allows it to be instantiated equivalently using a γ-bounded IHwUF if $\gamma \geq n$, with the following minor modification to the final key-generation step:

$$\mathsf{k}^* = \text{HardCore}\Big(\mathcal{R}\Big(\bigotimes_{j\in[n]} y_{j,s_j}\Big)\Big) = \text{HardCore}\left(\mathcal{R}\left(F\left(k, x^*_{\mathcal{A}}\right)\right)\right),$$

where $\mathcal{R} : \mathcal{Y} \to \mathcal{Z}$ is a universal map (see Definition 3.1.5).

Note 5.1.2. The aforementioned NIKE protocol can only be instantiated using an IHwUF family for which the input space is *independent* of the key.[1]

5.2 Passively Secure Encryption

We present a CPA-secure public-key encryption scheme from any IHwUF. First we provide a formal definition of a CPA-secure PKE scheme and next we state the construction.

[1]Note that this property does not hold for some instantiations from concrete assumptions, e.g., QR assumption. See Section 3.4 for more details.

Definition 5.2.1. (CPA-Secure PKE.) Let $\Pi = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme. Π is said to be CPA-secure if for all PPT adversaries $\mathcal{A}$, the views of $\mathcal{A}$ in the games $\mathsf{Expt}_0^{\text{ind-cpa}}$ and $\mathsf{Expt}_1^{\text{ind-cpa}}$ are computationally indistinguishable.

Experiment $\mathsf{Expt}_b^{\text{ind-cpa}}$:

1. The challenger runs the setup algorithm and generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}\,(\mathsf{pp})$, and provides pk to the adversary $\mathcal{A}$.

2. The adversary $\mathcal{A}$ issues a challenge encryption query for a pair of messages $(\mathsf{m}_0, \mathsf{m}_1)$. The challenger creates the challenge ciphertext

$$\mathsf{ct}^* \leftarrow \mathsf{Enc}\,(\mathsf{pk}, \mathsf{m}_b)\,,$$

and sends ct^* to the adversary $\mathcal{A}$.

We now present a CPA-secure PKE from any IHwUF family.

- $\mathsf{Setup}(1^\lambda)$: The setup algorithm creates a description $\mathcal{F}_{\text{IHwUF}}$ for an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$. It also fixes some integer $n > \log|\mathcal{X}| + \omega(\log(\lambda))$. The algorithm outputs $\mathcal{F}_{\text{IHwUF}}$ and n as the public parameter pp.

- $\mathsf{Gen}\,(\mathsf{pp})$: The key-generation algorithm samples $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ and a key $k \leftarrow \mathcal{K}$, and outputs

$$\mathsf{sk} = k \quad , \quad \mathsf{pk} = \{x_{j,b}, y_{j,b}\}_{j\in[n],b\in\{0,1\}},$$

where $y_{j,b} = F(k, x_{j,b})$ for each $j \in [n]$ and $b \in \{0,1\}$.

- $\mathsf{Enc}\,(\mathsf{pk}, \mathsf{m})$: Given the public key $\mathsf{pk} = \{x_{j,b}, y_{j,b}\}_{j\in[n],b\in\{0,1\}}$ and a single bit $\mathsf{m} \in \{0,1\}$, the encryption algorithm uniformly samples $\mathbf{r} = (r_1, \ldots, r_n) \leftarrow \{0,1\}^n$ and outputs the ciphertext $\mathsf{ct} = (c, \mathsf{e})$, where

$$c = \bigoplus_{j\in[n]} x_{j,r_j} \quad , \quad \mathsf{e} = \text{XOR}\Big(\text{HardCore}\Big(\bigotimes_{j\in[n]} y_{j,r_j}\Big), \mathsf{m}\Big).$$

- $\mathrm{Dec}\,(\mathsf{sk}, \mathsf{ct})$: Given the secret key $\mathsf{sk} = k$ and the ciphertext $\mathsf{ct} = (c, \mathsf{e})$, output the bit

$$\mathsf{m}' = \mathrm{XOR}\left(\mathrm{HardCore}\left(F(k, c)\right), \mathsf{e}\right).$$

Instantiation from the general protocol. The aforementioned PKE scheme can be instantiated from the general protocol described in Section 3.3 as follows:

- **Initialization.** Instantiate the protocol using an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with description $\mathcal{F}_{\text{IHwUF}}$, a fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, $N^* = 0$, $\bar{N} = 1$ and $\hat{N} = 1$. Set $\mathsf{pk}_1 = \mathrm{X}$.
- **Evaluation:** Set $\mathsf{sk} = k$ and $\mathsf{pk}_2 = \mathrm{Y}$, where Y is the output of the evaluation phase. Output $(\mathsf{sk}, \mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2))$.
- **Post-Evaluation:** Let $\hat{\mathrm{X}} = \{\hat{x}\}$ and $\hat{\mathrm{Y}} = \{\hat{y}\}$ be the outputs of the post-evaluation phase. Set:

$$c = \hat{x}, \quad \mathsf{e} = \mathrm{XOR}\left(\mathrm{HardCore}(\hat{y}), \mathsf{m}\right),$$

  where m is the message.

Correctness and security of the scheme follow from Claim 3.3.1 and Theorem 3.3.8, respectively.

Note 5.2.2. The aforementioned construction has an a priori bounded number of homomorphic operations, which allows it to be instantiated equivalently using a γ-bounded IHwUF if $\gamma \geq n$, with the following minor modification to the encryption algorithm:

$$\mathsf{e} = \mathrm{XOR}\Big(\mathrm{HardCore}\Big(\mathcal{R}\Big(\bigotimes_{j\in[n]} y_{j,r_j}\Big)\Big), \mathsf{m}\Big),$$

and the following minor modification to the decryption algorithm:

$$\mathsf{m}' = \mathrm{XOR}\left(\mathrm{HardCore}\left(\mathcal{R}\left(F(k, c)\right)\right), \mathsf{e}\right)$$

where $\mathcal{R} : \mathcal{Y} \to \mathcal{Z}$ is a universal map (see Definition 3.1.5).

Note 5.2.3. The aforementioned PKE can also be instantiated using a (γ-bounded) IHwUF family for which the input space depends on the key with the following modification: the setup algorithm only outputs the description of the key space and the output space of the IHwUF, while the description of the input space is published along with the public key by the key generation algorithm.

5.3 Trapdoor Functions

In this section, we show that IHwUFs imply trapdoor functions with almost-perfect correctness. Garg and Hajiabadi [GH18] introduced a primitive called recyclable *one-way function with encryption* (OWFE), and they showed that recyclable OWFEs imply TDFs (in a black-box way) with negligibly small inversion error. In this section, we demonstrate how to construct recyclable OWFE from IHwUF. We begin by presenting the formal definition of recyclable OWFE from [GH18], followed by our construction.

Definition 5.3.1. (Recyclable One-Way Function with Encryption.) A recyclable OWFE scheme is a tuple of four PPT algorithms $\mathsf{OWFE} = (\mathsf{Setup}, \mathsf{OWF}, \mathsf{Enc}, \mathsf{Dec})$ defined as follows:

- $\mathsf{Setup}\left(1^\lambda\right)$: Given the security parameter λ, it sets $n = n(\lambda)$ and $\ell = \ell(\lambda)$ for some fixed polynomial functions and outputs pp for a one-way function $\mathsf{OWF} : \{0,1\}^n \to \{0,1\}^\ell$.
- $\mathsf{OWF}\left(\mathsf{pp}, \mathbf{s}\right)$: Given the public parameter pp, it maps a string $\mathbf{s} \in \{0,1\}^n$ to an image $h \in \{0,1\}^\ell$.
- $\mathsf{Enc}\left(\mathsf{pp}, h, (i, b^*)\right)$: Given the public parameter pp, an image $h \in \{0,1\}^\ell$, an index $i \in [n]$ and a bit $b^* \in \{0,1\}$, it outputs a ciphertext ct and an additional bit $\mathsf{e} \in \{0,1\}$.
- $\mathsf{Dec}\left(\mathsf{pp}, \mathbf{s}, (i, b^*), \mathsf{ct}\right)$:[1] Given the public parameter pp, a preimage string $\mathbf{s}$, an index $i \in [n]$, a bit $b^* \in \{0,1\}$ and a ciphertext ct, it outputs $\mathsf{e}' \in \{0,1\} \cup \{\perp\}$.

The following correctness and security properties must be satisfied:

- **Correctness:** If $\mathsf{pp} \leftarrow \mathsf{Setup}\left(1^\lambda\right)$, then for all $\mathbf{s} = (s_1, \ldots, s_n) \in \{0,1\}^n$ and all $i \in [n]$, letting $h = \mathsf{OWF}\left(\mathsf{pp}, \mathbf{s}\right)$ and $b^* = s_i$, it holds with overwhelming probability over the randomness of Enc that if $(\mathsf{ct}, \mathsf{e}) \leftarrow \mathsf{Enc}\left(\mathsf{pp}, h, (i, b^*)\right)$, then we have

$$\mathsf{Dec}\left(\mathsf{pp}, \mathbf{s}, (i, b^*), \mathsf{ct}\right) = \mathsf{e}.$$

- **One-Wayness:** For any PPT adversary $\mathcal{A}$ we have

$$\Pr[\mathsf{OWF}(\mathsf{pp}, \mathcal{A}(h)) = h] \leq \mathrm{negl}(\lambda),$$

  where $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathbf{s} \leftarrow \{0,1\}^n$ and $h = \mathsf{OWF}(\mathsf{pp}, \mathbf{s})$.

[1] Notice that although e is part of the output of the encryption algorithm, the decryption algorithm does *not* take e as part of its input. The aim of the decryption algorithm is in fact to output e given only the ciphertext ct. This property is used in the construction of TDFs. See [GH18] for details.

- **Security:** For $b \in \{0,1\}$, define the experiment $\mathsf{Expt}_b^{\text{ind-OWFE}}$ between a challenger and an adversary $\mathcal{A}$ as follows:

Experiment $\mathsf{Expt}_b^{\text{ind-OWFE}}$:

1. The adversary $\mathcal{A}$ takes as input 1^n and 1^ℓ, and sends a string $\mathbf{s} = (s_1, \ldots, s_n) \in \{0,1\}^n$ and an index $i \in [n]$ to the challenger.
2. The challenger generates $\mathsf{pp} \leftarrow \mathsf{Setup}\left(1^\lambda\right)$. It computes $h = \mathsf{OWF}\left(\mathsf{pp}, \mathbf{s}\right)$ and $(\mathsf{ct}^*, \mathsf{e}_0^*) \leftarrow \mathsf{Enc}\left(\mathsf{pp}, h, (i, 1-s_i)\right)$. Finally, it samples $\mathsf{e}_1^* \leftarrow \{0,1\}$ and sends $(\mathsf{pp}, \mathsf{ct}^*, \mathsf{e}_b^*)$ to the adversary.

An OWFE scheme is said to be secure if for all PPT adversaries $\mathcal{A}$, the views of the adversary in $\mathsf{Expt}_0^{\text{ind-OWFE}}$ and $\mathsf{Expt}_1^{\text{ind-OWFE}}$ are computationally indistinguishable.

- **Recyclability:** An OWFE scheme is said to be recyclable if for all $\mathsf{pp} \in \mathsf{Setup}\left(1^\lambda\right)$, all $\mathbf{s}_1, \mathbf{s}_2 \in \{0,1\}^n$, all $i \in [n]$, all $b^* \in \{0,1\}$, and all randomness r, letting

$$(\mathsf{ct}_1, \mathsf{e}_1) \leftarrow \mathsf{Enc}\left(\mathsf{pp}, h_1 = \mathsf{OWF}\left(\mathsf{pp}, \mathbf{s}_1\right), (i, b^*); r\right),$$
$$(\mathsf{ct}_2, \mathsf{e}_2) \leftarrow \mathsf{Enc}\left(\mathsf{pp}, h_2 = \mathsf{OWF}\left(\mathsf{pp}, \mathbf{s}_2\right), (i, b^*); r\right),$$

we have $\mathsf{ct}_1 = \mathsf{ct}_2$.[1]

Recently, Garg *et al.* [GGH19] introduced an enhanced version of recyclable OWFE called *smooth* recyclable OWFE. A recyclable OWFE $= (\mathsf{Setup}, \mathsf{OWF}, \mathsf{Enc}, \mathsf{Dec})$ is said to be (ℓ, n)-*smooth* if for any two (ℓ, n)-sources $\mathcal{S}_1$ and $\mathcal{S}_2$ and for any PPT adversary $\mathcal{A}$, we have

$$\left|\Pr\left[\mathcal{A}(\mathsf{pp}, \mathsf{OWF}(\mathsf{pp}, \mathbf{s}_1)) = 1\right] - \Pr\left[\mathcal{A}(\mathsf{pp}, \mathsf{OWF}(\mathsf{pp}, \mathbf{s}_2)) = 1\right]\right| \leq \mathrm{negl}(\lambda),$$

where $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathbf{s}_1 \leftarrow \mathcal{S}_1$ and $\mathbf{s}_2 \leftarrow \mathcal{S}_2$.

Construction from IHwUF. We show a black-box construction of smooth recyclable OWFE from any IHwUF family.

- $\mathsf{Setup}\left(1^\lambda\right)$: Given the security parameter λ the setup algorithm creates a description $\mathcal{F}_{\text{IHwUF}}$ for an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, and fixes some integer $n > \log|\mathcal{X}| + \omega(\log(\lambda))$. It samples

[1] Informally, this means that the ct component is independent of the image h.

$2n$ elements as $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ and outputs the public parameter pp as

$$\mathsf{pp} = \left(\mathcal{F}_{\text{IHwUF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}\right).$$

- $\mathsf{OWF}\,(\mathsf{pp}, \mathrm{s})$: Given pp and a string $\mathrm{s} \in \{0,1\}^n$, generate the corresponding image as

$$h = \bigoplus_{j\in[n]} x_{j,s_j}.$$

- $\mathsf{Enc}\,(\mathsf{pp}, h, (i, b^*))$: Given pp, an image h, an index $i \in [n]$ and $b^* \in \{0,1\}$, the encryption algorithm samples $k \leftarrow \mathcal{K}$ and computes the following

$$y_{i,b^*} = F\left(k, x_{i,b^*}\right), \quad y_{i,1-b^*} = \perp,$$

$$y_{j,b} = F\left(k, x_{j,b}\right) \text{ for } j \in [n] \setminus \{i\}, b \in \{0,1\}.$$

  It finally outputs the pair

$$(\mathsf{ct}, \mathsf{e}) = \left(\{y_{j,b}\}_{j\in[n],b\in\{0,1\}}, \text{HardCore}\left(F\left(k,h\right)\right)\right).$$

- $\mathsf{Dec}\,(\mathsf{pp}, \mathrm{s}, (i, b^*), \mathsf{ct})$: Given pp, a string s, and $\mathsf{ct} = \left(\{y_{j,b}\}_{j\in[n],b\in\{0,1\}}\right)$, the decryption algorithm outputs

$$\mathsf{e}' = \begin{cases} \text{HardCore}\left(\bigotimes_{j\in[n]} y_{j,s_j}\right) & \text{if } s_i = b^* \\ \perp & \text{otherwise.} \end{cases}$$

Instantiation from the general protocol. The aforementioned OWFE scheme can be instantiated from the general protocol described in Section 3.3 as follows:

- **Initialization.** Instantiate the protocol using an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with description $\mathcal{F}_{\text{IHwUF}}$, a fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, $N^* = 1$, $\bar{N} = 1$ and $\hat{N} = 0$. Set $\mathsf{pp} = (\mathcal{F}_{\text{IHwUF}}, \mathrm{X})$.
- **Pre-Evaluation.** In the generic protocol, the pre-evaluation phase samples a uniformly random binary string $\mathrm{s} \in \{0,1\}^n$. One may view this as an input to the OWF. If $\mathrm{X}^* = \{x^*\}$ is the output of the post-evaluation phase, set the image $h = x^*$.
- **Evaluation:** Let $\mathrm{Y} = \{y_{j,b}\}_{j\in[n],b\in\{0,1\}}$ and $\mathrm{Y}^* = \{y^*\}$ be the outputs of the evaluation phase. For a given $i \in [n]$ and $b^* \in \{0,1\}^n$, set the ciphertext $\mathsf{ct} = \mathrm{Y} \setminus \{y_{i,1-b^*}\}$ and the bit $\mathsf{e} = \text{HardCore}\,(y^*)$.

To see that the instantiation satisfies the desired properties of a recyclable OWFE scheme, consider the following:

- Correctness follows from Claim 3.3.1. More specifically, given a binary string $\mathbf{s} \in \{0,1\}^n$ and ct such that $(\mathsf{ct}, e) = \mathsf{Enc}(\mathsf{pp}, h, (i, b^*))$ for $h = \mathsf{OWF}(\mathsf{pp}, \mathbf{s})$ and $s_i = b^*$, the decryption algorithm does not need $y_{i,1-b^*}$ to recover the bit e.
- One-wayness follows from Lemma 3.2.1.
- Security follows from Lemma 3.3.9.
- Recyclability follows from the fact that ct does not depend on the image h.

Finally, the aforementioned OWFE scheme is (ℓ, n)-smooth for any choice of $\ell \geq \log|\mathcal{X}| + \omega(\log\lambda)$. This follows directly from the leftover hash lemma. More specifically, let $(\mathcal{S}_1, \mathcal{S}_2)$ be (ℓ, n)-sources for $\ell \geq \log|\mathcal{X}| + \omega(\log\lambda)$. Then, for any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathbf{s}_1 \leftarrow \mathcal{S}_1$ and $\mathbf{s}_2 \leftarrow \mathcal{S}_2$, the distributions of $\mathrm{OWF}(\mathsf{pp}, \mathbf{s}_1)$ and $\mathrm{OWF}(\mathsf{pp}, \mathbf{s}_2)$ are statistically close to uniform by the leftover hash lemma.

Implications. Garg *et al.* [GGH19] showed that an (ℓ, n)-smooth recyclable OWFE scheme implies:

1. TDF with almost-perfect correctness, which is an improvement over TDFs with negligible inversion error (see [GH18, GGH19] for details).
2. CCA2-secure deterministic encryption, where the CCA2-security guarantee holds with respect to plaintexts sampled from (ℓ, n)-sources.

Note 5.3.2. The aforementioned construction has an apriori bounded number of homomorphic operations, which allows it to be instantiated equivalently using a γ-bounded IHwUF if $\gamma \geq n$, with the following minor modification to the encryption algorithm:

$$\mathsf{e} = \mathrm{HardCore}\left(\mathcal{R}\left(F\left(k, h\right)\right)\right),$$

and the following minor modification to the decryption algorithm:

$$\mathsf{e}' = \begin{cases} \mathrm{HardCore}\left(\mathcal{R}\left(\bigotimes_{j\in[n]} y_{j,s_j}\right)\right) & \text{if } s_i = b^*, \\ \perp & \text{otherwise.} \end{cases}$$

where $\mathcal{R} : \mathcal{Y} \to \mathcal{Z}$ is a universal map (see Definition 3.1.5).

Note 5.3.3. The aforementioned construction can only be instantiated from a (bounded) IHwUF family for which the input space is independent of the key.

5.4 Blind Batch Encryption

In this section, we show that IHwUF imply batch encryption, a cryptographic primitive introduced by Brakerski *et al.* in [BLSV18].[1] We now present the formal definition of blind batch encryption, followed by our construction.

Definition 5.4.1. (Batch Encryption.) A batch encryption scheme is a tuple of four PPT algorithms $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ defined as follows:

- $\mathsf{Setup}\left(1^\lambda\right)$: Given the security parameter λ, it outputs the public parameter pp.
- $\mathsf{Gen}\left(\mathsf{pp}, \mathbf{s}\right)$: Given pp, it projects the string $\mathbf{s} \in \{0,1\}^n$ to a hash value h where $n = n(\lambda)$ is some fixed polynomial included in pp.
- $\mathsf{Enc}\left(\mathsf{pp}, h, (i, \mathsf{m}_0, \mathsf{m}_1)\right)$: Given pp, a hash value h, an index $i \in [n]$ and a message pair $(\mathsf{m}_0, \mathsf{m}_1)$, it outputs a ciphertext $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2)$.
- $\mathsf{Dec}\left(\mathsf{pp}, \mathbf{s}, i, \mathsf{ct}\right)$: Given pp, a string $\mathbf{s}$, an index $i \in [n]$ and a ciphertext ct, it outputs a message m.

The following completeness, succinctness, security, and blindness properties must be satisfied:

- **Correctness:** If $\mathsf{pp} \leftarrow \mathsf{Setup}\left(1^\lambda\right)$, then for all $\mathbf{s} \in \{0,1\}^n$, all $i \in [n]$, and all message pairs $(\mathsf{m}_0, \mathsf{m}_1)$, letting $h = \mathsf{Gen}\left(\mathsf{pp}, \mathbf{s}\right)$ and $\mathsf{ct} \leftarrow \mathsf{Enc}\left(\mathsf{pp}, h, (i, \mathsf{m}_0, \mathsf{m}_1)\right)$ it holds with overwhelming probability over the randomness of Enc that

$$\mathsf{Dec}\left(\mathsf{pp}, \mathbf{s}, i, \mathsf{ct}\right) = \mathsf{m}_{s_i}.$$

- **Succinctness.** A batch encryption scheme is fully succinct if for some string $\mathbf{s} \in \{0,1\}^n$, letting $\mathsf{pp} \leftarrow \mathsf{Setup}\left(1^\lambda\right)$ and $h = \mathsf{Gen}\left(\mathsf{pp}, \mathbf{s}\right)$, we have $|h| \leq \mathrm{poly}\left(\lambda\right)$ for some *fixed* polynomial in the security parameter λ.
- **Security:** For each bit $b \in \{0,1\}$, define the following experiment $\mathsf{Expt}_b^{\text{ind-batch}}$ between a challenger and an adversary $\mathcal{A}$:

[1] See [DGHM18] for a closely relative primitive known as hash encryption.

Experiment $\mathsf{Expt}_b^{\text{ind-batch}}$:

1. The adversary $\mathcal{A}$ takes 1^λ and 1^n as input. It chooses an index $i \in [n]$ and a binary string $\mathbf{s} \in \{0,1\}^n$, and sends $(\mathbf{s}, i)$ to the challenger.
2. The challenger generates $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, and sends pp to the adversary $\mathcal{A}$.
3. The adversary $\mathcal{A}$ generates $\mathbf{m}^{(0)} = \left(\mathsf{m}_0^{(0)}, \mathsf{m}_1^{(0)}\right)$ and $\mathbf{m}^{(1)} = \left(\mathsf{m}_0^{(1)}, \mathsf{m}_1^{(1)}\right)$ such that $\mathsf{m}_{s_i}^{(0)} = \mathsf{m}_{s_i}^{(1)}$, and sends them to the challenger.
4. The challenger computes the hash $h = \mathsf{Gen}\left(\mathsf{pp}, \mathbf{s}\right)$, generates the ciphertext

$$\mathsf{ct}^* \leftarrow \mathsf{Enc}\big(\mathsf{pp}, h, \big(i, \mathsf{m}_0^{(b)}, \mathsf{m}_1^{(b)}\big)\big),$$

   and sends ct^* to the adversary $\mathcal{A}$.

A batch encryption scheme is said to be secure if for all PPT adversaries $\mathcal{A}$, the views of the adversary in $\mathsf{Expt}_0^{\text{ind-batch}}$ and $\mathsf{Expt}_1^{\text{ind-batch}}$ are computationally indistinguishable.

- **Blindness:** Let $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a batch encryption scheme such that one can view a ciphertext produced by the encryption algorithm as $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2)$, where ct_1 is produced by the subroutine Enc_1 and ct_2 is produced by the subroutine Enc_2. Also, for $b \in \{0,1\}$, define the experiment $\mathsf{Expt}_b^{\text{blind-batch}}$ between a challenger and an adversary $\mathcal{A}$ as follows:

Experiment $\mathsf{Expt}_b^{\text{blind-batch}}$:

1. The adversary $\mathcal{A}$ takes 1^λ and 1^n as input, and sends a string $\mathbf{s} \in \{0,1\}^n$, and an index $i \in [n]$ to the challenger.
2. The challenger generates $\mathsf{pp} \leftarrow \mathsf{Setup}\left(1^\lambda\right)$ and $h = \mathsf{Gen}\left(\mathsf{pp}, \mathbf{s}\right)$.
3. The challenger randomly generates $\mathbf{m} = (\mathsf{m}_0, \mathsf{m}_1)$ and creates:

$$\mathsf{ct}_1^* \leftarrow \mathsf{Enc}_1\left(\mathsf{pp}, h, (i, \mathsf{m}_0, \mathsf{m}_1)\right), \mathsf{ct}_2^* \leftarrow \mathsf{Enc}_2\left(\mathsf{pp}, h, (i, \mathsf{m}_0, \mathsf{m}_1)\right).$$

   - If $b = 0$, the challenger sets $\mathsf{ct}^* = (\mathsf{ct}_1^*, \mathsf{ct}_2^*)$.
   - If $b = 1$, the challenger sets $\mathsf{ct}^* = (\mathsf{ct}_1^*, \sigma^*)$ where $\sigma^* \leftarrow \{0,1\}^{|\mathsf{ct}_2^*|}$.
4. Finally, the challenger sends $(\mathsf{pp}, \mathsf{ct}^*)$ to the adversary $\mathcal{A}$.

A batch encryption scheme $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is said to be blind if:

1. The encryption subroutine Enc_1 does not depend on either the value h or the message pair $(\mathsf{m}_0, \mathsf{m}_1)$. Hence Enc_1 can be written as $\mathsf{Enc}_1(\mathsf{pp}, h, (i, \mathsf{m}_0, \mathsf{m}_1); r) = \mathsf{Enc}_1(\mathsf{pp}, i; r)$.
2. For all PPT adversaries $\mathcal{A}$, the views of the adversary in $\mathsf{Expt}_0^{\text{blind-batch}}$ and $\mathsf{Expt}_1^{\text{blind-batch}}$ are computationally indistinguishable.

Construction from IHwUF. We present a construction of fully succinct blind batch encryption from any IHwUF family.

- $\mathsf{Setup}(1^\lambda)$: Given the security parameter λ the setup algorithm creates a description $\mathcal{F}_{\text{IHwUF}}$ for an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$. It then fixes some integer $n > \log|\mathcal{X}| + \omega(\log(\lambda))$ and samples $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$. It outputs the public parameter pp as

$$\mathsf{pp} = \left(\mathcal{F}_{\text{IHwUF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}\right).$$

- $\mathsf{Gen}(\mathsf{pp}, \mathrm{s})$: Given $\mathsf{pp} = \left(\mathcal{F}_{\text{IHwUF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}\right)$ and a binary string s, output the corresponding hash value as

$$h = \bigoplus_{j\in[n]} x_{j,s_j}.$$

- $\mathsf{Enc}(\mathsf{pp}, h, (i, \mathsf{m}_0, \mathsf{m}_1))$: Given $\mathsf{pp} = \left(\mathcal{F}_{\text{IHwUF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}\right)$, a hash value h, an index $i \in [n]$, and $(\mathsf{m}_0, \mathsf{m}_1) \in \{0,1\} \times \{0,1\}$, the encryption algorithm samples $k_0, k_1 \leftarrow \mathcal{K}$ and computes the following

$$\begin{aligned}
y_{i,0}^{(0)} &= F(k_0, x_{i,0}), \quad y_{i,1}^{(1)} = F(k_1, x_{i,1}), \quad y_{i,1}^{(0)} = y_{i,0}^{(1)} = \perp.\\
y_{j,b}^{(0)} &= F(k_0, x_{j,b}) \text{ for } j \in [n] \setminus \{i\}, b \in \{0,1\},\\
y_{j,b}^{(1)} &= F(k_1, x_{j,b}) \text{ for } j \in [n] \setminus \{i\}, b \in \{0,1\},\\
\mathsf{e}_0 &= \text{XOR}(\text{HardCore}(F(k_0, h)), \mathsf{m}_0),\\
\mathsf{e}_1 &= \text{XOR}(\text{HardCore}(F(k_1, h)), \mathsf{m}_1).
\end{aligned}$$

It finally outputs the ciphertext

$$\mathsf{ct} = \left(\mathsf{ct}_1 = \{y_{j,b}^{(b')}\}_{j\in[n],(b,b')\in\{0,1\}^2}, \mathsf{ct}_2 = (\mathsf{e}_0, \mathsf{e}_1)\right).$$

- $\mathsf{Dec}\,(\mathsf{pp}, \mathbf{s}, i, \mathsf{ct})$: Given pp, a string $\mathbf{s}$, and a ciphertext ct the decryption algorithm outputs

$$\mathsf{m}'_{s_i} = \mathrm{XOR}\Big(\mathrm{HardCore}\Big(\bigotimes_{j\in[n]} y_{j,s_j}^{(s_i)}\Big), \mathsf{e}_{s_i}\Big),$$

  where $\mathsf{ct} = \Big(\mathsf{ct}_1 = \big\{y_{j,b}^{(b')}\big\}_{j\in[n],(b,b')\in\{0,1\}^2}, \mathsf{ct}_2 = (\mathsf{e}_0, \mathsf{e}_1)\Big)$.

Instantiation from General Protocol. The aforementioned BE scheme can be instantiated from the general protocol described in Section 3.3 as follows:

- **Initialization.** Instantiate the protocol using an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with description $\mathcal{F}_{\text{IHwUF}}$, a fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, $\bar{N} = 2$ and $\hat{N} = 0$. Set $\mathsf{pp} = (\mathcal{F}_{\text{IHwUF}}, \mathrm{X})$, where X is the set of base elements.

- **Pre-Evaluation.** In the general protocol, the pre-evaluation phase samples a uniformly random binary string $\mathbf{s} \in \{0,1\}^n$. One may view this as an input to the generation algorithm. Consequently, if $\mathrm{X}^* = \{x^*\}$ is the output of the post-evaluation phase, set the image $h = x^*$.

- **Evaluation:** Let $\mathrm{Y} = \{y_{j,b}^{(0)}, y_{j,b}^{(1)}\}_{j\in[n],b\in\{0,1\}}$ and $\mathrm{Y}^* = (y_0^*, y_1^*)$ be the tuples output by the evaluation phase. For a given $i \in [n]$, set $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2)$ where

$$\begin{aligned}\mathsf{ct}_1 &= \mathrm{Y} \setminus \big\{y_{i,1}^{(0)}, y_{i,0}^{(1)}\big\},\\ \mathsf{ct}_2 &= \big(\mathsf{e}_0 = \mathrm{HardCore}\,(y_0^*)\,, \mathsf{e}_1 = \mathrm{HardCore}\,(y_1^*)\,\big).\end{aligned}$$

To see that the instantiation satisfies the properties of a blind batch encryption scheme, consider the following:

- Correctness follows from Claim 3.3.1. Specifically, given a binary string $\mathbf{s} \in \{0,1\}^n$ and a ciphertext $\mathsf{ct} = \mathsf{Enc}\,(\mathsf{pp}, h, (i, \mathsf{m}_0, \mathsf{m}_1))$ such that $h = \mathsf{Gen}(\mathsf{pp}, \mathbf{s})$ and $s_i = b^*$, the decryption algorithm does not need $y_{i,1-b^*}^{(b^*)}$ to recover the message m.

- One-wayness follows from Lemma 3.2.1.

- To show the security of the scheme, observe that $\mathsf{m}_{s_i} = \mathsf{m}'_{s_i}$ and hence it is enough to show that for *any* $\mathbf{s} \in \{0,1\}^n$ it holds that

$$\begin{aligned}&\big(\mathbf{s}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{F(k_{1-s_i}, x_{j,b})\}_{j\in[n],b\in\{0,1\}} \setminus \{F(k_{1-s_i}, x_{i,s_i})\}, \mathrm{HardCore}(F(k_{1-s_i}, h))\big) \overset{c}{\approx}\\ &\big(\mathbf{s}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{F(k_{1-s_i}, x_{j,b})\}_{j\in[n],b\in\{0,1\}} \setminus \{F(k_{1-s_i,}, x_{i,s_i})\}, \beta\big),\end{aligned}$$

where $\beta \leftarrow \{0,1\}$ is a uniform bit and $h = \oplus_{j\in[n]} x_{j,s_j}$. This follows from Lemma 3.3.9.

- Blindness follows from the fact that the component ct_1 does not depend on the image h.

We showed that IHwUF implies blind batch encryption. As IHwUF is also enough to construct blind garbled circuits, it follows that any IHwUF implies the above three primitives.

Note 5.4.2. The aforementioned construction has an a priori bounded number of homomorphic operations, which allows it to be instantiated equivalently using a γ-bounded IHwUF if $\gamma \geq n$, with the following minor modification to the encryption algorithm:

$$\begin{aligned} \mathsf{e}_0 &= \text{XOR}\left(\text{HardCore}\left(\mathcal{R}\left(F\left(k_0, h\right)\right)\right), \mathsf{m}_0\right), \\ \mathsf{e}_1 &= \text{XOR}\left(\text{HardCore}\left(\mathcal{R}\left(F\left(k_1, h\right)\right)\right), \mathsf{m}_1\right), \end{aligned}$$

and the following minor modification to the decryption algorithm:

$$\mathsf{m}'_{s_i} = \text{XOR}\Big(\text{HardCore}\Big(\mathcal{R}\Big(\bigotimes_{j\in[n]} y_{j,s_j}^{(s_i)}\Big)\Big), \mathsf{e}_{s_i}\Big),$$

where $\mathcal{R} : \mathcal{Y} \to \mathcal{Z}$ is a universal map (see Definition 3.1.5).

Note 5.4.3. The aforementioned construction can only be instantiated from a (bounded) IHwPRF family for which the input space is independent of the key.

Implications. Brakerski *et al.* [BLSV18] showed that fully succinct blind batch encryption scheme along with blind garbled circuit (which can be constructed from any one-way function) imply:

1. Anonymous IBE,
2. Bounded KDM-secure PKE,
3. Leakage-resilient PKE with resilience to leakage of a $(1 - o(1))$-fraction of the secret key.

5.5 Hinting PRG

In this section, we show that IHwUF implies hinting PRG, which is a stronger variant of traditional PRG. Hinting PRG, which has been introduced by Koppula and Waters in [KW19], can be used to generically transform any CPA-secure ABE into a CCA-secure one.

Informally, a hinting PRG takes n bits as input and outputs $n \cdot L$ output bits for some fixed polynomials $n = n(\lambda)$ and $L = L(\lambda)$, such that no PPT adversary can distinguish between $2n$ uniformly random strings and $2n$ strings such that half the strings are output by the PRG, and the remaining half are uniformly random, *even if* the strings are arranged as a $2 \times n$ matrix as follows: in the i^{th} column of this matrix, the top entry is pseudorandom if the i^{th} bit of the seed is 0; else, the bottom entry is pseudorandom. Note that such a matrix-based arrangement carries some information (or "hint") about the seed, and the indistinguishability guarantee in the presence of such an arrangement is what makes a hinting PRG stronger than a traditional PRG.

Below, we provide the formal definition of Hinting PRG from [KW19]. The formal definition is slightly different from the informal description in terms of output length. Specifically, the formal definitions states that the PRG outputs $(n+1) \cdot L$ bits, where the first L bits of the output do not contain any hint about the seed.

Definition 5.5.1. (Hinting PRG.) A hinting PRG is a pair of PPT algorithms $(\mathsf{Setup}, \mathsf{Eval})$ defined as follows:

- $\mathsf{Setup}(1^\lambda)$: Given the security parameter λ, it sets $n = n(\lambda)$ and $L = L(\lambda)$ for some fixed polynomial functions and outputs (pp, n, L), where pp is the public parameter.
- $\mathsf{Eval}(\mathsf{pp}, \mathbf{s}, i^*)$: Given the public parameter pp, a seed $\mathbf{s} \in \{0,1\}^n$, and an index $i^* \in \{0\} \cup [n]$, it outputs a string $\mathbf{e} \in \{0,1\}^L$.

Security. For $b \in \{0,1\}$, define the experiment $\mathsf{Expt}_b^{\mathsf{HPRG}}$ between a challenger and an adversary $\mathcal{A}$ as follows:

Experiment $\mathsf{Expt}_b^{\mathsf{HPRG}}$:

1. The challenger generates $(\mathsf{pp}, n, L) \leftarrow \mathsf{Setup}(1^\lambda)$ and provides it to the adversary $\mathcal{A}$.
2. The challenger uniformly samples $\mathbf{s} = (s_1, \ldots, s_n) \leftarrow \{0,1\}^n$ and sets the following:

$$\begin{aligned} \bar{\mathbf{t}}^{(0)} &= \mathsf{Eval}(\mathsf{pp}, \mathbf{s}, 0), \quad \bar{\mathbf{t}}^{(1)} \leftarrow \{0,1\}^L, \\ \mathbf{t}^{(0)}_{i,s_i} &= \mathsf{Eval}(\mathsf{pp}, \mathbf{s}, i), \quad \mathbf{t}^{(0)}_{i,1-s_i} \leftarrow \{0,1\}^L \text{ for each } i \in [n], \\ \mathbf{t}^{(1)}_{i,b'} &\leftarrow \{0,1\}^L \text{ for each } i \in [n], b' \in \{0,1\}, \end{aligned}$$

and sends $\{\bar{\mathbf{t}}^{(b)}, \mathbf{t}^{(b)}_{j,b'}\}_{j \in [n], b' \in \{0,1\}}$ to the adversary $\mathcal{A}$.

A hinting PRG is secure if for all PPT adversaries $\mathcal{A}$, the views of the adversary in $\mathsf{Expt}_0^{\mathsf{HPRG}}$ and $\mathsf{Expt}_1^{\mathsf{HPRG}}$ are computationally indistinguishable.

Construction from IHwUF. We show a black-box construction of hinting PRG from any IHwUF family.

- $\mathsf{Setup}\left(1^\lambda\right)$: Given the security parameter λ the setup algorithm creates a description $\mathcal{F}_{\text{IHwUF}}$ for an IHwUF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, and fixes $n > \log|\mathcal{X}| + \omega(\log(\lambda))$ and $L = L(\lambda)$. It samples $2n$ group elements from $\mathcal{X}$ and $(2n+1) \cdot L$ keys from $\mathcal{K}$ as

$$\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}, \quad \left\{\bar{k}_\ell \leftarrow \mathcal{K}\right\}_{\ell\in[L]}, \quad \{k_{i,\ell,\beta} \leftarrow \mathcal{K}\}_{(i,\ell,\beta)\in[n]\times[L]\times\{0,1\}}.$$

  For each $\ell \in [L]$, it creates an $n \times 2$ matrix $\bar{\mathbf{Y}}^{(\ell)} \in \mathcal{Y}^{n\times 2}$ such that for each $(j,b) \in [n]\times\{0,1\}$ we have

$$\bar{\mathbf{Y}}^{(\ell)}_{j,b} = F(\bar{k}_\ell, x_{j,b}).$$

  In addition, for each $(i,\ell,\beta) \in [n] \times [L] \times \{0,1\}$, it creates an $n \times 2$ matrix $\mathbf{Y}^{(i,\ell,\beta)} \in \mathcal{Y}^{n\times 2}$ such that for each $(j,b) \in [n] \times \{0,1\}$, we have

$$\mathbf{Y}^{(i,\ell,\beta)}_{j,b} = \begin{cases} \perp & \text{if } (i,\beta) = (j, 1-b), \\ F\left(k_{i,\ell,\beta}, x_{j,b}\right) & \text{otherwise.} \end{cases}$$

  Finally, it outputs (pp, n, L) where[1]

$$\mathsf{pp} = \left(\mathcal{F}_{\text{IHwUF}}, \left\{\bar{\mathbf{Y}}^{(\ell)}\right\}_{\ell\in[L]}, \left\{\mathbf{Y}^{(i,\ell,\beta)}\right\}_{(i,\ell,\beta)\in[n]\times[L]\times\{0,1\}}\right).$$

- $\mathsf{Eval}(\mathsf{pp}, \mathbf{s}, i^*)$: The evaluation algorithm outputs $(t_1, \ldots, t_L)$ where for each $\ell \in [L]$ we have

$$t_\ell = \begin{cases} \text{HardCore}\left(\bigotimes_{j\in[n]} \bar{\mathbf{Y}}^{(\ell)}_{j,s_j}\right) & \text{if } i^* = 0, \\ \text{HardCore}\left(\bigotimes_{j\in[n]} \mathbf{Y}^{(i^*,\ell,s_{i^*})}_{j,s_j}\right) & \text{otherwise.} \end{cases}$$

Instantiation from the general protocol. The aforementioned HPRG can be instantiated from the general protocol described in Section 3.3 as follows:

[1] In [KW19], Koppula and Waters explicitly include the randomness for generating hardcore bits in pp. Here, we implicitly assume that every element in the output group of the IHwUF has a deterministic hardcore bit. If this is not the case, the randomness for hardcore bit generation should be included in pp.

- **Initialization.** Instantiate the protocol using an IHwUF F with description $\mathcal{F}_{\text{IHwUF}}$, a fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, $N^* = 0$, $\bar{N} = (2n+1) \cdot L$, and $\hat{N} = 1$. Set $\mathsf{pp}_1 = (\mathcal{F}_{\text{IHwUF}}, n, L)$.

- **Evaluation:** Let Y be the output of the evaluation phase where

$$\mathrm{Y} = \left(\left\{\bar{y}_{j,b}^{(\ell)}\right\}_{(\ell,j,b)\in[L]\times[n]\times\{0,1\}}, \left\{y_{j,b}^{(i,\ell,\beta)}\right\}_{(i,\ell,\beta,j,b)\in[n]\times[L]\times\{0,1\}\times[n]\times\{0,1\}}\right).$$

  For each $\ell \in [L]$, create an $n \times 2$ matrix $\bar{\mathbf{Y}}^{(\ell)} \in \mathcal{Y}^{n\times 2}$ such that for each $(j,b) \in [n] \times \{0,1\}$ we have

$$\bar{\mathbf{Y}}_{j,b}^{(\ell)} = \bar{y}_{j,b}^{(\ell)},$$

  and set

$$\mathsf{pp}_2 = \left\{\bar{\mathbf{Y}}_{j,b}^{(\ell)}\right\}_{(\ell,j,b)\in[L]\times n\times\{0,1\}}.$$

  In addition, for each $(i,\ell,\beta) \in [n] \times [L] \times \{0,1\}$ create an $n \times 2$ matrix $\mathbf{Y}^{(i,\ell,\beta)} \in \mathcal{Y}^{n\times 2}$ such that for each $(j,b) \in [n] \times \{0,1\}$ we have

$$\mathbf{Y}_{j,b}^{(i,\ell,\beta)} = \begin{cases} \perp & \text{if } (i,\beta) = (j, 1-b), \\ y_{j,b}^{(i,\ell,\beta)} & \text{otherwise.} \end{cases}$$

  Set pp as $\mathsf{pp} = (\mathsf{pp}_1, \mathsf{pp}_2, \mathsf{pp}_3)$ where $\mathsf{pp}_3 = \left\{\mathbf{Y}^{(i,\ell,\beta)}\right\}_{(i,\ell,\beta)\in[n]\times[L]\times\{0,1\}}$.

- **Post-Evaluation:** Recall that in the general protocol, the post-evaluation phase samples $\hat{n} = 1$ uniform string $\mathbf{s} \leftarrow \{0,1\}^N$. One may view this as the input string to the evaluation algorithm of the HPRG (along with some index $i^* \in \{0\} \cup [n]$). Let $\hat{\mathrm{Y}}$ be the output of the post-evaluation phase where

$$\hat{\mathrm{Y}} = \left(\left\{\hat{y}^{(\ell)}\right\}_{\ell\in[L]}, \left\{\hat{y}^{(i,\ell,\beta)}\right\}_{(i,\ell,\beta)\in[n]\times[L]\times\{0,1\}}\right).$$

  Let $\bar{\mathbf{t}} \in \{0,1\}^L$ be a string whose ℓ^{th} component is

$$\bar{t}_\ell = \text{HardCore}\left(\hat{y}^{(\ell)}\right).$$

  In addition, for each $i \in [n]$ let $\mathbf{t}_i \in \{0,1\}^L$ be a string whose ℓ^{th} component is

$$t_{i,\ell} = \text{HardCore}\left(\hat{y}^{(i,\ell,s_i)}\right).$$

  Set the output of the hinting PRG as $(\bar{\mathbf{t}}, \{\mathbf{t}_i\}_{i\in[n]})$.

Consider the following hybrids:

- $\mathcal{H}_0$: This is the real experiment, so the challenger generates (pp, n, L) and sends evaluations $\{\bar{\mathbf{t}}^{(b)}, \mathbf{t}^{(b)}_{j,b'}\}_{j\in[n],b'\in\{0,1\}}$ to the adversary.

- $\mathcal{H}_1$: This is similar to the real experiment, except that the challenger generates $\mathbf{t}^{(1)}_{i,b'}$ in the following way: for each $\ell \in L$, the ℓ^{th} bit of $\mathbf{t}^{(1)}_{i,b'}$ is equal to

$$\bigotimes_{j\in[n]} F\big(k_{i,\ell,1-s_i}, x_{j,s_j}\big) = F\big(k_{i,\ell,1-s_i}, \bigoplus_{j\in[n]} x_{j,s_j}\big).$$

- $\mathcal{H}_2$: This is similar to the previous hybrid, except that the challenger uses a uniform $u \leftarrow \mathcal{X}$ instead of $\bigoplus_{j\in[n]} x_{j,s_j}$ to generate $\bar{\mathbf{t}}^{(0)}$, $\mathbf{t}^{(0)}_{i,b'}$, and $\mathbf{t}^{(1)}_{i,b'}$. So the ℓ^{th} bit of $\bar{\mathbf{t}}^{(0)}$, $\mathbf{t}^{(0)}_{i,b'}$, and $\mathbf{t}^{(1)}_{i,b'}$ is equal to

$$F(\bar{k}_\ell, u), \quad F(k_{i,\ell,s_i}, u), \quad F(k_{i,\ell,1-s_i}, u),$$

  respectively.

- $\mathcal{H}_3$: This hybrid is the ideal experiment where all strings are generated uniformly.

By Corollary 3.3.10 the hybrids $\mathcal{H}_0$ and $\mathcal{H}_1$ are computationally indistinguishable. $\mathcal{H}_1$ and $\mathcal{H}_2$ are statistically indistinguishable by the leftover hash lemma. Computational indistinguishability of $\mathcal{H}_2$ and $\mathcal{H}_3$ follows from Theorem 3.3.8. It follows that $\mathcal{H}_0$ and $\mathcal{H}_3$ are computationally indistinguishable, as required.

Note 5.5.2. The aforementioned construction has an a priori bounded number of homomorphic operations, which allows it to be instantiated equivalently using a γ-bounded IHwUF if $\gamma \geq n$, with the following minor modification to the evaluation algorithm:

$$\mathsf{e}_\ell = \text{HardCore}\Big(\mathcal{R}\Big(\bigotimes_{j\in[n]} \mathbf{Y}^{(i,\ell,s_i)}_{j,s_j}\Big)\Big).$$

Note 5.5.3. The aforementioned construction can only be instantiated from a (bounded) IHwPRF family for which the input space is independent of the key.

CHAPTER 6

Primitives from IHwPRF

We present constructions of various (asymmetric) cryptographic primitives from (bounded) IHwPRF. We also show how to instantiate these constructions using the general protocol of Chapter 3.

6.1 Private Information Retrieval

A (single-database) private information retrieval (PIR) scheme is a two-party protocol between a sender and a receiver. The sender holds a public database (say, for concreteness, a string $\mathbf{s} = (s_1, \ldots, s_n) \in \{0,1\}^n$), and the receiver wishes to query an item in the database (say, the bit s_i for some $i \in [n]$) without revealing which item was queried (that is, i is not revealed to the sender). Note that in this model, the database is public, which implies that the unqueried items/bits need not be hidden from the receiver. A trivial solution is where the sender sends $\mathbf{s}$ to the receiver in the clear, which of course preserves receiver privacy. The total communication in such a protocol, measured as the number of bits exchanged between the sender and the receiver, is n. A *non-trivial* PIR protocol is one that securely achieves the aforementioned functionality with communication strictly smaller than n bits, where n is the size of the database. Black-box constructions of PIR protocols are known from different assumptions, e.g., group-homomorphic encryption [KO97], smooth subgroup assumptions [CMS99, GR05], and trapdoor permutations [KO00].

As a warm-up, we first demonstrate an inefficient PIR protocol that has a communication overhead of $O(n \cdot \ell(\lambda))$ bits, where ℓ is the maximum number of bits needed to encode a group element in either $\mathcal{X}$ or $\mathcal{Y}$. While this is even worse than the trivial protocol, we subsequently show how the efficiency of this protocol may be boosted to achieve a non-trivial PIR protocol, without any additional assumptions.

Inefficient PIR from IHwPRF. Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an IHwPRF:

1. On input an index $i \in [n]$, the receiver samples 2 elements $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ and a key $k \leftarrow \mathcal{K}$. Let $\tilde{y}$ be a non-identity element of $\mathcal{Y}$. It then sets the following

$$\begin{aligned} y_{i,0} &= F(k, x_{i,0}) \\ y_{i,1} &= F(k, x_{i,1}) \otimes \tilde{y} \\ y_{j,b} &= F(k, x_{j,b}) \text{ for } j \in [n] \setminus \{i\}, b \in \{0,1\} \end{aligned}$$

and sends $\{x_{j,b}, y_{j,b}\}_{j\in[n],b\in\{0,1\}}$ to the sender.

2. The sender, on input a string $\mathbf{s} = (s_1, \ldots, s_n) \in \{0,1\}^n$ and $\{x_{j,b}, y_{j,b}\}_{j\in[n],b\in\{0,1\}}$, sends (x^*, y^*) to the receiver where

$$(x^*, y^*) = \Big(\bigoplus_{j\in[n]} x_{j,s_j}, \bigotimes_{j\in[n]} y_{j,s_j} \Big).$$

3. The receiver retrieves the bit s_i as

$$s_i = \begin{cases} 0 & \text{if } y^* = F(k, x^*), \\ 1 & \text{otherwise.} \end{cases}$$

Boosting efficiency. We now apply a generic efficiency-boosting technique introduced in [KO97] to convert the inefficient protocol into a PIR protocol with a communication overhead of $O(\sqrt{n}\cdot\ell(\lambda))$ bits. Quite evidently, such a PIR protocol is non-trivial in the sense that the overall communication complexity is strictly smaller than n bits for sufficiently large n. The idea is to view the database string $\mathbf{s} = (s_1, \ldots, s_n) \in \{0,1\}^n$ as a binary matrix $\mathbf{S} \in \{0,1\}^{\sqrt{n}\times\sqrt{n}}$ such that:[1]

$$\mathbf{S}_{j_1,j_2} = s_{(j_1-1)\sqrt{n}+j_2} \text{ for } j_1, j_2 \in [\sqrt{n}]$$

The receiver sends across only $4\sqrt{n}$ group elements in its first message to the sender, as opposed to $4n$ in the inefficient protocol, while the receiver performs $2\sqrt{n}$ subset-sum/subset-product operations over these elements (one per column of the matrix $\mathbf{S}$) and sends back $2\sqrt{n}$ group elements to the receiver. The detailed protocol is as follows:

1. On input an index $i \in [n]$, the receiver uniformly samples $2\sqrt{n}$ elements from $\mathcal{X}$ as $\{x_{j_1,b} \leftarrow \mathcal{X}\}_{j_1\in[\sqrt{n}],b\in\{0,1\}}$ and $k \leftarrow \mathcal{K}$. Let $\tilde{y}$ be a non-identity element of $\mathcal{Y}$, and let

[1] We assume that n is an integer for the sake of simplicity.

$i_1 = \lceil i/\sqrt{n} \rceil$. The receiver sets the following

$$\begin{aligned} y_{i_1,0} &= F\left(k, x_{i_1,0}\right), \\ y_{i_1,1} &= F\left(k, x_{i_1,1} \oplus \tilde{x}\right), \\ y_{j_1,b} &= F\left(k, x_{j_1,b}\right) \text{ for } j_1 \in [\sqrt{n}] \setminus \{i_1\}, b \in \{0,1\}, \end{aligned}$$

and sends $\{x_{j_1,b}, y_{j_1,b}\}_{j_1 \in [\sqrt{n}], b \in \{0,1\}}$ to the sender.

2. The sender, on input a string $\mathbf{s} = (s_1, \ldots, s_n) \in \{0,1\}^n$ and $\{x_{j_1,b}, y_{j_1,b}\}_{j_1 \in [\sqrt{n}], b \in \{0,1\}}$, creates a binary matrix $\mathbf{S} \in \{0,1\}^{\sqrt{n} \times \sqrt{n}}$ where

$$\mathbf{S}_{j_1,j_2} = s_{\sqrt{n}(j_1-1)+j_2} \text{ for } j_1, j_2 \in [\sqrt{n}].$$

It then sends $\{x^*_{j_2}, y^*_{j_2}\}_{j_2 \in [\sqrt{n}]}$ to the receiver where

$$\begin{aligned} x^*_{j_2} &= \bigoplus_{j_1 \in [\sqrt{n}]} x_{j_1, \mathbf{s}_{j_1,j_2}} \text{ for } j_2 \in [\sqrt{n}], \\ y^*_{j_2} &= \bigotimes_{j_1 \in [\sqrt{n}]} y_{j_1, \mathbf{s}_{j_1,j_2}} \text{ for } j_2 \in [\sqrt{n}]. \end{aligned}$$

3. The receiver computes $i_2 = i \bmod \sqrt{n}$ and retrieves the bit s_i as

$$s_i = \begin{cases} 0 & \text{if } y^*_{i_2} = F\left(k, x^*_{i_2}\right), \\ 1 & \text{otherwise.} \end{cases}$$

Instantiation from the general protocol. Let $|\mathrm{DB}|$ denote the size of the database string in the PIR scheme. This scheme can be instantiated from the general protocol described in Section 3.3 as follows:

- **Initialization.** Instantiate the protocol using an IHwPRF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, a fixed $n = \sqrt{|\mathrm{DB}|}$ such that $n > \log |\mathcal{X}| + \omega(\log(\lambda))$, $N^* = 0$, $\bar{N} = 1$ and $\hat{N} = n$.
- **Evaluation:** Let $\mathrm{Y} = \{y_{j,b}\}_{j \in [n], b \in \{0,1\}}$ be the output of the evaluation phase. For a given $i \in [|\mathrm{DB}|]$, set $i_1 = \lceil i/n \rceil$ and reset $y_{i_1,1} := y_{i_1,1} \otimes \tilde{y}$, where $\tilde{y}$ is a non-identity element in $\mathcal{Y}$. Set the first message of the receiver to the sender as (X, Y).
- **Post-Evaluation:** Recall that in the protocol, the post-evaluation phase samples $\hat{N} = n$ binary strings, each of size n. One may view this as a random binary matrix of size $n \times n$. If

$\hat{\mathrm{X}}$ and $\hat{\mathrm{Y}}$ are the outputs of the post-evaluation phase, set the message from the sender to the receiver as $(\hat{\mathrm{X}}, \hat{\mathrm{Y}})$.

Correctness and security of the scheme follow from Claim 3.3.1 and Theorem 3.3.3, respectively.

Note 6.1.1. The aforementioned construction has an a priori bounded number of homomorphic operations, which allows it to be instantiated equivalently using a γ-bounded IHwPRF if $\gamma \geq n$, with the following minor modification to the final step:

$$s_i = \begin{cases} 0 & \text{if } \mathcal{R}\left(y^*_{i_2}\right) = \mathcal{R}\left(F\left(k, x^*_{i_2}\right)\right), \\ 1 & \text{otherwise.} \end{cases}$$

where $\mathcal{R} : \mathcal{Y} \to \mathcal{Z}$ is a universal map (see Definition 3.1.6).

Note 6.1.2. The aforementioned construction can also be instantiated using a (γ-bounded) IHwPRF family for which the input space depends on the key. In particular, since the receiver chooses the PRF key, it can set up the input space accordingly, and sample a random $2n$-vector of elements from this space. This in turn allows instantiating the PIR scheme from all concrete assumptions that give rise to (γ-bounded) IHwPRFs, including the ones with key-dependent input space (see Section 3.4).

6.2 Lossy Trapdoor Functions

We begin with the formal definition of a lossy trapdoor function family from [PW08], and then we show how to construct lossy TDF family from IHwPRFs.

Definition 6.2.1. (Lossy Trapdoor Function.) A lossy trapdoor function family is a tuple of four PPT algorithms $\mathsf{LTDF} = (\mathsf{GenInjective}, \mathsf{GenLossy}, \mathsf{Eval}, \mathsf{Invert})$ defined as follows:

- $\mathsf{GenInjective}\left(1^\lambda\right)$: Given the security parameter λ, the algorithm outputs the public parameter pp for an *injective* function, along with a trapdoor t.
- $\mathsf{GenLossy}\left(1^\lambda\right)$: Given the security parameter λ, the algorithm outputs the public parameter pp for a *lossy* function. It does not produce a trapdoor. (See below for a formal definition of lossiness.)
- $\mathsf{Eval}\left(\mathsf{pp}, \mathrm{s}\right)$: Given the public parameter pp and a preimage string $\mathrm{s} \in \{0,1\}^n$ (where $n = n(\lambda)$ is included in pp), the evaluation algorithm outputs the corresponding image h.

- $\mathsf{Invert}\,(\mathsf{t}, h)$: Given the trapdoor t and an image h, the inversion algorithm outputs $\mathbf{s}' \in \{0,1\}^n$.

The following completeness and security properties must be satisfied:

- **Completeness:** If $(\mathsf{pp}, \mathsf{t}) \leftarrow \mathsf{GenInjective}(1^\lambda)$, then for all preimage strings $\mathbf{s} \in \{0,1\}^n$, it holds with overwhelming probability over the random coins of $\mathsf{GenInjective}$ that

$$\mathsf{Invert}\,(\mathsf{t}, h = \mathsf{Eval}\,(\mathsf{pp}, \mathbf{s})) = \mathbf{s}.$$

- **One-wayness without trapdoor:** For any PPT adversary $\mathcal{A}$ we have

$$\Pr[\mathsf{Eval}(\mathsf{pp}, \mathcal{A}(h)) = h] \leq \mathrm{negl}(\lambda),$$

  where $\mathsf{pp} \leftarrow \mathsf{GenInjective}(1^\lambda)$, $\mathbf{s} \leftarrow \{0,1\}^n$ and $h = \mathsf{Eval}(\mathsf{pp}, \mathbf{s})$.

- **Lossiness:** A TDF family $(\mathsf{GenInjective}, \mathsf{GenLossy}, \mathsf{Eval}, \mathsf{Invert})$ is said to be ε-lossy if for any *unbounded* adversary $\mathcal{A}$ we have $\Pr[\mathcal{A}(h) = \mathbf{s}] \leq \varepsilon$ where $\mathsf{pp} \leftarrow \mathsf{GenLossy}(1^\lambda)$, $\mathbf{s} \leftarrow \{0,1\}^n$ and $h = \mathsf{Eval}(\mathsf{pp}, \mathbf{s})$.

- **Indistinguishability of modes:** For any PPT adversary $\mathcal{A}$ we have

$$\left| \Pr[\mathcal{A}(\mathsf{pp}_0) = 1] - \Pr[\mathcal{A}(\mathsf{pp}_1) = 1] \right| \leq \mathrm{negl}(\lambda),$$

  where $\mathsf{pp}_0 \leftarrow \mathsf{GenInjective}(1^\lambda)$ and $\mathsf{pp}_1 \leftarrow \mathsf{GenLossy}(1^\lambda)$.

Construction from IHwPRF. We present a black-box construction of a lossy TDF family from any IHwPRF family. The construction is inspired by the DDH-based lossy TDF family proposed by Peikert and Waters in [PW08]. First, create a description $\mathcal{F}_{\text{IHwPRF}}$ for an IHwPRF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, and fix some integer n such that $n > \log |\mathcal{X}| + \omega(\log(\lambda))$.

- $\mathsf{GenInjective}(1^\lambda)$: In the injective mode, the algorithm samples $2n$ uniform elements from $\mathcal{X}$ as $\{x_{j,b} \leftarrow \mathcal{X}\}_{j \in [n], b \in \{0,1\}}$. It also samples n uniform keys as $\{k_i \leftarrow \mathcal{K}\}_{i \in [n]}$. Let $\tilde{y}$ be a non-identity element of $\mathcal{Y}$. The algorithm sets the following:

$$\begin{aligned} y_{i,0}^{(i)} &= F\,(k_i, x_{i,0}) \text{ for } i \in [n], \\ y_{i,1}^{(i)} &= F\,(k_i, x_{i,1}) \otimes \tilde{y} \text{ for } i \in [n], \\ y_{j,b}^{(i)} &= F\,(k_i, x_{j,b}) \text{ for } (i,j) \in [n] \times [n], i \neq j, b \in \{0,1\}\,. \end{aligned}$$

It outputs the public parameter pp and the trapdoor t as

$$\begin{aligned}\mathsf{pp} &= \left(\mathcal{F}_{\text{IHwPRF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{y^{(i)}_{j,b}\}_{(i,j)\in[n]\times[n],b\in\{0,1\}}\right),\\ \mathsf{t} &= \{k_i\}_{i\in[n]}.\end{aligned}$$

- $\mathsf{GenLossy}(1^\lambda)$: The algorithm samples $2n$ uniform elements from $\mathcal{X}$ as $\{x_{j,b} \leftarrow \mathcal{X}\}_{j\in[n],b\in\{0,1\}}$ and n uniform keys as $\{k_i \leftarrow \mathcal{K}\}_{i\in[n]}$, and sets the following

$$y^{(i)}_{j,b} = F(k_i, x_{j,b}) \text{ for } (i,j) \in [n]\times[n], b \in \{0,1\}.$$

  It outputs the public parameter pp as

$$\mathsf{pp} = \left(\mathcal{F}_{\text{IHwPRF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{y^{(i)}_{j,b}\}_{(i,j)\in[n]\times[n],b\in\{0,1\}}\right).$$

- $\mathsf{Eval}(\mathsf{pp}, \mathbf{s})$: Given $\mathsf{pp} = (\mathcal{F}_{\text{IHwPRF}}, \{x_{j,b}\}_{j\in[n],b\in\{0,1\}}, \{y^{(b)}_{i,j}\}_{i,j\in[n],b\in\{0,1\}})$ and $\mathbf{s} \in \{0,1\}^n$, the evaluation algorithm computes

$$\begin{aligned}x^* &= \bigoplus_{j\in[n]} x_{j,s_j},\\ y^*_i &= \bigotimes_{j\in[n]} y^{(i)}_{j,s_j} \text{ for } i \in [n].\end{aligned}$$

  It outputs the tuple $(x^*, \{y^*_i\}_{i\in[n]})$.

- $\mathsf{Invert}(\mathsf{t}, h)$: Given the trapdoor $\mathsf{t} = \{k_i\}_{i\in[n]}$ and an image $h = (x^*, \{y^*_i\}_{i\in[n]})$, the inversion algorithm recovers the preimage bit s_i for each $i \in [n]$ as

$$s_i = \begin{cases} 0 & \text{if } y^*_i = F(k_i, x^*),\\ 1 & \text{otherwise.}\end{cases}$$

  Finally, it outputs the recovered string $\mathbf{s} = (s_1, \ldots, s_n)$.

Instantiation from the general protocol. The lossy trapdoor function family described above can be instantiated from the general protocol described in Section 3.3 as follows:

- **Initialization.** Instantiate the protocol using an IHwPRF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with description $\mathcal{F}_{\text{IHwPRF}}$, a fixed n such that $n > \log|\mathcal{X}| + \omega(\log(\lambda))$, $\bar{N} = n$ and $\hat{N} = 1$.

- **Evaluation:** Let $\mathrm{Y} = \{y_{j,b}^{(i)}\}_{(i,j)\in[n]\times[n],b\in\{0,1\}}$ be the output of the evaluation phase.
  - In the lossy mode, output $\mathsf{pp} = (\mathcal{F}_{\text{IHwPRF}}, \mathrm{X}, \mathrm{Y})$.
  - In the injective mode, reset $y_{i,1}^{(i)} := y_{i,1}^{(i)} \otimes \tilde{y}$ for each $i \in [n]$ and output the public parameter as $\mathsf{pp} = (\mathcal{F}_{\text{IHwPRF}}, \mathrm{X}, \mathrm{Y})$.
- **Post-Evaluation:** Recall that in the protocol, the post-evaluation phase uniformly samples a binary string $\mathbf{s} \in \{0,1\}^n$. One may view this as an input to the lossy TDF. If $\hat{\mathrm{X}}$ and $\hat{\mathrm{Y}}$ are the outputs of the post-evaluation phase, set the evaluation output of the lossy TDF as $(\hat{\mathrm{X}}, \hat{\mathrm{Y}})$.

To see that the instantiation satisfies the properties of a lossy TDF family, consider the following:

- Completeness in the injective mode follows from Claim 3.3.1.
- One-wayness without trapdoors follows from Lemma 3.2.1.
- Indistinguishability of modes follows from Theorem 3.3.3.
- To argue lossiness, let $h = (x^*, \{y_i^*\}_{i\in[n]})$ be the output of the evaluation algorithm on input $\mathbf{s} \in \{0,1\}^n$ in the lossy mode. Observe that the number of possible outputs in the lossy mode is upper bounded by $|\mathcal{X}|$, while the number of possible inputs is 2^n for some fixed $n > \log|\mathcal{X}| + \omega(\log(\lambda))$. Hence, the lossiness of the TDF family is $n - \log|\mathcal{X}|$. (The lossiness can be made arbitrarily large by choosing a large enough n.)

Note 6.2.2. The aforementioned construction has an a priori bounded number of homomorphic operations, which allows it to be instantiated equivalently using a γ-bounded IHwPRF if $\gamma \geq n$, with the following minor modification to the inversion algorithm:

$$s_i = \begin{cases} 0 & \text{if } \mathcal{R}\left(y_i^*\right) = \mathcal{R}\left(F\left(k_i, x^*\right)\right), \\ 1 & \text{otherwise.} \end{cases}$$

Note 6.2.3. The aforementioned construction can only be instantiated from a (bounded) IHwPRF family for which the input space is independent of the key.

Implications. The following are some of the implications of a lossy TDF..

- **CCA-secure PKE.** Peikert and Waters [PW08] showed that CCA2-secure PKE can be constructed from any lossy TDF family. Their construction uses a closely related primitive called All-But-One TDF (which can be built from any lossy TDF family). The decryption algorithm in the resulting PKE is witness recovering.

- **Selective Opening Attack (SOA)-secure PKE.** Bellare *et al.* [BHY09] showed that PKE schemes that are secure against selective opening attacks can be constructed from any lossy TDF family.[1] It follows that (bounded) IHwPRFs are sufficient to construct SOA-secure PKE.

6.3 Oblivious Transfer and Multi-Party Computation

We show how to construct *round-optimal* maliciously secure OT[2] and MPC in the plain model from IHwPRF. We use a recent result of Friolo *et al.* [FMV19] in which they showed that:[3]

1. A CPA-secure PKE with pseudorandom public keys over a group (i.e., the distribution of public keys are computationally indistinguishable from the uniform distribution over an efficiently samplable group) implies (in a black-box way) a two-round *strongly uniform* key-exchange protocol, where the distribution of messages sent by one of the parties is computationally indistinguishable from the uniform distribution over an efficiently samplable group, even when the other party is malicious.

2. For any $t \geq 2$, a t-round strongly uniform secure key-exchange protocol is black-box equivalent to a t-round strongly uniform semi-honestly secure OT protocol in the plain model, where the distribution of all the messages sent by the receiver are computationally indistinguishable from the uniform distribution over an efficiently samplable group, even when the sender is malicious.

3. For any odd $t \geq 3$, a t-round strongly uniform semi-honestly secure OT protocol in the plain model together with a noninteractive statistically binding commitment scheme, implies (in a black-box manner) a $(t+1)$-round maliciously secure OT protocol in the plain model.

In addition, a recent result of Choudhuri *et al.* [CCG$^+$19] showed that 4-round maliciously secure OT protocol implies a 4-round maliciously secure MPC protocol.

Construction from IHwPRF. We now demonstrate that the framework of Friolo *et al.* [FMV19] can be instantiated using any IHwPRF family. We use an IHwPRF (instead of an IHwUF) in the construction of Section 5.2 to get a PKE with pseudorandom public keys. As a result, pk and sk for

[1] See [BHY09] for the details of the construction.
[2] Two-round semi-honest OT in the plain model is implied by lossy TDF [BL18], which can be based on IHwPRF.
[3] For the sake of succinctness, we state the results informally. See [FMV19] for the formal description.

the modified scheme will have the form

$$\mathsf{sk} = k, \quad \mathsf{pk} = \{(x_{j,b}, y_{j,b} = F(k, x_{j,b}))\}_{j\in[n],b\in\{0,1\}},$$

where $y_{j,b}$ are the evaluations of the IHwPRF. It is easy to see that the modified scheme satisfies CPA security. Moreover, any PPT algorithm that distinguishes the public key in the modified scheme from a uniform tuple breaks the weak pseudorandomness of F. Finally, the aforementioned PKE constructions immediately implies a noninteractive statistically binding commitment scheme.

By combining the observations above with the recent results of Friolo *et al.* [FMV19] and Choudhuri *et al.* [CCG+19], it follows that

- Any (bounded) IHwPRF implies a 4-round maliciously secure OT in the plain model.
- Any (bounded) IHwPRF implies a 4-round maliciously secure MPC in the plain model.

CHAPTER 7

Primitives from KHwPRF

In this section we show how to construct PKE and input-homomorphic weak PRF (IHwPRF) from a key-homomorphic weak PRF (KHwPRF). First, we introduce a hardness assumption over the *output group* of a KHwPRF. This hardness assumption has the advantage that it does not directly involve the input set $\mathcal{X}$ of the KHwPRF, which may be algebraically unstructured. Here (and in the following two sections), we assume that the KHwPRF has unbounded (or exact) homomorphism. Later in this chapter, we show how to extend our results to bounded KHwPRF. Finally, we provide a construction of Naor-Reingold style PRF from an exact KHwPRF.

7.1 On the Output Group of a KHwPRF

We formally state a hardness assumption which can be based on the security of a KHwPRF. As before, we denote the security parameter by λ.

Theorem 7.1.1. *Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a KHwPRF, and let $m = \mathrm{poly}(\lambda)$ be an (arbitrary) positive integer. Assume that $d = \mathrm{poly}(\lambda)$ be a positive integer such that $d > \log|\mathcal{K}| + \omega(\log(\lambda))$. Let $\mathbf{Y} \in \mathcal{Y}^{m\times d}$ be a matrix of group elements such that each entry $y_{i,j}$ (for $i \in [m], j \in [d]$) is drawn uniformly and independently from $\mathcal{Y}$. If $\mathbf{s} \leftarrow \{0,1\}^d$, then for any PPT adversary we have*

$$(\mathbf{Y}, \mathbf{Ys}) \overset{c}{\approx} (\mathbf{Y}, \mathbf{u}),$$

where $\mathbf{u} \leftarrow \mathcal{Y}^m$ is a a vector of m uniformly chosen elements from $\mathcal{Y}$.

Proof. Let $\mathbf{F} \in \mathcal{Y}^{m\times d}$ be a matrix formed in the following way: first sample m uniform elements from $\mathcal{X}$ as $\{x_i \leftarrow \mathcal{X}\}_{i\in[m]}$, and generate d uniform elements from $\mathcal{K}$ as $\{k_j \leftarrow \mathcal{K}\}_{j\in[d]}$. Now we set $\mathbf{F}_{i,j} = F(k_j, x_i)$, i.e., each row (respectively, column) has the same input (respectively, key).

In the first part we prove that $\mathbf{F} \overset{c}{\approx} \mathbf{Y}$. We define the hybrids $\mathcal{H}_j$ over the columns as follows: let $\mathcal{H}_j$ be the hybrid that the first j columns are generated using the weak PRF and the remaining columns are generated using uniform and independent values. By construction, we have $\mathcal{H}_0 \equiv \mathbf{Y}$ and $\mathcal{H}_d \equiv \mathbf{F}$. It is enough to show that $\mathcal{H}_{j-1} \overset{c}{\approx} \mathcal{H}_j$ for each $j \in [d]$. Given access to an oracle $\mathcal{O}$ which is either F or a truly random function, the reduction invokes its oracle m times and receives $\{x_{i'}, \mathcal{O}(x_{i'})\}_{i' \in [m]}$. It then samples $j-1$ keys as $\{k_{j'} \leftarrow \mathcal{K}\}_{j' \in [j-1]}$ and forms the matrix $\mathbf{M} \in \mathcal{Y}^{m \times d}$ as follows:

- If $j' < j$, set $\mathbf{M}_{i',j'} = F(k_{j'}, x_{i'})$.
- If $j' = j$, set $\mathbf{M}_{i',j'} = \mathcal{O}(x_{i'})$.
- If $j' > j$, for each $i' \in [m]$ and $j' \in \{j+1, \ldots, d\}$ sample a fresh $y \leftarrow \mathcal{Y}$ and set $\mathbf{M}_{i',j'} = y$.

Observe that $\mathbf{M} \equiv \mathcal{H}_{j-1}$ if $\mathcal{O}$ corresponds to a truly random function, and $\mathbf{M} \equiv \mathcal{H}_j$ if $\mathcal{O}$ corresponds to the pseudorandom function F. It follows that $\mathcal{H}_{j-1} \overset{c}{\approx} \mathcal{H}_j$.

In the second part of the proof, we show that $(\mathbf{F}, \mathbf{Fs}) \overset{c}{\approx} (\mathbf{F}, \mathbf{u})$. Given an attacker $\mathcal{A}$ that distinguishes $(\mathbf{F}, \mathbf{Fs})$ from $(\mathbf{F}, \mathbf{u})$, we describe an attacker $\mathcal{B}$ against the weak pseudorandomness of F. Given access to an oracle $\mathcal{O}$ which is either F or a truly random function, $\mathcal{B}$ invokes its oracle m times and receives $\{x_i, \mathcal{O}(x_i)\}_{i \in [m]}$. The reduction then samples d keys as $\{k_j \leftarrow \mathcal{K}\}_{j \in [d]}$ and forms the matrix $\mathbf{F}$ as $\mathbf{F}_{i,j} = F(k_j, x_i)$. Define the vectors $\mathbf{y}^* \in \mathcal{Y}^m$ and $\mathbf{k} \in \mathcal{K}^d$ as

$$\mathbf{k} = (k_1, \ldots, k_d), \qquad \mathbf{y}^* := (\mathcal{O}(x_1), \ldots, \mathcal{O}(x_m)) .$$

Finally, $\mathcal{B}$ runs $\mathcal{A}$ on the input $(\mathbf{F}, \mathbf{y}^*)$ and $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs. It is easy to see that if $\mathcal{O}$ is a truly random function we have $(\mathbf{F}, \mathbf{y}^*) \equiv (\mathbf{F}, \mathbf{u})$. Observe that by the leftover hash lemma, we have $(\mathbf{k}, \bigoplus_{\mathbf{s}} \mathbf{k}) \overset{s}{\approx} (\mathbf{k}, k^*)$ where k^* is uniform over $\mathcal{K}$. If $\mathbf{y}^*$ corresponds to the weak PRF outputs ($\mathcal{O}$ is the weak PRF), by key homomorphism of F we have

$$\mathbf{Fs} = \begin{pmatrix} F\left(\bigoplus_{\mathbf{s}} \mathbf{k}, x_1\right) \\ F\left(\bigoplus_{\mathbf{s}} \mathbf{k}, x_2\right) \\ \vdots \\ F\left(\bigoplus_{\mathbf{s}} \mathbf{k}, x_m\right) \end{pmatrix} \overset{s}{\approx} \begin{pmatrix} F\left(k^*, x_1\right) \\ F\left(k^*, x_2\right) \\ \vdots \\ F\left(k^*, x_m\right) \end{pmatrix} \equiv \mathbf{y}^* .$$

Therefore, the advantage of $\mathcal{B}$ (in the weak PRF game) is negligibly different from the advantage of $\mathcal{A}$. It follows that $(\mathbf{F}, \mathbf{Fs}) \overset{c}{\approx} (\mathbf{F}, \mathbf{u})$, as required.

Using the first part of the proof by a straightforward reduction we have $(\mathbf{Y}, \mathbf{Ys}) \overset{c}{\approx} (\mathbf{F}, \mathbf{Fs})$ and $(\mathbf{F}, \mathbf{u}) \overset{c}{\approx} (\mathbf{Y}, \mathbf{u})$. Using the second part, it follows that

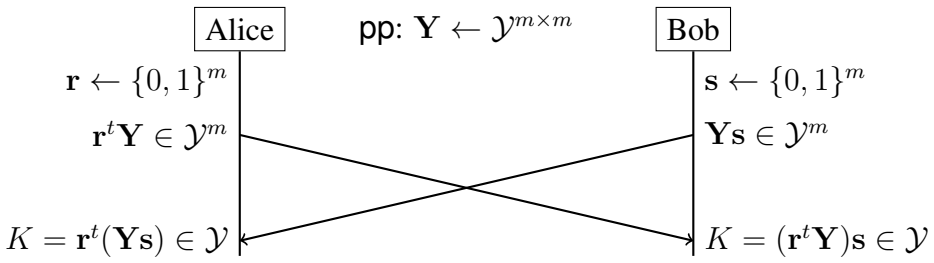
$$(\mathbf{Y}, \mathbf{Ys}) \overset{c}{\approx} (\mathbf{F}, \mathbf{Fs}) \overset{c}{\approx} (\mathbf{F}, \mathbf{u}) \overset{c}{\approx} (\mathbf{Y}, \mathbf{u}),$$

and hence we get $(\mathbf{Y}, \mathbf{Ys}) \overset{c}{\approx} (\mathbf{Y}, \mathbf{u})$. □

7.2 Public-Key Encryption

We now describe a noninteractive key-exchange protocol (which is sufficient to realize PKE) based on any KHwPRF whose output group is abelian. Later, we demonstrate a construction of an IHwPRF from any KHwPRF, which in turn implies a variety of cryptographic primitives. We first start with an inefficient protocol, and then we show how to improve its efficiency.

Given a KHwPRF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ such that $\mathcal{Y}$ is an abelian group, fix some positive integer $m > \log|\mathcal{K}| + \omega(\log(\lambda))$ and let $\mathbf{Y} \in \mathcal{Y}^{m \times m}$ be a matrix of uniformly chosen group elements from $\mathcal{Y}$. Alice (respectively, Bob) chooses binary vector $\mathbf{r} \leftarrow \{0,1\}^m$ (respectively, $\mathbf{s} \leftarrow \{0,1\}^m$), and sends $\mathbf{r}^t\mathbf{Y}$ (respectively, $\mathbf{Ys}$) to Bob (respectively, Alice). The final secret will be $\mathbf{r}^t(\mathbf{Ys}) = (\mathbf{r}^t\mathbf{Y})\mathbf{s} \in \mathcal{Y}$. The following figure is a simple visualization of the aforementioned protocol.



We sketch the security proof for the mentioned protocol. It is enough to show

$$(\mathbf{Y}, \mathbf{r}^t\mathbf{Y}, \mathbf{Ys}, \mathbf{r}^t\mathbf{Ys}) \overset{c}{\approx} (\mathbf{Y}, \mathbf{y}_1, \mathbf{y}_2, y),$$

where $\mathbf{Y} \leftarrow \mathcal{Y}^{m \times m}, \mathbf{r} \leftarrow \{0,1\}^m, \mathbf{s} \leftarrow \{0,1\}^m, \mathbf{y}_1 \leftarrow \mathcal{Y}^m, \mathbf{y}_2 \leftarrow \mathcal{Y}^m, y \leftarrow \mathcal{Y}$.

Observe that by Theorem 7.1.1 and a simple hybrid argument we can replace $\mathbf{r}^t\mathbf{Y}$ with a random vector $\mathbf{u} \leftarrow \mathcal{Y}^m$ and so

$$(\mathbf{Y}, \mathbf{u}, \mathbf{Ys}, \mathbf{us}) \overset{c}{\approx} (\mathbf{Y}, \mathbf{y}_1, \mathbf{y}_2, y).$$

Now let $\hat{\mathbf{Y}} \in \mathcal{Y}^{(m+1)\times m}$ be the matrix that has $\mathbf{Y}$ as its top submatrix and $\mathbf{u}$ as its last row. By applying Theorem 7.2.1 again, it follows that

$$(\hat{\mathbf{Y}}, \hat{\mathbf{Y}}\mathbf{s}) \overset{c}{\approx} (\hat{\mathbf{Y}}, y),$$

as required.

The reader may notice that the aforementioned key exchange protocol is too expensive in terms of communication complexity, i.e, to agree on some group element the parties need to exchange $2m^2$ group elements. Using the following lemma, we immediately get a key exchange protocol for which the whole cost of communication is twice the size of the final secret (like DDH).

Lemma 7.2.1. *Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a KHwPRF, and let $m > \log|\mathcal{K}| + \omega(\log\lambda)$ be a positive integer. For any PPT adversary we have*

$$(\mathbf{Y}, \mathbf{RY}, \mathbf{YS}, \mathbf{RYS}) \overset{c}{\approx} (\mathbf{Y}, \mathbf{Y}', \mathbf{Y}'', \mathbf{Y}''')$$

where $\mathbf{Y}, \mathbf{Y}', \mathbf{Y}'', \mathbf{Y}'''$ are matrices of uniform group elements in $\mathcal{Y}^{m\times m}$, and $\mathbf{S}$, $\mathbf{R}$ are uniform binary matrices, i.e., $\mathbf{R} \leftarrow \{0,1\}^{m\times m}$ and $\mathbf{S} \leftarrow \{0,1\}^{m\times m}$.[1]

Proof. The lemma follows from Theorem 7.1.1, and a standard hybrid argument. □

7.3 Input-Homomorphic weak PRF

Here we show a simple construction of an IHwPRF from any KHwPRF. We remark that although an IHwPRF implies a variety of cryptographic primitives, the constructions will not be necessarily efficient. More efficient constructions can be obtained by directly building the primitive using the assumption in Theorem 7.1.1.

Lemma 7.3.1. *Let $F : \mathcal{K}\times\mathcal{X} \to \mathcal{Y}$ be a KHwPRF. If $d > \log|\mathcal{K}| + +\omega(\log\lambda)$ be a positive integer and $\mathcal{Y}$ is an abelian group, the function $\tilde{F} : \{0,1\}^d \times \mathcal{Y}^d \to \mathcal{Y}$ defined as*

$$\tilde{F}\big(\mathbf{s} = (s_1, \ldots, s_d), \mathbf{y} = (y_1, \ldots, y_d)\big) = \bigotimes_{\mathbf{s}} \mathbf{y} = \bigotimes_{j:s_j=1} y_j$$

is an IHwPRF.

[1] Notice that for the correctness of key exchange, we require the group $\mathcal{Y}$ to be abelian.

Proof. First, observe that $\tilde{F}$ is input homomorphic since for any $\mathbf{y}, \mathbf{y}' \in \mathcal{Y}^d$ and $\mathbf{s} \in \{0,1\}^d$ we have

$$\begin{aligned}
\tilde{F}(\mathbf{s}, \mathbf{y}) \otimes \tilde{F}(\mathbf{s}, \mathbf{y}') &= \left(\bigotimes_{\mathbf{s}} \mathbf{y} \right) \otimes \left(\bigotimes_{\mathbf{s}} \mathbf{y}' \right) \\
&= \left(\bigotimes_{j:s_j=1} y_j \right) \otimes \left(\bigotimes_{j:s_j=1} y'_j \right) \\
&= \bigotimes_{j:s_j=1} (y_j \otimes y'_j) \\
&= \bigotimes_{\mathbf{s}} (\mathbf{y} \otimes \mathbf{y}') = \tilde{F}(\mathbf{s}, \mathbf{y} \otimes \mathbf{y}').
\end{aligned}$$

Given $m = \mathrm{poly}(\lambda)$ samples of the form $(\mathbf{y}_i, \mathcal{O}(\mathbf{y}_i))$, form the matrix $\mathbf{Y} \in \mathcal{Y}^{m \times d}$ such that the ith row of $\mathbf{Y}$ is $\mathbf{y}_i$. In addition, define $\mathbf{y}^*$ as $\mathbf{y}^* := (\mathcal{O}(\mathbf{y}_1), \ldots, \mathcal{O}(\mathbf{y}_m))$. Observe that if $\mathcal{O}$ is a truly random function then $\mathbf{y}^*$ is uniformly distributed in $\mathcal{Y}^m$. On the other hand, if $\mathcal{O}$ is the weak PRF, we have $\mathbf{y}^* = \mathbf{Y}\mathbf{s}$ for some uniform $\mathbf{s} \in \{0,1\}^d$. By applying Theorem 7.1.1 and observing the fact that $m = \mathrm{poly}(\lambda)$, it follows that F is a weak PRF. □

Implications. By plugging in the results of Chapter 5 and Chapter 6, and using the Lemma 7.3.1 it follows that KHwPRF imply noninteractive key exchange, private information retrieval [KO97], lossy trapdoor functions [PW08], identity-based encryption (in a non-blackbox manner) [DG17b, DG17a, BLSV18], and hinting PRG [KW19].

We remark that KHwPRF trivially implies homomorphic one-way function (HOWF) and hence using the results of Chapter 4, KHwPRF implies collision-resistant hash function, Schnorr signature, and chameleon hash function.

7.4 Asymmetric Primitives from Bounded KHwPRF

In this part, we show that the "approximate" (some prior works called it "almost") version of key-homomorphic weak PRF with certain properties imply a variety of asymmetric primitives, such as public-key encryption (PKE). Approximate KHwPRF has the property that $F_{k \oplus k'}(x)$ is *close* to $F_k(x) \otimes F_{k'}(x)$ where closeness is measured with respect to some distance function.

An Algebraic Definition. Formalizing a general definition for "approximate" homomorphism requires a somewhat involved *geometric* definition that needs a distance function, which also does

not nicely fit into the (algebraic) framework of Chapter 3. Therefore, we use the *algebraic* definition of *bounded* key-homomorphic weak PRF, which is similar to the definition of bounded IHwPRF (see Section 3.1 for more details).

Bounded KHwPRFs and LWR. All of the currently known instantiations of "approximate" key-homomorphic (weak) PRF use Learning With Rounding (LWR) [BPR12] as their underlying assumption. It is easy to see that if the *output* group of some LWR-based KHwPRF is $\mathbb{Z}_p^n$ for some *superpolynomial* modulus p and some dimension n, we can define the mapping $\mathcal{R}_{\mathsf{in}}$ (respectively, $\mathcal{R}_{\mathsf{out}}$) to be rounding with respect to some modulus p_{in} (respectively, p_{out}) such that p/p_{in} and $p_{\mathsf{in}}/p_{\mathsf{out}}$ are both superpolynomial.

The reader may note that the resulting construction of bounded KHwPRF from LWR has a triple rounding, one that is embedded in the (weak) PRF F and one for each mapping $\mathcal{R}_{\mathsf{out}}$ and $\mathcal{R}_{\mathsf{in}}$ in the definition of bounded KHwPRF. Although this property is inherent for the LWR-based construction, in general there may not be any similarity between F and $\mathcal{R}_{\mathsf{in}}$ or $\mathcal{R}_{\mathsf{out}}$ for a bounded KHwPRF.

PKE Construction from Bounded KHwPRF. We show a construction of public-key encryption scheme from a *bounded* KHwPRF. The construction is almost identical to the case of unbounded KHwPRF, with the difference being applying the mappings $\mathcal{R}_{\mathsf{in}}$ and $\mathcal{R}_{\mathsf{out}}$ of the bounded KHwPRF. The argument for the security is also similar to the exact/unbounded case, and we omit the details.

Given a γ-bounded KHwPRF $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ (with mappings $\mathcal{R}_{\mathsf{in}}$ and $\mathcal{R}_{\mathsf{out}}$ defined as in Section 3.1) such that $\mathcal{Y}$ and $\mathcal{Z}_{\mathsf{in}}$ are abelian groups and $\gamma > \log|\mathcal{K}| + \omega(\log(\lambda))$, fix some integer m such that $\gamma \geq m > \log|\mathcal{K}| + \omega(\log(\lambda))$ and let $\mathbf{Y} \in \mathcal{Y}^{m \times m}$ be a matrix of uniformly chosen group elements from $\mathcal{Y}$. Alice (respectively, Bob) chooses a binary vector $\mathbf{r} \leftarrow \{0,1\}^m$ (respectively, $\mathbf{s} \leftarrow \{0,1\}^m$), and sends $\mathcal{R}_{\mathsf{in}}(\mathbf{r}^t\mathbf{Y})$ (respectively, $\mathcal{R}_{\mathsf{in}}(\mathbf{Ys})$) to Bob (respectively, Alice). The final secret will be

$$\mathcal{R}_{\mathsf{out}}\big(\mathbf{r}^t\mathcal{R}_{\mathsf{in}}(\mathbf{Ys})\big) = \mathcal{R}_{\mathsf{out}}\big(\mathcal{R}_{\mathsf{in}}(\mathbf{r}^t\mathbf{Y})\mathbf{s}\big) \in \mathcal{Z}_{\mathsf{out}}.$$

Bounded IHwPRF from Bounded KHwPRF. We now construct a bounded IHwPRF from a *bounded* KHwPRF. The construction is almost identical to the case of unbounded KHwPRF, with the difference being applying the mappings $\mathcal{R}_{\mathsf{in}}$ and $\mathcal{R}_{\mathsf{out}}$ of the bounded KHwPRF.

Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a γ-bounded KHwPRF (with mappings $\mathcal{R}_{\mathsf{in}}$ and $\mathcal{R}_{\mathsf{out}}$ defined as in Section 3.1) such that $\mathcal{Y}$ and $\mathcal{Z}_{\mathsf{in}}$ are abelian groups and $\gamma > \log|\mathcal{K}| + \omega(\log(\lambda))$. Let d be an integer such that $\log|\mathcal{K}| + \omega(\log(\lambda)) < d \leq \gamma$. We define a bounded IHwPRF $\tilde{F} : \{0,1\}^d \times \mathcal{Y}^d \to \mathcal{Z}_{\mathsf{in}}$

with its associated mapping $\tilde{\mathcal{R}} : \mathcal{Z}_{\mathsf{in}} \to \mathcal{Z}_{\mathsf{out}}$ as

$$\tilde{F}\big(\mathbf{s} = (s_1, \ldots, s_d), \mathbf{y} = (y_1, \ldots, y_d)\big) = \mathcal{R}_{\mathsf{in}}\big(\bigotimes_{\mathbf{s}} \mathbf{y}\big) = \mathcal{R}_{\mathsf{in}}\Big(\bigotimes_{j:s_j=1} y_j\Big),$$

where $\tilde{\mathcal{R}}$ (the associated mapping with $\tilde{F}$) is identical to $\mathcal{R}_{\mathsf{out}}$. The security proof is similar to the exact/unbounded case, and we omit the details.

7.5 Naor-Reingold PRF

Here we show a construction of Naor-Reingold style PRF from any KHwPRF. Before we do so, we provide some background on the Naor-Reingold PRF and explain why PRFs in this style are important. We start by recalling the original Naor-Reingold PRF [NR97]:

Let $\mathbb{G}$ be a DDH-hard group of order p, and let $F_{NR} : \mathbb{Z}_p^{(\ell+1)} \times \{0,1\}^\ell \to \mathbb{G}$ be the function family defined by

$$F_{NR}\left(\{\alpha_j\}_{j\in[0,\ell]} \in \mathbb{Z}_p^{(\ell+1)}, \mathbf{x} \in \{0,1\}^\ell\right) = g^{\alpha_0 \prod_{i=1}^{\ell} \alpha_i^{x_i}},$$

where the values $\alpha_0, \alpha_1, \ldots, \alpha_\ell$ form the key and $\mathbf{x}$ is the input. Informally, a Naor-Reingold style PRF requires a constant number of computations (for instance, the exponentiation in F_{NR}) on which the assumption related to its hardness depends, while all of the operations that scale with the length of the input (for instance, the integer multiplications in the exponent of F_{NR}) are less expensive. This feature allows Naor-Reingold style PRFs to be potentially efficient. In particular, assuming that the underlying operations have reasonably low circuit depth, such PRFs typically have polylogarithmic evaluation circuits.

We now show a simple construction of Naor-Reingold style PRF from any exact KHwPRF with abelian output group. Our construction involves a subset product of binary matrices and one "multiplication" of a group matrix and an integer matrix. The depth of the PRF evaluation circuit is polylogarithmic provided that the group operation can be done efficiently.

Theorem 7.5.1. *Let $\tilde{F} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a KHwPRF, and fix some integer $m > \log|\mathcal{K}| + \omega(\log(\lambda))$. Let $\mathbf{Y} \in \mathcal{Y}^{m\times m}$ be a (public) matrix of group elements such that each entry $y_{i,j}$ (for $i \in [m], j \in [m]$) is drawn uniformly and independently from $\mathcal{Y}$. The function $F : \mathcal{Y}^{(\ell+1)\times m^2} \times \{0,1\}^\ell \to \mathcal{Y}^{m\times m}$ defined as*

$$F\Big((\mathbf{S}_0, \mathbf{S}_1, \ldots, \mathbf{S}_\ell), \mathbf{x} = (x_1, \ldots, x_\ell)\Big) = \mathbf{Y}\mathbf{S}_0 \prod_{i=1}^{\ell} \mathbf{S}_i^{x_i}$$

is a pseudorandom function where $\mathbf{S}_i \leftarrow \{0,1\}^{m\times m}$ *for* $i \in \{0,\ldots,\ell\}$.

Proof. To prove this theorem, we use the following lemma.

Lemma 7.5.2. *Let* $\mathbf{Y}_1,\ldots,\mathbf{Y}_Q \in \mathcal{Y}^{m\times m}$ *be matrices with uniformly and independently sampled entries from* $\mathcal{Y}$ *for some* $Q = \mathrm{poly}(\lambda)$*, and let* $\mathbf{S} \leftarrow \{0,1\}^{m\times m}$ *be a uniformly sampled binary matrix. Then for any PPT adversary we have*

$$\{(\mathbf{Y}_q, \mathbf{Y}_q\mathbf{S})\}_{q\in[Q]} \overset{c}{\approx} \{(\mathbf{Y}_q, \mathbf{U}_q)\}_{q\in[Q]}.$$

where for each $q \in [Q]$, $\mathbf{U}_q \leftarrow \mathcal{Y}^{m\times m}$ *is a matrix of uniformly chosen elements from* $\mathcal{Y}$.

This lemma follows directly from Theorem 7.1.1, and a standard hybrid argument over the columns of $\mathbf{Y}_q\mathbf{S}$. The proof of pseudorandomness now proceeds via a series of $(\ell+1)$ hybrids, where for each $j \in [0,\ell]$, the j^{th} hybrid is as described below.

1. The challenger samples $(\ell - j)$ uniform binary matrices as $\mathbf{S}_i \leftarrow \{0,1\}^{m\times m}$ for $i \in [j+1,\ell]$. It also maintains a list $\mathcal{L}$ of $m \times m$ matrices over the group $\mathcal{Y}$. Initially, this list is empty. The challenger also creates and stores an $m \times m$ matrix $\mathbf{Y}_0$ consisting of uniformly and independently sampled entries from $\mathcal{Y}$.

2. The adversary adaptively issues a maximum of $Q = \mathrm{poly}(\lambda)$ PRF queries of the form $\mathbf{x}_1,\ldots,\mathbf{x}_Q$, where for each $q \in [Q]$, we have $\mathbf{x}_q = (x_{1,q},\ldots,x_{\ell,q})$. For ease of representation, we divide each query string as $\mathbf{x}_q = (\mathbf{x}_q^{(0)}, \mathbf{x}_q^{(1)})$, where

$$\mathbf{x}_q^{(0)} = (x_{1,q},\ldots,x_{j,q}), \quad \mathbf{x}_q^{(1)} = (x_{j+1,q},\ldots,x_{\ell,q}).$$

3. Upon receipt of the q^{th} query, the challenger proceeds as follows:

   (a) If $j = 0$, it sets $\mathbf{Y}_q = \mathbf{Y}_0$.

   (b) Otherwise, it checks if there exists a $q' < q$ such that $\mathbf{x}_q^{(0)} = \mathbf{x}_{q'}^{(0)}$.

      i. If yes, it sets $\mathbf{Y}_q = \mathbf{Y}_{q'}$.

      ii. Otherwise, it sets $\mathbf{Y}_q$ to be an $m \times m$ matrix with uniformly and independently sampled entries from $\mathcal{Y}$.

   (c) It updates the list $\mathcal{L}$ as $\mathcal{L} = \mathcal{L} \cup \{\mathbf{Y}_q\}$ and responds to the q^{th} query as

$$f_{j,q} = \mathbf{Y}_q \prod_{i=j+1}^{\ell} \mathbf{S}_i^{x_{i,q}}.$$

Note that in the zero[th] hybrid, we replaced the component $\mathbf{YS}_0$ in the original PRF construction by an $m \times m$ matrix $\mathbf{Y}_0$ consisting of uniformly and independently sampled entries from $\mathcal{Y}$. It follows from Theorem 7.1.1 that this hybrid is indistinguishable from the real PRF experiment.

Now, for each $j \in [0, \ell]$, let $\mathcal{F}_j = \{f_{j,q}\}_{q \in [Q]}$ be the set of responses generated by the challenger in the j^{th} game. The proof of Theorem 7.5.1 now follows immediately from the following claim:

Claim 7.5.3. *For each $j \in [0, \ell - 1]$ and for any PPT adversary we have*

$$\mathcal{F}_j \overset{c}{\approx} \mathcal{F}_{j+1}.$$

Let $\mathcal{A}$ be a PPT adversary such that for some $j \in [\ell]$, $\mathcal{A}$ efficiently distinguishes between $\mathcal{F}_j$ and $\mathcal{F}_{j+1}$. We construct an attacker $\mathcal{B}$ against the assumption in Lemma 7.5.2. $\mathcal{B}$ receives as input a tuple of the form $\{(\mathbf{Y}_q, \mathbf{Z}_q)\}_{q \in [Q']}$ for some $Q' > Q$, where either each $\mathbf{Z}_q$ is of the form $\mathbf{Y}_q\mathbf{S}_j$ for some uniformly random $m \times m$ binary matrix $\mathbf{S}_j$, or each $\mathbf{Z}_q$ is a uniformly random matrix over $\mathcal{Y}^{m \times m}$. It proceeds as follows:

1. $\mathcal{B}$ samples $(\ell - j - 1)$ uniform binary matrices as $\mathbf{S}_i \leftarrow \{0,1\}^{m \times m}$ for $i \in [j+2, \ell]$. It also maintains a counter variable cnt. Initially, $\mathsf{cnt} = 1$.

2. $\mathcal{A}$ adaptively issues a maximum of $Q = \text{poly}(\lambda)$ PRF queries of the form $\mathbf{x}_1, \ldots, \mathbf{x}_Q$, where for each $q \in [Q]$, we have $\mathbf{x}_q = (x_{1,q}, \ldots, x_{\ell,q})$. Again, for ease of representation, we divide each query string as $\mathbf{x}_q = (\mathbf{x}_q^{(0)}, \mathbf{x}_q^{(1)})$, where

$$\mathbf{x}_q^{(0)} = (x_{1,q}, \ldots, x_{j,q}), \quad \mathbf{x}_q^{(1)} = (x_{j+1,q}, \ldots, x_{\ell,q}).$$

3. Upon receipt of the q^{th} query, $\mathcal{B}$ checks if there exists a $q' < q$ such that $\mathbf{x}_q^{(0)} = \mathbf{x}_{q'}^{(0)}$.

   (a) If yes, it sets $\widetilde{\mathbf{Y}}_q = \widetilde{\mathbf{Y}}_{q'}$ and $\widetilde{\mathbf{Z}}_q = \widetilde{\mathbf{Z}}_{q'}$.

   (b) Otherwise, it sets $\widetilde{\mathbf{Y}}_q = \mathbf{Y}_{\mathsf{cnt}}$ and $\widetilde{\mathbf{Z}}_q = \mathbf{Z}_{\mathsf{cnt}}$, and updates $\mathsf{cnt} = \mathsf{cnt} + 1$.

4. $\mathcal{B}$ now responds to the q^{th} query as

$$\tilde{f}_{j,q} = \begin{cases} \widetilde{\mathbf{Y}}_q \prod_{i=j+2}^{\ell} \mathbf{S}_i^{x_{i,q}} & \text{if } x_{j+1,q} = 0, \\ \widetilde{\mathbf{Z}}_q \prod_{i=j+2}^{\ell} \mathbf{S}_i^{x_{i,q}} & \text{if } x_{j+1,q} = 1. \end{cases}$$

5. Eventually, the adversary $\mathcal{A}$ outputs a bit b. $\mathcal{B}$ outputs the same bit b.

Let $\widetilde{\mathcal{F}} = \{\tilde{f}_{j,q}\}_{q\in[Q]}$ be the set of responses generated by $\mathcal{B}$. It is easy to see the following:

- If each $\mathbf{Z}_q$ is of the form $\mathbf{Y}_q\mathbf{S}_j$ for some uniformly random $m \times m$ binary matrix $\mathbf{S}_j$, then the distribution of $\widetilde{\mathcal{F}}$ is identical to that of $\mathcal{F}_j$.
- On the other hand, if each $\mathbf{Z}_q$ is a uniformly random matrix over $\mathcal{Y}^{m\times m}$, then the distribution of $\widetilde{\mathcal{F}}$ is identical to that of $\mathcal{F}_{j+1}$.

It now follows that the advantage of $\mathcal{B}$ is identical to that of $\mathcal{A}$. This completes the proof of Claim 7.5.3. The proof of Theorem 7.5.1 follows immediately. □

Naor-Reingold PRF from Bounded KHwPRF. Our definition of bounded KHwPRF does not allow a direct construction of Naor-Reingold PRF. However, there are known constructions of Naor-Reingold PRF from lattice-based assumptions [BPR12, BLMR13, Mon18]. Our algebraic definition of bounded KHwPRF does not encompass multiple levels of rounding, and hence we omit constructing Naor-Reingold PRF from bounded KHwPRF.

CHAPTER 8

A Framework Based on Group Actions

In the previous chapters, we discussed the power of basic symmetric primitives endowed with some algebraic structure, namely group homomorphism. However, there are certain cryptographic constructions that are not captured in the homomorphism-based framework, most notably the isogeny-based cryptography. In this chapter, we present a framework based on group actions.

8.1 Cryptographic Group Actions

In this section we present our definitions of cryptographic group actions. We use the definitions of Brassard and Yung [BY91] and Couveignes [Cou06] as starting points and aim to provide definitions that allow for easy use in cryptographic protocols. We mainly focus on regular group actions in this thesis. Recall that a group action $(G, X, \star)$ is said to be *regular* if it is *both* free *and* transitive. For such a regular group action, the set X is called a *principal homogeneous space* for the group G, or a G*-torsor*. We define an effective group action as follows.

Definition 8.1.1. A group action $(G, X, \star)$ is said to be *effective* if the following properties are satisfied:

- The group G is finite and there exist PPT algorithms for:
  1. Membership testing, i.e., to decide if a given bit string represents a valid group element.
  2. Equality testing, i.e., to decide if two bit strings represent the same group element.
  3. Sampling, i.e., to sample an element g from a distribution $\mathcal{D}_G$ on G. In this thesis, we consider distributions that are statistically close to uniform.
  4. Operation, i.e., to compute gh for any $(g, h) \in G^2$.
  5. Inversion, i.e., to compute g^{-1} for any $g \in G$.

- The set X is finite and there exist efficient algorithms for:

  1. Membership testing, i.e., to decide if a bit string represents a valid set element.
  2. Unique representation, i.e., given any arbitrary set element $x \in X$, compute a string $\hat{x}$ that canonically represents x.

- There exists a distinguished element $x_0 \in X$, called the *origin*, such that its representation as a bit string is known.

- There exists an efficient algorithm that given (some bit string representations of) any $g \in G$ and any $x \in X$, outputs $g \star x$.

We define some basic cryptographic primitives pertaining to group actions.

Definition 8.1.2. (One-Way Group Action.) A group action is said to be $(\mathcal{D}_X, \mathcal{D}_G)$-one-way if the family of efficiently computable functions

$$\{f_x : G \to X\}_{x \in X}$$

is $(\mathcal{D}_X, \mathcal{D}_G)$-one-way, where $f_x : g \mapsto g \star x$, and $\mathcal{D}_X$, $\mathcal{D}_G$ are distributions on X, G respectively.

Definition 8.1.3. (Weak Unpredictable Group Action.) Let $\mathcal{D}_X$ and $\mathcal{D}_G$ be distributions on X and G, respectively. A group action is said to be $(\mathcal{D}_G, \mathcal{D}_X)$-weakly unpredictable if the family of (efficiently computable) permutations $\{\pi_g : X \to X\}_{g \in G}$ is $(\mathcal{D}_G, \mathcal{D}_X)$-weakly unpredictable, where π_g is defined as $\pi_g : x \mapsto g \star x$.

Definition 8.1.4. (Weak Pseudorandom Group Action.) Let $\mathcal{D}_X$ and $\mathcal{D}_G$ be distributions on X and G, respectively. A group action is said to be $(\mathcal{D}_G, \mathcal{D}_X)$-weakly pseudorandom if the family $\{\pi_g : X \to X\}_{g \in G}$ of (efficiently computable) permutations is $(\mathcal{D}_G, \mathcal{D}_X)$-weakly pseudorandom, where $\pi_g : x \mapsto g \star x$.

In each of the definitions above, if $\mathcal{D}_G$ is a probability distribution on G and $\mathcal{D}_X$ is the distribution induced on X by taking $g \leftarrow \mathcal{D}_G$ and outputting $g \star x_0$, then we simply write $\mathcal{D}_G$-one-way group action (OWGA) for $(\mathcal{D}_X, \mathcal{D}_G)$-OW group action, and similarly for weakly unpredictable and weakly pseudorandom group actions, abbreviated as $(\mathcal{D}_G, \mathcal{D}_X)$-wU-GA and $(\mathcal{D}_G, \mathcal{D}_X)$-wPR-GA, respectively. If both distributions are uniform (or statistically close to uniform), we omit $(\mathcal{D}_G, \mathcal{D}_X)$.

In what follows, we will focus on group actions where G is abelian and the action is regular. We will characterize them by the computational assumption and their effectivity properties, and we

assume that they are abelian and regular unless stated otherwise. Therefore, we use the abbreviations OW-EGA/wU-EGA/wPR-EGA to denote one-way/weakly unpredictable/weakly pseudorandom abelian regular effective group action. We remark that Couveignes used the terminology *Hard Homogeneous Space* for wU-EGA, and *Very Hard Homogeneous Space* for wPR-EGA [Cou06]; subsequent literature on isogeny-based cryptography has mostly followed his conventions [DKS18, CLM+18].

Generic attacks. All known generic attacks against cryptographic group actions are attacks against the one-wayness. Given a pair $(x, g \star x)$, Stolbunov [Sto12] called the problem of finding g the *Group Action Inverse Problem (GAIP)*. The best known classical algorithm for GAIP is a meet-in-the-middle graph walk technique dating back to Pohl [Poh69], with a low-memory variant by Galbraith, Hess and Smart [GHS02], both running in time $O(\sqrt{|G|})$.

Childs, Jao and Soukharev [CJS14] were the first to point out that GAIP can be formulated as a *hidden shift problem*, and it can be solved by Kuperberg's quantum algorithm and its variants [Kup05, Reg04, Kup13], provided a quantum oracle to evaluate the group action. All of these algorithms have subexponential complexity between $\exp(\sqrt{\log N})$ and $L_N(1/2)$ where $N = |G|$.

In the context of isogenies, there are known classical and quantum attacks [Gal99, GS13, BIJ18, BS20, Pei20]. Very little is known in terms of of non-generic attacks: a recent result gives an attack against pseudorandomness [CSV20], which applies to some isogeny-based group actions but not to CSIDH and similar constructions.

Alternative axioms. In some circumstances, it is useful to strengthen or weaken the definition of an effective group action (EGA) by slightly modifying the set of axioms. Here we name a couple of most important variants.

- Uncertified EGA: Brassard and Yung [BY91] has considered group actions without the *Set Membership Testing* axiom. They call *certified* those group actions that have *Set Membership Testing*, and *uncertified* otherwise. It is easy to construct examples of uncertified actions, see, e.g., [BY91, §6.2].

- Hashable OW-EGA: In an OW-EGA, one can efficiently sample from X as follows: first sample $g \leftarrow \mathcal{D}_G$ using the *Group Sampling* axiom, then output $g \star x_0$. However in some applications it is useful to sample from X in a way that does not automatically reveal the group action inverse.

  In a *Hashable OW-EGA*, the existence of the *origin* x_0 is replaced with a *Hashing to the Set* axiom, stating that there exists an efficient sampler $H : [N] \to X$ (where the integer N

depends on the security parameter) such that for any adversary $\mathcal{A}$

$$\Pr[\mathcal{A}(i,j) \star H(i) = H(j)] \leq \mathrm{negl}(\lambda),$$

for $i, j \leftarrow [N]$.

Restricted Effective Group Action. An EGA is a useful abstraction, but sometimes it is too powerful in comparison to what is achievable in practice. A *Restricted Effective Group Action* (REGA) is a weakening of EGA, where we can only evaluate the action of a generating set of small cardinality.

Definition 8.1.5. (Restricted Effective Group Action (REGA).) Let $(G, X, \star)$ be a group action and let $\mathbf{g} = (g_1, \ldots, g_n)$ be a (not necessarily minimal) generating set for G. The action is said to be $\mathbf{g}$-*restricted effective*, if the following properties are satisfied:

- G is finite and $n = \mathrm{poly}(\log(|G|))$.
- The set X is finite and there exist efficient algorithms for:
  1. **Membership testing**, i.e., to decide if a bit string represents a valid set element.
  2. **Unique representation**, i.e., to compute a string $\hat{x}$ that canonically represents any given set element $x \in X$.
- There exists a distinguished element $x_0 \in X$, called the *origin*, such that its representation as a bit string is known.
- There exists an efficient algorithm that given any $i \in [n]$ and any bit string representation of $x \in X$, outputs $g_i \star x$ and $g_i^{-1} \star x$.

Although an REGA is limited to evaluations of the form $g_i \star x$, this is actually enough to evaluate the action of many, and potentially all elements of G without even needing axioms on the effectivity of G. By definition, any element of G can be represented by a word on $\mathbf{g}$, however this representation needs not be unique, nor equality needs to be efficiently testable. From the definition of a $\mathbf{g}$-REGA, it is clear that the action on $x \in X$ of any word of polynomial length on $\mathbf{g}$ can be computed in polynomial time.

When G is abelian, words on $\mathbf{g}$ can be rewritten as vectors in $\mathbb{Z}^n$, canonically mapped to G by

$$(a_1, \ldots, a_n) \mapsto \prod_{i=1}^{n} g_i^{a_i}.$$

It follows from the axioms of REGA that the action of a vector $\mathbf{a} \in \mathbb{Z}^n$ can be efficiently evaluated on any $x \in X$ as long as $\|\mathbf{a}\|$ is polynomial in $\log(|G|)$, where $\|\cdot\|$ is any L^p-norm.

Protocols built on REGA need to sample elements from G that are statistically close to uniform and for which the group action is efficiently computable. Prior works suggest sampling from a distribution on the words on $\mathbf{g}$ in the non-abelian case, or from a distribution on vectors in $\mathbb{Z}^n$ in the abelian case. Some classic choices in the abelian case are balls of fixed radius in L^∞-norm [CLM+18], L^1-norm [NOTT20], weighted infinity norms [Sto12, MR18], or discrete Gaussian distribution [DG19]. The latter is plausibly sufficient for applications that require group elements to be sampled from distributions that are statistically close to uniform [DG19].

Below, we describe a *known-order* EGA, which is a strengthening of EGA to the setting where the structure of G is known.

Known-order effective group action. As a strengthening of EGA, we may assume that the group structure of G is known. By "known order" we mean that a minimal list of generators $\mathbf{g} = (g_1, \ldots, g_n)$ together with their orders $(m_1, \ldots, m_n)$ is known, which in turn is equivalent to a decomposition

$$G \simeq \mathbb{Z}_{m_1} \oplus \cdots \oplus \mathbb{Z}_{m_n}.$$

An important special case is when G is cyclic, i.e., $G = \langle g \rangle \simeq \mathbb{Z}/m\mathbb{Z}$.

Denote by $\mathcal{L}$ the lattice $m_1\mathbb{Z} \oplus \cdots \oplus m_n\mathbb{Z}$, the map $\phi : \mathbb{Z}^n/\mathcal{L} \to G$ defined as

$$(a_1, \ldots, a_n) \mapsto \prod_{i=1}^{n} g_i^{a_i}$$

is an effective isomorphism, its inverse being a generalized discrete logarithm. If $(G, X, \star)$ is an EGA, then it is immediate to verify that $(\mathbb{Z}^n/\mathcal{L}, X, \star)$ is an EGA through ϕ. We may just use $\mathbb{Z}^n/\mathcal{L}$ as the standard representation for G.

Definition 8.1.6. (Known-order Effective Group Action (KEGA).) A *known-order effective group action* is an EGA $(\mathbb{Z}^n/\mathcal{L}, X, \star)$ where the lattice $\mathcal{L}$ is given by the tuple $(m_1, \ldots, m_n)$.

It may look like we "lose some cryptography" when we replace the group G by its isomorphic image $\mathbb{Z}^n/\mathcal{L}$. However, we stress that the main purpose of cryptography based on group actions is to design protocols that do not rely on discrete log assumptions. Thus, as soon as the group structure of G is known, KEGA is a more appropriate tool to design protocols, owing to its simplicity. For examples of protocols that require the KEGA setting, see [DM20].

Furthermore, KEGA and abelian EGA are quantumly equivalent. Indeed, given any abelian group G, Shor's algorithm and its generalization [Sho97, CM01] precisely compute an isomorphism $G \simeq \mathbb{Z}_{m_1} \oplus \cdots \oplus \mathbb{Z}_{m_n}$ (along with a minimal set of generators) in quantum polynomial time.

Remark 8.1.7. An REGA of known order is not automatically a KEGA, indeed the list of generators g of a REGA need not be minimal. As an extreme example, consider the case where $G = \langle g_1 \rangle$ is cyclic, and $\mathsf{g} = (g_1, \ldots, g_n)$. Any element of G can be uniquely represented as an integer in $\mathbb{Z}_{m_1}$, however this representation does not lead to an efficiently computable group action.

What is needed is an efficient algorithm to convert between the "minimal" representation $G \simeq \mathbb{Z}/\mathcal{L}$, and products of small powers of $(g_1, \ldots, g_n)$. In some instances, this conversion is possible via lattice reduction techniques [BKV19].

8.2 Two-Message Statistically Sender-Private OT

In this section, we demonstrate how to construct a two-message *statistically* sender-private oblivious transfer (SSP-OT) protocol without trusted setup based on any weak pseudorandom effective group action. We begin by recalling the definition of SSP-OT (we adopt the notation from [HLOV11, BD18]).

Definition 8.2.1. (Two-Message SSP-OT.) A two-message SSP-OT protocol is a tuple of three algorithms $(\mathsf{OTR}, \mathsf{OTS}, \mathsf{OTD})$ that satisfies correctness, receiver privacy, and statistical sender privacy (described below).

- $\mathsf{OTR}(1^\lambda, \beta)$: Given λ and a bit $\beta \in \{0,1\}$, outputs a message ot_1 and a (secret) state st.
- $\mathsf{OTS}(1^\lambda, (\mathsf{m}_0, \mathsf{m}_1), \mathsf{ot}_1)$: Given λ, a pair of bits $(\mathsf{m}_0, \mathsf{m}_1)$, and a message ot_1, outputs a message ot_2.
- $\mathsf{OTD}(1^\lambda, \mathsf{st}, \beta, \mathsf{ot}_2)$: Given λ, a secret state st, a bit $\beta \in \{0,1\}$, and a message ot_2, it outputs a bit $\mathsf{m}' \in \{0,1\}$.

The following properties should be satisfied:

Correctness. For any bit $\beta \in \{0,1\}$, and any pair of bits $\mathsf{m}_0, \mathsf{m}_1$, letting

$$(\mathsf{ot}_1, \mathsf{st}) = \mathsf{OTR}(1^\lambda, \beta), \quad \mathsf{ot}_2 = \mathsf{OTS}(1^\lambda, (\mathsf{m}_0, \mathsf{m}_1), \mathsf{ot}_1), \quad \mathsf{m}' = \mathsf{OTD}(1^\lambda, \mathsf{st}, \beta, \mathsf{ot}_2),$$

we have $\mathsf{m}' = \mathsf{m}_\beta$ with overwhelming probability.

Receiver Privacy. If $(\mathsf{ot}_1, \mathsf{st}) \leftarrow \mathsf{OTR}(1^\lambda, 0)$ and $(\mathsf{ot}'_1, \mathsf{st}') \leftarrow \mathsf{OTR}(1^\lambda, 1)$ be the receiver's output on 0 and 1 respectively, then $\mathsf{ot}_1 \overset{c}{\approx} \mathsf{ot}'_1$.

Statistical Sender Privacy. For any bit $\beta \in \{0,1\}$, any message ot_1, and two pairs of bits $(\mathsf{m}_0, \mathsf{m}_1)$ and $(\mathsf{m}'_0, \mathsf{m}'_1)$ such that $\mathsf{m}_\beta = \mathsf{m}'_\beta$, we have

$$\mathsf{OTS}(1^\lambda, (\mathsf{m}_0, \mathsf{m}_1), \mathsf{ot}_1) \approx_s \mathsf{OTS}(1^\lambda, (\mathsf{m}'_0, \mathsf{m}'_1), \mathsf{ot}_1).$$

Construction. Let $(G, X, \star)$ be a weak pseudorandom effective group action and let $\ell = \omega(\log(\lambda))$ be a parameter such that $H : X^\ell \to \{0,1\}$ is a pairwise independent hash function. Our construction of SSP-OT is as follows.

- $\mathsf{OTR}(1^\lambda, \beta)$: First, sample $\bar{x}_0 \leftarrow X$ and $\bar{x}_1 \leftarrow X$. Also, sample group elements $s \leftarrow G$ and $u \leftarrow G$ such that $s \neq u$, and output $\mathsf{ot}_1 = (\bar{x}_0, \bar{x}_1, y, x_0, x_1)$ and $\mathsf{st} = s$, where

$$y = s \star \bar{x}_0, \quad x_\beta = s \star \bar{x}_1, \quad x_{1-\beta} = u \star \bar{x}_1.$$

- $\mathsf{OTS}(1^\lambda, (\mathsf{m}_0, \mathsf{m}_1), \mathsf{ot}_1)$: Parse $\mathsf{ot}_1 = (\bar{x}_0, \bar{x}_1, y, x_0, x_1)$. If $x_0 = x_1$ output $\perp$. Otherwise, sample two vectors of group elements and two binary strings as

$$\mathbf{r}^{(0)} \leftarrow G^\ell, \quad \mathbf{r}^{(1)} \leftarrow G^\ell, \quad \mathbf{b}^{(0)} \leftarrow \{0,1\}^\ell, \quad \mathbf{b}^{(1)} \leftarrow \{0,1\}^\ell.$$

  Let $(\mathbf{c}^{(0)}, \mathbf{c}^{(1)})$ and $(\mathbf{d}^{(0)}, \mathbf{d}^{(1)})$ be two pairs of vectors of set elements such that for each $i \in [\ell]$ and each $\mu \in \{0,1\}$ we have

$$c_i^{(\mu)} = r_i^{(\mu)} \star \bar{x}_{b_i^{(\mu)}}, \quad d_i^{(\mu)} = \begin{cases} r_i^{(\mu)} \star y & \text{if } b_i^{(\mu)} = 0, \\ r_i^{(\mu)} \star x_\mu & \text{if } b_i^{(\mu)} = 1. \end{cases}$$

  Let $\mathcal{H}$ be the description of a pairwise independent hash function H. Output the message ot_2 as

$$\mathsf{ot}_2 = (\mathcal{H}, \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \mathsf{e}_0, \mathsf{e}_1),$$

  where $\mathsf{e}_\mu = H(\mathbf{d}^{(\mu)}) \oplus \mathsf{m}_\mu$ for each $\mu \in \{0,1\}$.

- $\mathsf{OTD}(1^\lambda, \mathsf{st} = s, \beta, \mathsf{ot}_2)$: Parse the message ot_2 as $(\mathcal{H}, \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \mathsf{e}_0, \mathsf{e}_1)$, and let H be the function whose description is $\mathcal{H}$. Output the bit m' as

$$\mathsf{m}' = H(s \star c_1^{(\beta)}, \ldots, s \star c_\ell^{(\beta)}) \oplus \mathsf{e}_\beta.$$

Correctness of the scheme follows by inspection. In the next lemmas, we prove that our construction satisfies receiver privacy and statistical sender privacy.

Lemma 8.2.2. *If $(G, X, \star)$ be a weak pseudorandom EGA, the construction above satisfies receiver privacy.*

Proof. By the weak pseudorandomness of group action, an immediate reduction implies that the receiver's message to the sender when $\beta = 0$ is computationally indistinguishable from a random tuple, i.e.,

$$(\bar{x}_0, \bar{x}_1, s \star \bar{x}_0, s \star \bar{x}_1, u \star \bar{x}_1) \stackrel{c}{\approx} (\bar{x}_0, \bar{x}_1, z_0, z_1, u \star \bar{x}_1),$$

where $z_0 \leftarrow X$ and $z_1 \leftarrow X$ are uniformly chosen set elements. A similar reduction implies that the receiver's message to sender when $\beta = 1$ is computationally indistinguishable from a random tuple, i.e., we have

$$(\bar{x}_0, \bar{x}_1, s \star \bar{x}_0, u \star \bar{x}_1, s \star \bar{x}_1) \stackrel{c}{\approx} (\bar{x}_0, \bar{x}_1, z_0, u \star \bar{x}_1, z_1).$$

Since the receiver's message is computationally distinguishable from a random tuple in both of the cases $\beta = 0$ and $\beta = 1$, it follows that the scheme satisfies receiver's privacy, as required. □

Lemma 8.2.3. *The construction of SSP-OT satisfies statistical sender privacy.*

Proof. Let $\mathsf{ot}_1 = (\bar{x}_0, \bar{x}_1, y, x_0, x_1)$ be an arbitrary message from a (possibly malicious) receiver. Notice that if $x_0 = x_1$ the sender outputs $\perp$, so we can assume that $x_0 \neq x_1$. Observe that if $x_0 \neq x_1$ there is some $\beta' \in \{0, 1\}$ for which there does *not* exist any $g \in G$ such that $y = g \star \bar{x}_0$ and $x_{\beta'} = g \star \bar{x}_1$. Without loss of generality we can assume that $\beta' = 0$, and we show that in this case e_0 component of sender's message is statistically indistinguishable from a uniform bit (and hence m_0 is statistically hidden). We let h and h' be two group elements such that $h \star \bar{x}_0 = y$ and $h' \star \bar{x}_1 = x_0$.

We make an argument similar to the proof of smoothness property for the construction of smooth projective hash functions. Recall that the sender's output has the form $(\mathcal{H}, \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \mathsf{e}_0, \mathsf{e}_1)$ where $\mathsf{e}_\mu = H(\mathbf{d}^{(\mu)}) \oplus \mathsf{m}_\mu$ for each $\mu \in \{0, 1\}$. First, observe that for each $i \in [\ell]$, there exists group elements $g_{i,0} \in G$ and $g_{i,1} \in G$ such that for each $\mu \in \{0, 1\}$ we have

$$g_{i,\mu} \star \bar{x}_\mu = c_i^{(0)}.$$

Since $h \neq h'$ it follows that for each $i \in [\ell]$ we have $H_\infty\big(d_i^{(0)} \mid (\mathsf{ot}_1, c_i^{(0)})\big) = 1$, which in turn implies that

$$H_\infty\big(\mathbf{d}^{(0)} \mid (\mathsf{ot}_1, \mathbf{c}^{(0)})\big) = \ell.$$

By the leftover hash lemma and using the fact that H is a pairwise independent hash function, it follows that

$$(\mathcal{H}, \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \mathsf{e}_0, \mathsf{e}_1) \stackrel{s}{\approx} (\mathcal{H}, \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \mathsf{e}', \mathsf{e}_1),$$

where e' is a uniform bit. A similar argument implies that if $\beta' = 1$ then

$$(\mathcal{H}, \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \mathsf{e}_0, \mathsf{e}_1) \stackrel{s}{\approx} (\mathcal{H}, \mathbf{c}^{(0)}, \mathbf{c}^{(1)}, \mathsf{e}_0, \mathsf{e}').$$

So the construction satisfies statistical sender privacy, as desired. □

The aforementioned lemmas yield the following theorem.

Theorem 8.2.4. *There exists a construction of two-message* SSP-OT *in the plain model from any weak pseudorandom effective group action.*

Remark 8.2.5. We note that our construction and proof work in essentially the same way from a restricted EGA provided that we can sample group elements from a distribution that is *statistically* close to uniform over the group G while retaining the ability to efficiently compute the action. We note that this is plausibly the case with respect to the instantiation of restricted EGA from CSIDH and other similar isogeny-based assumptions. We refer the reader to [DG19] for more details.

Further Applications. Two-message statistically sender-private oblivious transfer in the plain model has many applications such as non-malleable commitment [KS17], two-round witness indistinguishable proofs with private-coin verifier [JKKR17, BGI$^+$17, KKS18], and three-message statistical receiver-private OT in the plain model [GJJM20]. Thus, the SSP-OT construction above enables realization of these primitives from any weak pseudorandom (R)EGA.

8.3 Dual-Mode Public-Key Encryption

We recall the definition of dual-mode PKE from [PVW08] and then we show our construction of dual-mode PKE from any weak pseudorandom EGA.

Definition 8.3.1. (Dual-Mode PKE.) A dual-mode public-key encryption scheme is a tuple of six PPT algorithms $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FindMessy}, \mathsf{TrapKeyGen})$ that satisfy correctness, mode indistinguishability, messy branch identification, and decryptability on both branches in decryption mode (described below).

- $\mathsf{Setup}(1^\lambda, \mathsf{mode})$: Given the parameter λ and $\mathsf{mode} \in \{\mathsf{dec}, \mathsf{messy}\}$, outputs a public parameter pp and a trapdoor t.
- $\mathsf{Gen}(\mathsf{pp}, \sigma \in \{0,1\})$: Given the public parameter pp and a branch $\sigma \in \{0,1\}$, outputs $(\mathsf{pk}, \mathsf{sk})$, where pk is a public key and sk is a corresponding secret key for the branch σ.
- $\mathsf{Enc}(\mathsf{pk}, \beta \in \{0,1\}, \mathsf{m})$: Given a public key pk, a branch $\beta \in \{0,1\}$ and a message m, outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: Given a secret key sk and a ciphertext ct, outputs a message m'.
- $\mathsf{FindMessy}(\mathsf{t}, \mathsf{pk})$: Given a trapdoor t and a (possibly malformed) public key pk, outputs a branch value $\beta \in \{0,1\}$ corresponding to a messy branch of pk.
- $\mathsf{TrapKeyGen}(\mathsf{t})$: Given a trapdoor t, outputs $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1)$, where pk is a public key and $\mathsf{sk}_0, \mathsf{sk}_1$ are corresponding secret keys for branches 0 and 1.

The following properties should be satisfied:

- **Correctness:** For any $\mathsf{mode} \in \{\mathsf{dec}, \mathsf{messy}\}$, any branch $\beta \in \{0,1\}$, and any message m, letting $(\mathsf{pp}, \mathsf{t}) = \mathsf{Setup}(1^\lambda, \mathsf{mode})$, $(\mathsf{pk}, \mathsf{sk}) = \mathsf{Gen}(\mathsf{pp}, \mu)$ we have,

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, \mu, \mathsf{m})) = \mathsf{m}] \leq \mathrm{negl}(\lambda).$$

- **Mode indistinguishability:** If $\mathsf{pp}_{\mathsf{dec}}$ and $\mathsf{pp}_{\mathsf{messy}}$ be the public parameters obtained by running $\mathsf{Setup}(1^\lambda, \mathsf{dec})$ and $\mathsf{Setup}(1^\lambda, \mathsf{messy})$ respectively, it holds that $\mathsf{pp}_{\mathsf{dec}} \overset{c}{\approx} \mathsf{pp}_{\mathsf{messy}}$.
- **Messy branch identification:** For any pair $(\mathsf{pp}_{\mathsf{messy}}, \mathsf{t}_{\mathsf{messy}}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{messy})$ and any (possibly malformed) public key pk, the algorithm $\mathsf{FindMessy}$ outputs a branch $\beta \in \{0,1\}$ such that for any two messages $\mathsf{m}_0, \mathsf{m}_1$ we have

$$\mathsf{Enc}(\mathsf{pk}, \beta, \mathsf{m}_0) \overset{s}{\approx} \mathsf{Enc}(\mathsf{pk}, \beta, \mathsf{m}_1).$$

- **Decryptability on both branches:** For any $(\mathsf{pp}_{\mathsf{dec}}, \mathsf{t}_{\mathsf{dec}}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{dec})$, the algorithm $\mathsf{TrapKeyGen}$ on the input $\mathsf{t}_{\mathsf{dec}}$ outputs $\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1$ such that for any $\sigma \in \{0,1\}$ we have

$$(\mathsf{pk}, \mathsf{sk}_\sigma) \overset{s}{\approx} \mathsf{Gen}(\mathsf{pp}_{\mathsf{dec}}, \sigma).$$

Construction. Let $(G, X, \star)$ be a weak pseudorandom EGA and let $\ell = \omega(\log \lambda)$ be an integer such that $H : X^\ell \to \{0,1\}$ is a pairwise independent hash function. Our construction of dual-mode PKE is as follows.

- $\mathsf{Setup}(1^\lambda, \mathsf{mode})$: Proceed as follows depending on mode.

  If $\mathsf{mode} = \mathsf{dec}$ sample $\bar{x}_0 \leftarrow X$ and $t \leftarrow G$, and set $\bar{x}_1 = t \star \bar{x}_0$. Sample $h \leftarrow G$ and output

  $$\mathsf{pp}_{\mathsf{dec}} = (\bar{x}_0, \bar{x}_1, h \star \bar{x}_0, h \star \bar{x}_1), \quad \mathsf{t}_{\mathsf{dec}} = t.$$

  If $\mathsf{mode} = \mathsf{messy}$ sample set elements $\bar{x}_0 \leftarrow X$ and $\bar{x}_1 \leftarrow X$. In addition, sample $t_0 \leftarrow G$ and $t_1 \leftarrow G$ such that $t_0 \neq t_1$, and output

  $$\mathsf{pp}_{\mathsf{messy}} = (\bar{x}_0, \bar{x}_1, t_0 \star \bar{x}_0, t_1 \star \bar{x}_1), \quad \mathsf{t}_{\mathsf{messy}} = (t_0, t_1).$$

- $\mathsf{Gen}(\mathsf{pp} = (\bar{x}_0, \bar{x}_1, x_0, x_1), \sigma \in \{0,1\})$: Sample $s \leftarrow G$ and output

  $$\mathsf{sk} = s, \quad \mathsf{pk} = (s \star \bar{x}_\sigma, s \star x_\sigma).$$

- $\mathsf{Enc}(\mathsf{pk} = (\bar{x}, x), \beta \in \{0,1\}, \mathsf{m})$: Let $\mathsf{pp} = (\bar{x}_0, \bar{x}_1, x_0, x_1)$. Sample two vectors $\mathbf{r} \leftarrow G^\ell$ and $\mathbf{b} \leftarrow \{0,1\}^\ell$. Let $\mathbf{c} \in X^\ell$ and $\mathbf{c}' \in X^\ell$ be two vectors such that

  $$c_i = \begin{cases} r_i \star \bar{x}_\beta & \text{if } b_i = 0 \\ r_i \star x_\beta & \text{if } b_i = 1 \end{cases}, \quad c'_i = \begin{cases} r_i \star \bar{x} & \text{if } b_i = 0 \\ r_i \star x & \text{if } b_i = 1. \end{cases}$$

  Output the ciphertext $\mathsf{ct} = (\mathbf{c}, H(\mathbf{c}') \oplus \mathsf{m})$.

- $\mathsf{Dec}(\mathsf{sk} = s, \mathsf{ct})$: Parse $\mathsf{ct} = (\mathbf{c} \in X^\ell, \mathsf{e} \in \{0,1\})$ and output the message

  $$\mathsf{m}' = H(s \star c_1, \ldots, s \star c_\ell) \oplus \mathsf{e}.$$

- $\mathsf{FindMessy}(\mathsf{t}_{\mathsf{messy}}, \mathsf{pk} = (\bar{x}, x))$: Let $\mathsf{pp} = (\bar{x}_0, \bar{x}_1, x_0, x_1)$ and $\mathsf{t}_{\mathsf{messy}} = (t_0, t_1)$. If $x \neq t_0 \star \bar{x}$, output 0 as the messy branch. Otherwise, output 1.

- $\mathsf{TrapKeyGen}(\mathsf{t}_{\mathsf{dec}} = t)$: Let $\mathsf{pp} = (\bar{x}_0, \bar{x}_1, x_0, x_1)$. Sample $s \leftarrow G$ and output

  $$\mathsf{sk}_0 = ts, \quad \mathsf{sk}_1 = s, \quad \mathsf{pk} = (s \star \bar{x}_1, s \star x_1).$$

It is straightforward to see that the scheme satisfies correctness. Next, we show that our construction satisfies the mode indistinguishability property.

Lemma 8.3.2. *If $(G, X, \star)$ be a weak pseudorandom EGA, the construction above satisfies mode indistinguishability.*

Proof. Notice that $\mathsf{pp}_{\mathsf{messy}}$ is statistically close to a random tuple, i.e.,

$$(\bar{x}_0, \bar{x}_1, t_0 \star \bar{x}_0, t_1 \star \bar{x}_1) \stackrel{s}{\approx} (\bar{x}_0, \bar{x}_1, u_0, u_1),$$

where $u_0 \leftarrow X$ and $u_1 \leftarrow X$. On the other hand by the weak pseudorandomness of group action we have

$$(\bar{x}_0, \bar{x}_1, h \star \bar{x}_0, h \star \bar{x}_1) \stackrel{c}{\approx} (\bar{x}_0, \bar{x}_1, u_0, u_1).$$

It follows that $\mathsf{pp}_{\mathsf{messy}} \stackrel{c}{\approx} \mathsf{pp}_{\mathsf{dec}}$, which completes the proof. □

Lemma 8.3.3. *The construction of dual-mode PKE satisfies messy branch identification property.*

Proof. Let $(\mathsf{pp}, \mathsf{t}) = \mathsf{Setup}(1^\lambda, \mathsf{messy})$, and let pk be a (possibly malformed) public key. In addition, let $\beta = \mathsf{FindMessy}(\mathsf{t}, \mathsf{pk})$ be the messy branch identified by the $\mathsf{FindMessy}$ algorithm. First, observe that by construction there exists group elements $t_0 \neq t_1$ such that $x_b = t_b \star \bar{x}_b$ for both $b = 0$ and $b = 1$. In addition, if $\mathsf{pk} = (\bar{x}, x)$ then (by construction of $\mathsf{FindMessy}$ algorithm) we have $t_\beta \star \bar{x} \neq x$. Therefore, there exists *distinct* group elements $\bar{g}$ and g such that

$$\bar{x} = \bar{g} \star \bar{x}_\beta, \quad x = g \star x_\beta.$$

Now, observe that a ciphertext encrypting a message $\mathsf{m} \in \{0,1\}$ under public key $(\bar{x}, x)$ has the form $(\mathbf{c}, H(\mathbf{c}') \oplus \mathsf{m})$ where

$$c_i = \begin{cases} r_i \star \bar{x}_\beta & \text{if } b_i = 0 \\ r_i \star x_\beta & \text{if } b_i = 1 \end{cases}, \quad c'_i = \begin{cases} r_i \star \bar{x} & \text{if } b_i = 0 \\ r_i \star x & \text{if } b_i = 1, \end{cases}$$

and the vectors $\mathbf{r} \leftarrow G^\ell$ and $\mathbf{b} \leftarrow \{0,1\}^\ell$ are the randomness for encryption. The rest of argument is similar to the proof of smoothness property for the construction of hash proof system. Specifically, observe that for each $i \in [\ell]$ there exists $\bar{d}_i \in G$ and $d_i \in G$ such that

$$\bar{d}_i \star \bar{x}_\beta = d_i \star x_\beta = c_i.$$

This means that given the tuple of set elements $(\bar{x}_\beta, x_\beta, \bar{x}, x, c_i)$ the bit b_i in the randomness has full entropy. In addition, since $\bar{g} \neq g$ it follows that given the tuple $(\bar{x}_\beta, x_\beta, \bar{x}, x, c_i)$, the element c'_i has one bit of entropy. By extending this argument to all components and using the leftover hash lemma, it follows that

$$(\bar{x}_\beta, x_\beta, \bar{x}, x, \mathbf{c}, H(\mathbf{c}')) \stackrel{s}{\approx} (\bar{x}_\beta, x_\beta, \bar{x}, x, \mathbf{c}, u),$$

where $u \leftarrow \{0,1\}$ is a random bit. Since the last component is statistically close to uniform, it follows that for any two messages m_0 and m_1 we have

$$\mathsf{Enc}(\mathsf{pk}, \beta, \mathsf{m}_0) \stackrel{s}{\approx} \mathsf{Enc}(\mathsf{pk}, \beta, \mathsf{m}_1),$$

which completes the proof. □

Lemma 8.3.4. *The construction of dual-mode PKE scheme satisfies the property of decryptability on both branches in the decryption mode.*

Proof. Let $(\mathsf{pp}, \mathsf{t}) = \mathsf{Setup}(1^\lambda, \mathsf{dec})$ and let

$$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{pp}, \sigma), \quad (\overline{\mathsf{pk}}, \mathsf{sk}_0, \mathsf{sk}_1) \leftarrow \mathsf{TrapKeyGen}(\mathsf{t}),$$

for some $\sigma \in \{0,1\}$. It is easy to see that if $\sigma = 1$ we have $(\overline{\mathsf{pk}}, \mathsf{sk}_\sigma) \stackrel{s}{\approx} \mathsf{Gen}(\mathsf{pp}, \sigma)$. When $\sigma = 0$, the algorithm $\mathsf{TrapKeyGen}$ will output the keys

$$\mathsf{sk} = ts \quad , \quad \overline{\mathsf{pk}} = (s \star \bar{x}_1, s \star x_1).$$

Observe that if s and t are uniform, the distribution of ts is also uniform (because G is a group). Therefore, we can write

$$(s, s \star \bar{x}_0, s \star x_0) \stackrel{s}{\approx} (ts, ts \star \bar{x}_0, ts \star x_0).$$

It is easy to see that $(ts \star \bar{x}_0, ts \star x_0) = (s \star \bar{x}_1, s \star x_1)$. It follows that $(\overline{\mathsf{pk}}, \mathsf{sk}_\sigma) \stackrel{s}{\approx} \mathsf{Gen}(\mathsf{pp}, \sigma)$ also holds when $\sigma = 0$, and the proof is complete. □

The following theorem follows from combining the aforementioned lemmas.

Theorem 8.3.5. *There exists a construction of dual-mode PKE from any weak pseudorandom effective group action.*

Remark 8.3.6. We note that our construction and proof work in essentially the same way from a restricted EGA provided that we can sample group elements from a distribution that is *statistically* close to uniform over the group G while retaining the ability to efficiently compute the action. We note that this is plausibly the case with respect to the instantiation of restricted EGA from CSIDH and other similar isogeny-based assumptions. We refer the reader to [DG19] for more details.

Further Applications. Dual-mode public-key encryption implies two-message OT protocols that are *UC secure* in the common reference string model against corrupt receivers and senders [PVW08]. Such OT protocols are sufficient to construct round-optimal multiparty computation protocols for general functionalities that are *UC secure* in the common reference string model against static, malicious adversaries [GS18]. Hence, our construction of dual-mode PKE allows all of these primitives to be built from any weak pseudorandom (R)EGA.

8.4 Hash Proof System

In this section, we demonstrate how to construct universal and smooth projective hashing schemes (also known as hash proof systems or projective hash functions) from any weak pseudorandom effective group action. We begin by recalling the definition of a universal projective hashing scheme as in [CS02].

Definition 8.4.1. Let $\Lambda : K \times \Sigma \to \Gamma$ be an efficiently computable function, and let $L \subset \Sigma$. In addition, let $\alpha : K \to P$ be a "projection" function. We say that the tuple $\Pi = (\Lambda, K, P, \Sigma, \Gamma, L)$ is a universal projective hash function if the following properties hold:

- **Samplability:** There exist efficient algorithms to sample uniformly from Σ and from K. In addition, there exists an efficient algorithm to sample uniformly from L along with a witness w that proves membership in L.
- **Subset Membership Problem:** If $\sigma_0 \leftarrow L$ and $\sigma_1 \leftarrow \Sigma$ then $\sigma_0 \overset{c}{\approx} \sigma_1$.
- **Projective Evaluation:** There exists an efficient algorithm $\mathsf{ProjEval}$ such that for any $\mathsf{hk} \in K$ and any $\sigma \in L$ with membership witness w, we have

$$\mathsf{ProjEval}(\alpha(\mathsf{hk}), w) = \Lambda(\mathsf{hk}, \sigma).$$

- **Universality:** Π is said to be ε-universal if for any $\sigma \in \Sigma \setminus L$, if $\mathsf{hk} \leftarrow K$ it holds that

$$H_\infty\big(\Lambda\left(\mathsf{hk}, \sigma\right) \big| (\alpha(\mathsf{hk}), \sigma)\big) \geq \log(\varepsilon^{-1}).$$

Universality$_2$ and Smoothness. We also recall two stronger notions of security for projective hash proof systems, namely universality$_2$ and smoothness, as described in [CS02].

- **Universality$_2$**: A hash proof system $\Pi = (\Lambda, K, P, \Sigma, \Gamma, L)$ is said to be ε-universal$_2$ if for any $\sigma, \sigma^* \in \Sigma$ such that $\sigma \in \Sigma \setminus (L \cup \{\sigma^*\})$, if $\mathsf{hk} \leftarrow K$ it holds that

$$H_\infty\big(\Lambda\left(\mathsf{hk}, \sigma\right) \big| (\alpha(\mathsf{hk}), \sigma, \sigma^*, \Lambda\left(\mathsf{hk}, \sigma^*\right))\big) \geq \log(\varepsilon^{-1}).$$

- **Smoothness:** A hash proof system $\Pi = (\Lambda, K, P, \Sigma, \Gamma, L)$ is said to be smooth if for any $\sigma \in \Sigma \setminus L$, if $\mathsf{hk} \leftarrow K$ and $\gamma \leftarrow \Gamma$ it holds that

$$\big(\alpha(\mathsf{hk}), \sigma, \Lambda\left(\mathsf{hk}, \sigma\right)\big) \approx_s \big(\alpha(\mathsf{hk}), \sigma, \gamma\big).$$

We now show how to construct a universal hash proof system from any weak pseudorandom EGA.

Construction. Let $(G, X, \star)$ be a weak pseudorandom EGA and let $\ell = \omega(\log \lambda)$ be an integer. Additionally, let $\bar{x}_0 \leftarrow X$ and $\bar{x}_1 \leftarrow X$ be publicly available set elements. We define the input space Σ as

$$\Sigma = \Big\{(x_0, x_1) \in X^2 : \exists (g_0, g_1) \in G^2 \text{ s.t. } x_0 = g_0 \star \bar{x}_0,\ x_1 = g_1 \star \bar{x}_1\Big\}.$$

By the regularity of the group action, this is equivalent to defining $\Sigma = X^2$. We also define the subset $L \subset \Sigma$ as

$$L = \Big\{(x_0, x_1) \in X^2 : \exists g \in G \text{ s.t. } x_0 = g \star \bar{x}_0,\ x_1 = g \star \bar{x}_1\Big\},$$

where the group element g is the witness for membership in L. In addition, we let $\Gamma = X^\ell$ and $K = G^\ell \times \{0,1\}^\ell$, and we define the hash function $\Lambda : K \times \Sigma \to \Gamma$ to be

$$\Lambda\big((\mathbf{h}, \mathbf{b}), (x_0, x_1)\big) = (h_1 \star x_{b_1}, \ldots, h_\ell \star x_{b_\ell}),$$

where $\mathbf{h} = (h_1, \ldots, h_\ell)$ and $\mathbf{b} = (b_1, \ldots, b_\ell)$. We set the projection space to be $P = X^\ell$, and we define the projection function $\alpha : K \to P$ as

$$\alpha(\mathbf{h}, \mathbf{b}) = (h_1 \star \bar{x}_{b_1}, \ldots, h_\ell \star \bar{x}_{b_\ell}).$$

Subset Membership Problem. We state and prove the following lemma.

Lemma 8.4.2. *If $(G, X, \star)$ is a weak pseudorandom EGA, we have $\sigma_0 \overset{c}{\approx} \sigma_1$ where $\sigma_0 \leftarrow L$ and $\sigma_1 \leftarrow \Sigma$.*

Proof. By the weak pseudorandomness of group action we have

$$(\bar{x}_0, \bar{x}_1, g \star \bar{x}_0, g \star \bar{x}_1) \overset{c}{\approx} (\bar{x}_0, \bar{x}_1, x_0, x_1),$$

where $g \leftarrow G$ and $\bar{x}_1, x_0, x_1$ are all sampled uniformly and independently from X. It is easy to see that the "left" tuple corresponds to a uniformly sampled member $\sigma_0 \in L$ and the "right" tuple corresponds to a uniformly sampled element $\sigma_1 \in \Sigma$ (because the action is regular), as required.□

Projective Evaluation. We define $\mathsf{ProjEval}$ as

$$\mathsf{ProjEval}\big(\mathbf{y}, g\big) = (g \star y_1, \ldots, g \star y_\ell),$$

where $\mathbf{y} = (y_1, \ldots, y_\ell)$ and g is the witness. Let $(x_0, x_1) = (g \star \bar{x}_0, g \star \bar{x}_1)$ be a member of L with witness g, and let $\mathbf{y} = \alpha(\mathbf{h}, \mathbf{b})$ for some hash key $(\mathbf{h}, \mathbf{b}) \in K$. The algorithm $\mathsf{ProjEval}$ satisfies the projective evaluation property by observing that

$$\begin{aligned}
\mathsf{ProjEval}\big(\alpha(\mathbf{h}, \mathbf{b}), g\big) &= (g \star y_1, \ldots, g \star y_\ell) \\
&= (g \star (h_1 \star \bar{x}_{b_1}), \ldots, g \star (h_\ell \star \bar{x}_{b_\ell})) \\
&= (h_1 \star (g \star \bar{x}_{b_1}), \ldots, h_\ell \star (g \star \bar{x}_{b_\ell})) \\
&= (h_1 \star x_{b_1}, \ldots, h_\ell \star x_{b_\ell}) \\
&= \Lambda\big((\mathbf{h}, \mathbf{b}), (x_0, x_1)\big).
\end{aligned}$$

Universality. We prove the universality property as follows.

Lemma 8.4.3. *If $(G, X, \star)$ is a weak pseudorandom EGA, then the projective hash function is $2^{-\ell}$-universal.*

Proof. Let $(x_0, x_1) \in \Sigma \setminus L$ be an arbitrary non-member, and let $(\mathbf{h}, \mathbf{b}) \leftarrow K$ be a randomly chosen hash key. We need to show that

$$H_\infty\big(\Lambda((\mathbf{h}, \mathbf{b}), (x_0, x_1)) \big| \big(\bar{x}_0, \bar{x}_1, x_0, x_1, \alpha(\mathbf{h}, \mathbf{b})\big)\big) = \ell.$$

First, observe that there exists $g_0 \neq g_1$ such that $(x_0, x_1) = (g_0 \star \bar{x}_0, g_1 \star \bar{x}_1)$ because $(x_0, x_1) \notin L$. In addition, let $\mathbf{y} = \alpha(\mathbf{h}, \mathbf{b})$, i.e., for each $i \in [\ell]$ we have $y_i = h_i \star \bar{x}_{b_i}$. By the regularity of the group action, for each $i \in [\ell]$ there exists $d_{i,0} \in G$ and $d_{i,1} \in G$ such that

$$d_{i,0} \star \bar{x}_0 = d_{i,1} \star \bar{x}_1 = y_i.$$

In other words, given the tuple $(\bar{x}_0, \bar{x}_1, x_0, x_1, y_i)$, the bit b_i in the hash-key component (h_i, b_i) has full entropy. On the other hand, we have

$$h_i \star x_{b_i} = h_i \star (g_{b_i} \star \bar{x}_{b_i}) = g_{b_i} \star (h_i \star \bar{x}_{b_i}) = g_{b_i} \star y_i.$$

Since $g_0 \neq g_1$, it follows that given the tuple $(\bar{x}_0, \bar{x}_1, x_0, x_1, y_i)$, the set element $h_i \star x_{b_i} = g_{b_i} \star y_i$ has one bit of entropy (even in the view of a computationally unbounded adversary). By extending the same argument, we get

$$H_\infty\big(\{h_i \star x_{b_i}\}_{i \in [\ell]} \big| \big(\bar{x}_0, \bar{x}_1, x_0, x_1, \{y_i\}_{i \in [\ell]}\big)\big) = \ell,$$

as desired. This completes the proof of Lemma 8.4.3. □

The aforementioned lemmas yield the following theorem.

Theorem 8.4.4. *There exists a construction of a $2^{-\ell}$-universal projective hash function for any $\ell > 0$ from any weak pseudorandom EGA.*

Remark 8.4.5. Our construction and proof work in essentially the same way from a restricted EGA provided that we can sample group elements from a distribution that is *statistically* close to uniform over the group G while retaining the ability to efficiently compute the action. We note that this is plausibly the case with respect to the instantiation of restricted EGA from CSIDH and other similar isogeny-based assumptions (see [DG19] for more details).

Remark 8.4.6. In the aforementioned description of the HPS scheme, the hardness of the language membership problem crucially relies on the fact that the group element h such that $x_1 = h \star x_0$ is computationally hidden from the adversary. Note that most applications of HPS typically assume a

trusted setup. For applications that necessarily require an untrusted setup, our proposed HPS can still be used, albeit from a hashable EGA.

Universal$_2$ and Smooth Projective Hashing. Based on known reductions from Section 2.1 of [CS02], Theorem 8.4.4 implies the following corollary.

Corollary 8.4.7. *Let $(G, X, \star)$ be any weak pseudorandom EGA. Assuming the existence of an injective function $f : X^\ell \to \{0,1\}^m$ for some $m = \omega(\log \lambda)$ and the existence of a pairwise independent hash function $H : X^\ell \to \{0,1\}$ for some $\ell = \omega(\log \lambda)$, there exists a $2^{-\ell}$-universal$_2$ projective hash function and a smooth projective hash function, respectively.*

Further Applications. Smooth projective hashing implies chosen-ciphertext secure PKE [CS02], password authenticated key-exchange (PAKE) [GL03], privacy-preserving protocols [BPV12], and many other cryptographic primitives. Hence, our construction allows all of these primitives to be constructed from any weak pseudorandom (R)EGA.

8.5 Linear Hidden Shift (LHS) Assumption

In this section we introduce a hardness assumption called Linear Hidden Shift (LHS) problem and describe its cryptographic applications.

Notation. Unless stated otherwise, we use $+$ to denote the group operation, and we assume that e denotes the identity element of the group. For a binary vector $\mathbf{s} \in \{0,1\}^n$ and a group element $h \in G$, we use $h \cdot \mathbf{s}$ to denote a vector of group elements whose ith component is $s_i \cdot h$.

For a vector of group elements $\mathbf{g} \in G^n$ and a binary vector $\mathbf{s} \in \{0,1\}^n$, we use $\langle \mathbf{g}, \mathbf{s} \rangle$ to denote $s_1 \cdot g_1 + \cdots + s_n \cdot g_n$ where $+$ denotes the group operation (we remark that although the notation resembles an inner product, we do *not* necessarily have an inner product space).

Given a group action $\star : G \times X \to X$, the action naturally extends to the direct product group G^n for any positive integer n. So if $\mathbf{g} \in G^n$ and $\mathbf{x} \in X^n$ are two vectors of group elements and set elements respectively, we use $\mathbf{g} \star \mathbf{x}$ to denote a vector of set element whose ith component is $g_i \star x_i$.

Below, we formally state the search and decision versions of the assumption. Later, we show a simple *search to decision* reduction for the LHS assumption.

Definition 8.5.1. (Search Linear Hidden Shift) Let $\star : G \times X \to X$ be a regular group action, and let $n = \mathrm{poly}(\lambda)$ be a parameter. We say that (search) LHS problem is hard over $(G, X, \star)$ if for any polynomial $m = \mathrm{poly}(\lambda)$ and for any PPT attacker $\mathcal{A}$, we have

$$\Pr\left[\mathcal{A}\Big(\big\{(x_i, \mathbf{g}_i, (\langle \mathbf{g}_i, \mathbf{s}\rangle) \star x_i)\big\}_{i\in[m]}\Big) \text{ outputs } \mathbf{s}\right] \leq \mathrm{negl}(\lambda),$$

where $\mathbf{g}_i \leftarrow G^n$, $\mathbf{s} \leftarrow \{0,1\}^n$, $x_i \leftarrow X$ (all sampled independently), and the probability is taken over all random coins in the experiment.

Definition 8.5.2. (Decision Linear Hidden Shift) Let $\star : G \times X \to X$ be a group action, and let $n = \mathrm{poly}(\lambda)$ be a parameter. We say that LHS assumption holds over $(G, X, \star)$ if for any polynomial $m = \mathrm{poly}(\lambda)$ we have

$$\{(x_i, \mathbf{g}_i, (\langle \mathbf{g}_i, \mathbf{s}\rangle) \star x_i)\}_{i\in[m]} \quad \overset{c}{\approx} \quad \{(x_i, \mathbf{g}_i, u_i)\}_{i\in[m]},$$

where $\mathbf{g}_i \leftarrow G^n$, $\mathbf{s} \leftarrow \{0,1\}^n$, $x_i \leftarrow X$ and $u_i \leftarrow X$ (all sampled independently).

We naturally extend the notation $\langle \mathbf{g}, \mathbf{s}\rangle$ to matrices, i.e., for a matrix $\mathbf{M} \in G^{n\times \ell}$ and a binary vector $\mathbf{s} \in \{0,1\}^n$, we use $\mathbf{s}^t\mathbf{M}$ to denote a vector whose ith component is $\langle \mathbf{m}_i, \mathbf{s}\rangle$ where $\mathbf{m}_i$ is the ith column of $\mathbf{M}$.

Search to Decision Reduction. Using the notation described above the search LHS problem can be stated as the problem of recovering $\mathbf{s}$ given a tuple of the form $(\mathbf{x}, \mathbf{M}, \mathbf{Ms} \star \mathbf{x})$ where $\mathbf{x} \leftarrow X^n$ and $\mathbf{M} \leftarrow G^{m\times n}$. Similarly, the decision LHS problem states that

$$(\mathbf{x}, \mathbf{M}, \mathbf{Ms} \star \mathbf{x}) \quad \overset{c}{\approx} \quad (\mathbf{x}, \mathbf{M}, \mathbf{u}),$$

where $\mathbf{u} \leftarrow X^n$ and $m \gg n$. Now we show a simple search to decision reduction for LHS problem, which is similar to the reductions in [IN96, MM11] for (generalized) knapsack functions.

Lemma 8.5.3. (Search to Decision) *Let $\mathcal{A}$ be a distinguisher that distinguishes between LHS samples of the form $(\mathbf{x}, \mathbf{M}, \mathbf{Ms} \star \mathbf{x})$ and all-random tuple with probability $1 - \mathrm{negl}(\lambda)$. There exists a PPT attacker $\mathcal{A}'$ that recovers $\mathbf{s}$ from an instance of search LHS problem with probability $1 - \mathrm{negl}(\lambda)$.*

Proof. Given an instance of a search problem $(\mathbf{x}, \mathbf{M}, \mathbf{y})$ where $\mathbf{y} = \mathbf{Ms} \star \mathbf{x}$ for some (unknown) vector $\mathbf{s}$, the attacker $\mathcal{A}'$ does the following for each $i \in [n]$: it samples a column vector $\mathbf{r} \leftarrow G^m$, and

let $\mathbf{R}_i$ be a matrix whose ith column is $\mathbf{r}$ while *all other* columns are identical to the corresponding columns of $\mathbf{M}$ (so $\mathbf{R}_i$ and $\mathbf{M}$ only differ in the ith column). $\mathcal{A}'$ runs $\mathcal{A}$ on the tuple $(\mathbf{x}, \mathbf{R}_i, \mathbf{y})$. If $\mathcal{A}$ outputs "LHS samples," $\mathcal{A}'$ sets s_i to be zero. Otherwise, $\mathcal{A}'$ sets s_i to be 1.

Observe that if s_i were zero, then $(\mathbf{x}, \mathbf{R}_i, \mathbf{y})$ is distributed as LHS samples because $\mathbf{R}_i\mathbf{s} = \mathbf{M}\mathbf{s}$. On the other hand, if $s_i = 1$ then $(\mathbf{x}, \mathbf{R}_i, \mathbf{y})$ is a random tuple because the action is regular and hence the distribution of $\mathbf{R}_i\mathbf{s} \star \mathbf{x}$ is uniform and independent of $\mathbf{y}$. □

Remark 8.5.4. We note that the reduction above also works if the group action is *restricted* (where we can only evaluate the action of a set of small cardinality), provided that it is possible to sample a group element from a distribution that is statistically close to uniform.

8.5.1 Symmetric KDM-CPA Security from LHS

We describe a *symmetric* encryption scheme that satisfies KDM-CPA security (for projection functions) based on the LHS assumption. Let $\star : G \times X \to X$ be a group action such that LHS holds. We assume that all parties have access to a *public fixed* non-identity group element $h \in G$. Our construction of symmetric-key bit encryption $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ scheme is as follows:

- $\mathsf{Gen}(1^\lambda)$: To generate a secret key, sample a binary vector $\mathbf{s} \leftarrow \{0,1\}^n$.
- $\mathsf{Enc}(\mathbf{s}, b \in \{0,1\})$: Sample $\mathbf{g} \leftarrow G^n$, $x \leftarrow X$, and output

$$\mathsf{ct} = \big(x, \mathbf{g}, (b \cdot h + \langle \mathbf{g}, \mathbf{s} \rangle) \star x\big).$$

- $\mathsf{Dec}(\mathbf{s}, \mathsf{ct} = (x, \mathbf{g}, y))$: Output 0 if $y = \langle \mathbf{g}, \mathbf{s} \rangle \star x$, otherwise output 1.

Lemma 8.5.5. *The scheme Π above is CPA secure.*

Proof. We sketch a simple proof. Notice that a tuple of $m = \mathrm{poly}(\lambda)$ ciphertexts encrypting m (arbitrary) bits $\{b_i\}_{i\in[m]}$ in the scheme above has the form $\{x_i, \mathbf{g}_i, (b_i \cdot h) \star y_i\}_{i\in[m]}$ where $\{x_i, \mathbf{g}_i, y_i\}_{i\in[m]}$ are LHS samples. Therefore, by the LHS assumption we have

$$\{x_i, \mathbf{g}_i, (b_i \cdot h) \star y_i\}_{i\in[m]} \quad \stackrel{c}{\approx} \quad \{x_i, \mathbf{g}_i, (b_i \cdot h) \star u_i\}_{i\in[m]},$$

where each u_i is a random set element. It follows that encryptions of $\{b_i\}_{i\in[m]}$ are indistinguishable from a (truly) random tuple, as required. □

Lemma 8.5.6. *The scheme* Π *is KDM secure with respect to projection functions.*

Proof. Observe that encryptions of all bits of the secret key have the form $(\mathbf{x}, \mathbf{M}, (\mathbf{M}\mathbf{s} + h \cdot \mathbf{s}) \star \mathbf{x})$, where $\mathbf{x} \leftarrow X^n$, $\mathbf{M} \leftarrow G^{n \times n}$ and the action is applied componentwise. By a simple rearrangement we have

$$\big(\mathbf{x}, \mathbf{M}, (\mathbf{M}\mathbf{s} + h \cdot \mathbf{s}) \star \mathbf{x}\big) = \big(\mathbf{x}, \mathbf{M}, (\mathbf{M} + h \cdot \mathbf{I})\mathbf{s} \star \mathbf{x}\big).$$

Similarly, it is straightforward to see that encryptions of $\{1 - s_i\}_{i \in [n]}$ have the form

$$(\mathbf{x}', \mathbf{M}', (\mathbf{M}'\mathbf{s} + h \cdot (\mathbf{1} - \mathbf{s})) \star \mathbf{x}'),$$

where $\mathbf{1}$ is the all-one vector. By a simple rearrangement we have

$$\big(\mathbf{x}', \mathbf{M}', (\mathbf{M}'\mathbf{s} + h \cdot (\mathbf{1} - \mathbf{s})) \star \mathbf{x}'\big) = \big(\mathbf{x}', \mathbf{M}', [(\mathbf{M}' - h \cdot \mathbf{I})\mathbf{s} + h \cdot \mathbf{1}] \star \mathbf{x}'\big).$$

Clearly, if $\mathbf{M}$ (resp., $\mathbf{M}'$) is a uniform matrix, then $\mathbf{M}_1 := \mathbf{M} + h \cdot \mathbf{I}$ (resp., $\mathbf{M}_2 := \mathbf{M}' - h \cdot \mathbf{I}$) is also a uniform matrix. Given $2n$ samples of LHS challenges of the form $\{(\mathbf{x}_j, \mathbf{M}_j, \mathbf{y}_j)\}_{j \in [2]}$ where either $\{\mathbf{y}_j = \mathbf{M}_j\mathbf{s} \star \mathbf{x}_j\}_{j \in [2]}$ or $\{\mathbf{y}_j\}_{j \in [2]}$ are truly random vectors of set elements, the reduction simulates encryptions of projection functions of the secret key by computing $(\mathbf{x}_1, \mathbf{M}_1 - h \cdot \mathbf{I}, \mathbf{y}_1)$ and $(\mathbf{x}_2, \mathbf{M}_2 + h \cdot \mathbf{I}, (h \cdot \mathbf{1}) \star \mathbf{y}_2)$. By the LHS assumption it follows that

$$\big(\mathbf{x}, \mathbf{M}, (\mathbf{M} + h \cdot \mathbf{I})\mathbf{s} \star \mathbf{x}\big) \stackrel{c}{\approx} (\mathbf{x}, \mathbf{M}, \mathbf{u}),$$

$$(\mathbf{x}', \mathbf{M}', (\mathbf{M}'\mathbf{s} + h \cdot (\mathbf{1} - \mathbf{s})) \star \mathbf{x}') \stackrel{c}{\approx} (\mathbf{x}', \mathbf{M}', \mathbf{u}'),$$

where $\mathbf{u} \leftarrow X^n$ and $\mathbf{u}' \leftarrow X^n$ are uniform vectors of set elements. Therefore, encryptions of all projection functions of secret key are indistinguishable from tuples of truly random elements. On the other hand, by Lemma 8.5.5 we know that encryptions of zero are indistinguishable from truly random tuples. It follows that

$$\big(\{\mathsf{Enc}(\mathbf{s}, s_i)\}_{i \in [n]}, \{\mathsf{Enc}(\mathbf{s}, 1 - s_i)\}_{i \in [n]}\big) \quad \stackrel{c}{\approx} \quad \{\mathsf{Enc}(\mathbf{s}, 0)\}_{i \in [2n]},$$

as required. Indistinguishability of multiple encryptions of a projection function of the secret key from random tuples follows from a standard hybrid argument, and the proof is complete. □

Instatiation from Restricted EGA. Notice that the reduction above does *not* work in case of a *restricted* EGA because the relation lattice (i.e., the group structure) is not known. However, it is possible to show that an alternative version of the scheme described above is KDM-CPA secure

in case of a restricted EGA (for which the LHS assumption holds). Therefore, it is possible to realize symmetric KDM-CPA encryption from a *restricted* EGA provided that we can sample group elements from a distribution over the group G that is statistically close to uniform while retaining the ability to compute the action efficiently. Note that this is plausibly true for the restricted EGAs implied by CSIDH and other similar isogeny-based assumptions [DG19].

- $\mathsf{Gen}(1^\lambda)$: To generate a secret key, sample a binary vector $\mathbf{s} \leftarrow \{0,1\}^n$.
- $\mathsf{Enc}(\mathbf{s}, b \in \{0,1\})$: Sample $\mathbf{g} \leftarrow G^n$, $x \leftarrow X$, and $u \leftarrow X$. If $b = 0$, output the ciphertext $\mathsf{ct} = (x, \mathbf{g}, \langle \mathbf{g}, \mathbf{s} \rangle \star x)$. Otherwise, output $\mathsf{ct} = (x, \mathbf{g}, u)$.
- $\mathsf{Dec}(\mathbf{s}, \mathsf{ct} = (x, \mathbf{g}, y))$: Output 0 if $y = \langle \mathbf{g}, \mathbf{s} \rangle \star x$, otherwise output 1.

Lemma 8.5.7. *If $(G, X, \star)$ be a restricted EGA that satisfies the LHS assumption, the construction above is KDM-CPA secure.*

Proof. Observe that an encryption of 0 corresponds to an LHS sample while an encryption of 1 corresponds to a random tuple, so it is easy to see that the construction above is CPA secure based on the LHS assumption. The argument for KDM security is quite similar to the search to decision reduction for the LHS assumption (Lemma 8.5), and hence we omit the details. □

Implications. Using the general amplification of [App14], one can transform a symmetric-key KDM-secure scheme (with respect to projection functions) to a symmetric-key KDM-secure scheme with respect to circuits of a priori bounded size. Therefore, one can construct a symmetric-key KDM-secure scheme (with respect to bounded circuits) based on the LHS assumption. In a recent work, Lombardi *et al.* [LQR$^+$19] showed a construction of reusable designated-verifier NIZK (DV-NIZK) argument for NP assuming *any* PKE and a symmetric-key KDM-secure scheme. Hence, any PKE along with the LHS assumption implies reusable DV-NIZK arguments for NP.

In the same vein, Kitagawa and Matsuda [KM19] showed a construction of KDM-CCA PKE assuming PKE, DV-NIZK, and symmetric-key KDM security with respect to projection functions. Therefore, any PKE along with the LHS assumption implies KDM-CCA PKE.

Furthermore, Kitagawa *et al.* [KMT19] showed a construction of trapdoor function (with adaptive one-wayness) from a randomness-recovering symmetric-key KDM-secure scheme and a PKE scheme with pseudorandom ciphertexts. By plugging in their result, we obtain trapdoor function with adaptive one-wayness based the LHS assumption and any weak pseudorandom EGA.

8.5.2 On the LHS Assumption

In what follows we provide some insights on the security of the LHS assumption. We consider an additive variant of the LHS assumption, which we call it LHS(+), where $G = \mathbb{Z}_N^*$ and the product term $\mathbf{Ms}$ is computed by a *subset sum* over the columns of $\mathbf{M}$. We show that in this setting the LHS assumption is equivalent to the weak pseudorandomness for (effective) group actions provided that $\mathbf{M}$ is a structured matrix. We also describe an attack that breaks the search/decision LHS assumption in certain settings, and we explain how such attacks can be avoided.

LHS(+) Assumption. Let $(G, X, \star)$ be an EGA such that $G = \mathbb{Z}_N^*$ and $\varphi(N)/N \geq 1 - \mathrm{negl}(\lambda)$. Consider the following *additive* variant of the LHS assumption

$$(\mathbf{x}, \mathbf{M}, \mathbf{Ms} \star \mathbf{x}) \overset{c}{\approx} (\mathbf{x}, \mathbf{Ms}, \mathbf{u}),$$

where $\mathbf{Ms}$ is computed over $(\mathbb{Z}_N, +)$, i.e., subset sum over the columns of $\mathbf{M}$ modulo N. We show that if $\mathbf{M}$ is a structured "rank" 1 matrix (instead of a uniformly chosen matrix), the additive LHS assumption is equivalent to the weak pseudorandomness of the $(G, X, \star)$.

Let $\overline{\mathbf{M}} = \mathbf{a} \otimes \mathbf{b}$ where $\mathbf{a} \leftarrow \mathbb{Z}_N^m$ and $\mathbf{b} \leftarrow \mathbb{Z}_N^n$ are two randomly chosen vectors of group elements and $\otimes$ denotes the "tensor product" with respect to $\mathbb{Z}_N^*$. To put it differently, the ij^{th} entry of $\overline{\mathbf{M}}$ is equal to $a_i \cdot b_j$ where $\cdot$ denotes the multiplication modulo N. First, observe that $\overline{\mathbf{M}}\mathbf{s} = \mathbf{a} \otimes b^*$ where $b^* = \mathbf{b}^t\mathbf{s}$. In addition, if n is an integer such that $n > \log(N) + \omega(\log(\lambda))$, then by the leftover hash lemma b^* is distributed uniformly and independent of others. Furthermore, given any $\mathbf{M}$ with the aforementioned structure, one can compute two vectors $\mathbf{a}$ and $\mathbf{b}$ such that $\mathbf{M} = \mathbf{a} \otimes \mathbf{b}$. Consider the rows of LHS(+) assumption, which has the following form:

$$\begin{gathered}
(x_1, a_1 \otimes \mathbf{b}, (a_1 \cdot b^*) \star x_1), \\
(x_2, a_2 \otimes \mathbf{b}, (a_2 \cdot b^*) \star x_2), \\
\vdots \\
(x_m, a_m \otimes \mathbf{b}, (a_m \cdot b^*) \star x_m).
\end{gathered}$$

For each $i \in [m]$, compute $y_i = a_i \star x_i$. So, given an instance of the LHS(+) problem one can compute the following:

$$(y_1, b^* \star y_1), (y_2, b^* \star y_2), \ldots, (y_m, b^* \star y_m).$$

Therefore, LHS(+) assumption is equivalent to the weak pseudorandomness for effective group action in the aforementioned setting (the proof for the other direction is similar).

Attacks on LHS. To analyze the quantum security of LHS assumption, it is reasonable to assume that discrete logarithms are easy in the group G. Then, the LHS problem becomes essentially a linear algebra one. For example, if G is cyclic of order q, we can rewrite all elements of G as their discrete log to a fixed basis $\mathbf{g}$, the subset product $\langle \mathbf{g}, \mathbf{s} \rangle$ becomes the standard inner product over $(\mathbb{Z}_q)^n$, and LHS becomes similar to LWE [Reg05], with the main difference that the algebraic structure is hidden by the group action, rather than by noise.

It is then evident that both decision and search LHS can be solved by breaking the one-wayness of the group action, recovering a list of tuples $(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle)$, and then using linear algebra over $\mathbb{Z}_q$. The same blueprint also applies to non-cyclic groups G. To the best of our knowledge, this is the most efficient generic attack on LHS in the post-quantum setting.

However, some instantiations may offer easier paths to attack LHS: isogenies are an interesting example. The recent work of Castryck, Sotáková and Vercauteren [CSV20] shows that some instantiations of group actions from isogenies are not pseudorandom EGAs. While it is not evident how breaking pseudorandomness could help solve LHS, their technique is actually more powerful. Indeed, it provides an efficient algorithm to compute some quadratic characters of the group G, directly on its isomorphic representation on X. More precisely, for a fixed quadratic character χ of the class group $\mathrm{Cl}(\mathcal{O})$, on input a pair $(x, y) \in X^2$ such that $y = g \star x$, their algorithm outputs $\chi(g) = \pm 1$.

We can use this algorithm to solve LHS as follows. Define $f : G \to \{0, 1\}$ as $f = (1 - \chi)/2$. For any tuple $(x_i, \mathbf{g}_i = (g_i^{(1)}, \ldots, g_i^{(n)}), \langle \mathbf{g}_i, \mathbf{s} \rangle \star x_i)$ we compute the following

$$\big(f(g_i^{(1)}), \ldots, f(g_i^{(n)}), f(\langle \mathbf{g}_i, \mathbf{s} \rangle)\big).$$

After we collect enough tuples, we obtain a linear system over $\mathbb{Z}_2$, which we solve to recover $\mathbf{s}$. This is analogous to the attack on the discrete logarithm equivalent of LHS using Legendre symbols, and applies to any other group action where the group G has low order characters which can be "read" on X.

Castryck *et al.*'s attack does not apply against CSIDH, because the class group associated to it has no quadratic characters. Even for instantiations where class groups do have quadratic order characters, e.g., isogeny schemes based on ordinary elliptic curves, it is easy to block the attack by restricting to the subgroup of squares inside $\mathrm{Cl}(\mathcal{O})$.

Bibliography

[ADMP20] N. Alamati, L. De Feo, H. Montgomery, and S. Patranabis. Cryptographic group actions and applications. In *ASIACRYPT (to Appear)*. 2020.

[AKC+17] R. Azarderakhsh, B. Koziel, M. Campagna, B. LaMacchia, C. Costello, P. Longa, L. De Feo, M. Naehrig, B. Hess, J. Renes, A. Jalali, V. Soukharev, D. Jao, and D. Urbanik. Supersingular isogeny key encapsulation, 2017.

[AMP19] N. Alamati, H. Montgomery, and S. Patranabis. Symmetric primitives with structured secrets. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2019, Part I*, LNCS, pages 650–679. Springer, Heidelberg, August 2019.

[AMPR19] N. Alamati, H. Montgomery, S. Patranabis, and A. Roy. Minicrypt primitives with algebraic structure and applications. In V. Rijmen and Y. Ishai, editors, *EUROCRYPT 2019, Part II*, LNCS, pages 55–82. Springer, Heidelberg, May 2019.

[App14] B. Applebaum. Key-dependent message security: Generic amplification and completeness. *Journal of Cryptology*, 27(3):429–451, July 2014.

[Bar17] B. Barak. The complexity of public-key cryptography. In *Tutorials on the Foundations of Cryptography*, pages 45–77. 2017.

[BBS04] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Heidelberg, August 2004.

[BD18] Z. Brakerski and N. Döttling. Two-message statistically sender-private OT from LWE. In A. Beimel and S. Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 370–390. Springer, Heidelberg, November 2018.

[BF01] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001.

[BFP+15] A. Banerjee, G. Fuchsbauer, C. Peikert, K. Pietrzak, and S. Stevens. Key-homomorphic constrained pseudorandom functions. In Y. Dodis and J. B. Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 31–60. Springer, Heidelberg, March 2015.

[BGI+17] S. Badrinarayanan, S. Garg, Y. Ishai, A. Sahai, and A. Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 275–303. Springer, Heidelberg, December 2017.

[BHY09] M. Bellare, D. Hofheinz, and S. Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 1–35. Springer, Heidelberg, April 2009.

[BIJ18] J.-F. Biasse, A. Iezzi, and M. J. Jacobson Jr. A note on the security of CSIDH. In *INDOCRYPT 2018*, LNCS, pages 153–168. Springer, Heidelberg, 2018.

[BKV19] W. Beullens, T. Kleinjung, and F. Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In *ASIACRYPT 2019, Part I*, LNCS, pages 227–247. Springer, Heidelberg, December 2019.

[BL18] F. Benhamouda and H. Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 500–532. Springer, Heidelberg, April / May 2018.

[BLMR13] D. Boneh, K. Lewi, H. W. Montgomery, and A. Raghunathan. Key homomorphic PRFs and their applications. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428. Springer, Heidelberg, August 2013.

[BLSV18] Z. Brakerski, A. Lombardi, G. Segev, and V. Vaikuntanathan. Anonymous IBE, leakage resilience and circular security from new assumptions. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 535–564. Springer, Heidelberg, April / May 2018.

[BM82] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo random bits. In *23rd FOCS*, pages 112–117. IEEE Computer Society Press, November 1982.

[BOB18] P. Barreto, G. Oliveira, and W. Benits. Supersingular isogeny oblivious transfer. Cryptology ePrint Archive, Report 2018/459, 2018. `https://eprint.iacr.org/2018/459`.

[BP14] A. Banerjee and C. Peikert. New and improved key-homomorphic pseudorandom functions. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 353–370. Springer, Heidelberg, August 2014.

[BPR12] A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom functions and lattices. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Heidelberg, April 2012.

[BPV12] O. Blazy, D. Pointcheval, and D. Vergnaud. Round-optimal privacy-preserving protocols with smooth projective hash functions. In R. Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 94–111. Springer, Heidelberg, March 2012.

[BS20] X. Bonnetain and A. Schrottenloher. Quantum security analysis of CSIDH. In V. Rijmen and Y. Ishai, editors, *EUROCRYPT 2020, Part II*, LNCS, pages 493–522. Springer, Heidelberg, May 2020.

[BV96] D. Boneh and R. Venkatesan. Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In N. Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 129–142. Springer, Heidelberg, August 1996.

[BV15] Z. Brakerski and V. Vaikuntanathan. Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In Y. Dodis and J. B. Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 1–30. Springer, Heidelberg, March 2015.

[BY91] G. Brassard and M. Yung. One-way group actions. In A. J. Menezes and S. A. Vanstone, editors, *CRYPTO'90*, volume 537 of *LNCS*, pages 94–107. Springer, Heidelberg, August 1991.

[Can01] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

[CCG+19] A. R. Choudhuri, M. Ciampi, V. Goyal, A. Jain, and R. Ostrovsky. Round optimal secure multiparty computation from minimal assumptions. Cryptology ePrint Archive, Report 2019/216, 2019. `https://eprint.iacr.org/2019/216`.

[CGGM00] R. Canetti, O. Goldreich, S. Goldwasser, and S. Micali. Resettable zero-knowledge (extended abstract). In *32nd ACM STOC*, pages 235–244. ACM Press, May 2000.

[Chi17] A. M. Childs. Lecture notes on quantum algorithms, 2017. `https://www.cs.umd.edu/~amchilds/qa/qa.pdf`.

[CJS14] A. Childs, D. Jao, and V. Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *Journal of Mathematical Cryptology*, 8(1):1–29, 2014.

[CLG09] D. X. Charles, K. E. Lauter, and E. Z. Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22(1):93–113, January 2009.

[CLM+18] W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes. CSIDH: An efficient post-quantum commutative group action. In T. Peyrin and S. Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Heidelberg, December 2018.

[CM01] K. K. H. Cheung and M. Mosca. Decomposing finite abelian groups. *Quantum Information & Computation*, 1(3):26–32, 2001.

[CMS99] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In J. Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 402–414. Springer, Heidelberg, May 1999.

[Cou06] J.-M. Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. `http://eprint.iacr.org/2006/291`.

[CS02] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In L. R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, Heidelberg, April / May 2002.

[CSV20] W. Castryck, J. Sotáková, and F. Vercauteren. Breaking the decisional Diffie-Hellman problem for class group actions using genus theory. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2020, Part II*, LNCS, pages 92–120. Springer, Heidelberg, August 2020.

[De 17] L. De Feo. Mathematics of isogeny based cryptography, 2017.

[DG17a] N. Döttling and S. Garg. From selective IBE to full IBE and selective HIBE. In Y. Kalai and L. Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 372–408. Springer, Heidelberg, November 2017.

[DG17b] N. Döttling and S. Garg. Identity-based encryption from the Diffie-Hellman assumption. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 537–569. Springer, Heidelberg, August 2017.

[DG19] L. De Feo and S. D. Galbraith. SeaSign: Compact isogeny signatures from class group actions. In V. Rijmen and Y. Ishai, editors, *EUROCRYPT 2019, Part III*, LNCS, pages 759–789. Springer, Heidelberg, May 2019.

[DGHM18] N. Döttling, S. Garg, M. Hajiabadi, and D. Masny. New constructions of identity-based and key-dependent message secure encryption schemes. In M. Abdalla and R. Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 3–31. Springer, Heidelberg, March 2018.

[DHP+18] Y. Doröz, J. Hoffstein, J. Pipher, J. H. Silverman, B. Sunar, W. Whyte, and Z. Zhang. Fully homomorphic encryption from the finite field isomorphism problem. In M. Abdalla and R. Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 125–155. Springer, Heidelberg, March 2018.

[DJP14] L. De Feo, D. Jao, and J. Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.

[DKS18] L. De Feo, J. Kieffer, and B. Smith. Towards practical key exchange from ordinary isogeny graphs. Cryptology ePrint Archive, Report 2018/485, 2018. `https://eprint.iacr.org/2018/485`.

[DM20] L. De Feo and M. Meyer. Threshold schemes from isogeny assumptions. In *PKC 2020, Part II*, LNCS, pages 187–212. Springer, Heidelberg, 2020.

[DMPS19] L. De Feo, S. Masson, C. Petit, and A. Sanso. Verifiable delay functions from supersingular isogenies and pairings. In *ASIACRYPT 2019, Part I*, LNCS, pages 248–277. Springer, Heidelberg, December 2019.

[dOPS18] C. D. de Saint Guilhem, E. Orsini, C. Petit, and N. P. Smart. Secure oblivious transfer from semi-commutative masking. Cryptology ePrint Archive, Report 2018/648, 2018. `https://eprint.iacr.org/2018/648`.

[EHK+13] A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An algebraic framework for Diffie-Hellman assumptions. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, August 2013.

[ElG84] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and D. Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 10–18. Springer, Heidelberg, August 1984.

[EPRS17] A. Everspaugh, K. G. Paterson, T. Ristenpart, and S. Scott. Key rotation for authenticated encryption. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 98–129. Springer, Heidelberg, August 2017.

[FMV19] D. Friolo, D. Masny, and D. Venturi. A black-box construction of fully-simulatable, round-optimal oblivious transfer from strongly uniform key agreement. In *TCC 2019, Part I*, LNCS, pages 111–130. Springer, Heidelberg, March 2019.

[Gal99] S. D. Galbraith. Constructing isogenies between elliptic curves over finite fields. *LMS Journal of Computation and Mathematics*, 2:118–138, 1999.

[GGH19] S. Garg, R. Gay, and M. Hajiabadi. New techniques for efficient trapdoor functions and applications. In V. Rijmen and Y. Ishai, editors, *EUROCRYPT 2019, Part III*, LNCS, pages 33–63. Springer, Heidelberg, May 2019.

[GGM84] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984.

[GH18] S. Garg and M. Hajiabadi. Trapdoor functions from the computational Diffie-Hellman assumption. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 362–391. Springer, Heidelberg, August 2018.

[GHMM18] S. Garg, M. Hajiabadi, M. Mahmoody, and A. Mohammed. Limits on the power of garbling techniques for public-key encryption. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 335–364. Springer, Heidelberg, August 2018.

[GHS02] S. D. Galbraith, F. Hess, and N. P. Smart. Extending the GHS Weil descent attack. In L. R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 29–44. Springer, Heidelberg, April / May 2002.

[GJJM20] V. Goyal, A. Jain, Z. Jin, and G. Malavolta. Statistical zaps and new oblivious transfer protocols. In V. Rijmen and Y. Ishai, editors, *EUROCRYPT 2020, Part III*, LNCS, pages 668–699. Springer, Heidelberg, May 2020.

[GL03] R. Gennaro and Y. Lindell. A framework for password-based authenticated key exchange. In E. Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 524–543. Springer, Heidelberg, May 2003. `http://eprint.iacr.org/2003/032.ps.gz`.

[GM82] S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th ACM STOC*, pages 365–377. ACM Press, May 1982.

[Gol01] O. Goldreich. *Foundations of Cryptography*, volume I. Cambridge University Press, 2001.

[Gol04] O. Goldreich. *Foundations of Cryptography*, volume II. Cambridge University Press, 2004.

[GPS08] S. D. Galbraith, K. G. Paterson, and N. P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113 – 3121, 2008. ISSN 0166-218X. Applications of Algebra to Cryptography.

[GPS17] S. D. Galbraith, C. Petit, and J. Silva. Identification protocols and signature schemes based on supersingular isogeny problems. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 3–33. Springer, Heidelberg, December 2017.

[GPSV18] S. Galbraith, L. Panny, B. Smith, and F. Vercauteren. Quantum equivalence of the DLP and CDHP for group actions. Cryptology ePrint Archive, Report 2018/1199, 2018. `https://eprint.iacr.org/2018/1199`.

[GR02] M. Goldmann and A. Russell. The complexity of solving equations over finite groups. *Inf. Comput.*, 178(1):253–262, 2002.

[GR05] C. Gentry and Z. Ramzan. Single-database private information retrieval with constant communication rate. In L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *ICALP 2005*, volume 3580 of *LNCS*, pages 803–815. Springer, Heidelberg, July 2005.

[GS10] D. Grigoriev and V. Shpilrain. Authentication schemes from actions on graphs, groups, or rings. *Annals of Pure and Applied Logic*, 162(3):194–200, 2010.

[GS13] S. D. Galbraith and A. Stolbunov. Improved algorithm for the isogeny problem for ordinary elliptic curves. *Applicable Algebra in Engineering, Communication and Computing*, 24(2):107–131, June 2013. ISSN 1432-0622.

[GS18] S. Garg and A. Srinivasan. Two-round multiparty secure computation from minimal assumptions. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 468–499. Springer, Heidelberg, April / May 2018.

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

[HK07] D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 553–571. Springer, Heidelberg, August 2007.

[HLOV11] B. Hemenway, B. Libert, R. Ostrovsky, and D. Vergnaud. Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. In D. H. Lee and X. Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 70–88. Springer, Heidelberg, December 2011.

[How01] N. Howgrave-Graham. Approximate integer common divisors. In *Cryptography and Lattices, International Conference (CaLC)*, pages 51–66. 2001.

[IKO05] Y. Ishai, E. Kushilevitz, and R. Ostrovsky. Sufficient conditions for collision-resistant hashing. In J. Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 445–456. Springer, Heidelberg, February 2005.

[Imp95] R. Impagliazzo. A personal view of average-case complexity. In *Proceedings of Structure in Complexity Theory. Tenth Annual IEEE Conference*, pages 134–147. June 1995. ISSN 1063-6870.

[IN96] R. Impagliazzo and M. Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of Cryptology*, 9(4):199–216, September 1996.

[IR89] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *21st ACM STOC*, pages 44–61. ACM Press, May 1989.

[JD11] D. Jao and L. De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In B.-Y. Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 19–34. Springer, Heidelberg, November / December 2011.

[JKKR17] A. Jain, Y. T. Kalai, D. Khurana, and R. Rothblum. Distinguisher-dependent simulation in two rounds and its applications. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 158–189. Springer, Heidelberg, August 2017.

[JQSY19] Z. Ji, Y. Qiao, F. Song, and A. Yun. General linear group action on tensors: A candidate for post-quantum cryptography. In *TCC 2019, Part I*, LNCS, pages 251–281. Springer, Heidelberg, March 2019.

[KKS18] Y. T. Kalai, D. Khurana, and A. Sahai. Statistical witness indistinguishability (and more) in two messages. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 34–65. Springer, Heidelberg, April / May 2018.

[KL14] J. Katz and Y. Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, second edition, November 2014.

[KM19] F. Kitagawa and T. Matsuda. CPA-to-CCA transformation for KDM security. In *TCC 2019, Part II*, LNCS, pages 118–148. Springer, Heidelberg, March 2019.

[KMT19] F. Kitagawa, T. Matsuda, and K. Tanaka. CCA security and trapdoor functions via key-dependent-message security. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2019, Part III*, LNCS, pages 33–64. Springer, Heidelberg, August 2019.

[KO97] E. Kushilevitz and R. Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th FOCS*, pages 364–373. IEEE Computer Society Press, October 1997.

[KO00] E. Kushilevitz and R. Ostrovsky. One-way trapdoor permutations are sufficient for non-trivial single-server private information retrieval. In B. Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 104–121. Springer, Heidelberg, May 2000.

[KR00] H. Krawczyk and T. Rabin. Chameleon signatures. In *NDSS 2000*. The Internet Society, February 2000.

[KS17] D. Khurana and A. Sahai. How to achieve non-malleability in one or two rounds. In *58th FOCS*, pages 564–575. IEEE Computer Society Press, 2017.

[Kup05] G. Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal of Computing*, 35(1):170–188, 2005.

[Kup13] G. Kuperberg. Another Subexponential-time Quantum Algorithm for the Dihedral Hidden Subgroup Problem. In *8th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2013)*. 2013.

[KW19] V. Koppula and B. Waters. Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2019, Part II*, LNCS, pages 671–700. Springer, Heidelberg, August 2019.

[LMR14] K. Lewi, H. W. Montgomery, and A. Raghunathan. Improved constructions of PRFs secure against related-key attacks. In I. Boureanu, P. Owesarski, and S. Vaudenay, editors, *ACNS 14*, volume 8479 of *LNCS*, pages 44–61. Springer, Heidelberg, June 2014.

[LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Heidelberg, May / June 2010.

[LQR+19] A. Lombardi, W. Quach, R. D. Rothblum, D. Wichs, and D. J. Wu. New constructions of reusable designated-verifier NIZKs. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2019, Part III*, LNCS, pages 670–700. Springer, Heidelberg, August 2019.

[LT18] A. Lehmann and B. Tackmann. Updatable encryption with post-compromise security. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 685–716. Springer, Heidelberg, April / May 2018.

[Lyu09] V. Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In M. Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

[Mau09] U. M. Maurer. Unifying zero-knowledge proofs of knowledge. In B. Preneel, editor, *AFRICACRYPT 09*, volume 5580 of *LNCS*, pages 272–286. Springer, Heidelberg, June 2009.

[MM11] D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 465–484. Springer, Heidelberg, August 2011.

[Mon18] H. Montgomery. More efficient lattice PRFs from keyed pseudorandom synthesizers. In *INDOCRYPT 2018*, LNCS, pages 190–211. Springer, Heidelberg, 2018.

[MR18] M. Meyer and S. Reith. A faster way to the CSIDH. In *INDOCRYPT 2018*, LNCS, pages 137–152. Springer, Heidelberg, 2018.

[NOTT20] K. Nakagawa, H. Onuki, A. Takayasu, and T. Takagi. l_1-norm ball for CSIDH: optimal strategy for choosing the secret key space. Cryptology ePrint Archive, Report 2020/181, 2020.

[NP01] M. Naor and B. Pinkas. Efficient oblivious transfer protocols. In S. R. Kosaraju, editor, *12th SODA*, pages 448–457. ACM-SIAM, January 2001.

[NPR99] M. Naor, B. Pinkas, and O. Reingold. Distributed pseudo-random functions and KDCs. In J. Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 327–346. Springer, Heidelberg, May 1999.

[NR97] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467. IEEE Computer Society Press, October 1997.

[OK93] W. Ogata and K. Kurosawa. On claw free families. In H. Imai, R. L. Rivest, and T. Matsumoto, editors, *ASIACRYPT'91*, volume 739 of *LNCS*, pages 111–123. Springer, Heidelberg, November 1993.

[Pai99] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238. Springer, Heidelberg, May 1999.

[Pei20] C. Peikert. He gives C-sieves on the CSIDH. In V. Rijmen and Y. Ishai, editors, *EUROCRYPT 2020, Part II*, LNCS, pages 463–492. Springer, Heidelberg, May 2020.

[Poh69] I. Pohl. Bidirectional and heuristic search in path problems. Technical Report 104, Stanford LinearAccelerator Center, Stanford, California, 1969.

[PS00] D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000.

[PVW08] C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, Heidelberg, August 2008.

[PW08] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In R. E. Ladner and C. Dwork, editors, *40th ACM STOC*, pages 187–196. ACM Press, May 2008.

[Reg04] O. Regev. A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space. arXiv:quant-ph/0406151, June 2004.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In H. N. Gabow and R. Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[Rom90] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd ACM STOC*, pages 387–394. ACM Press, May 1990.

[RS06] A. Rostovtsev and A. Stolbunov. Public-Key Cryptosystem Based On Isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. http://eprint.iacr.org/2006/145.

[Sha07] H. Shacham. A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074, 2007. http://eprint.iacr.org/2007/074.

[Sho97] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[Sim98] D. R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In K. Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 334–345. Springer, Heidelberg, May / June 1998.

[Sto09] A. Stolbunov. Reductionist security arguments for Public-Key cryptographic schemes based on group action. In S. F. Mjølsnes, editor, *Norsk informasjonssikkerhetskonferanse (NISK)*. 2009.

[Sto10] A. Stolbunov. Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves. *Advances in Mathematics of Communications*, 4(2), 2010.

[Sto12] A. Stolbunov. Cryptographic schemes based on isogenies, 2012.

[Tes06] E. Teske. An elliptic curve trapdoor system. *Journal of Cryptology*, 19(1):115–133, January 2006.

[Vit19] V. Vitse. Simple oblivious transfer protocols compatible with supersingular isogenies. In *AFRICACRYPT 19*, LNCS, pages 56–78. Springer, Heidelberg, 2019.

[YAJ+17] Y. Yoo, R. Azarderakhsh, A. Jalali, D. Jao, and V. Soukharev. A post-quantum digital signature scheme based on supersingular isogenies. In A. Kiayias, editor, *FC 2017*, volume 10322 of *LNCS*, pages 163–181. Springer, Heidelberg, April 2017.