Public-Key Cryptosystems Resilient to Key Leakage

Moni Naor[*] Gil Segev[†]

Abstract

Most of the work in the analysis of cryptographic schemes is concentrated in abstract adversarial models that do not capture *side-channel attacks*. Such attacks exploit various forms of unintended information leakage, which is inherent to almost all physical implementations. Inspired by recent side-channel attacks, especially the "cold boot attacks" of Halderman et al. (USENIX Security '08), Akavia, Goldwasser and Vaikuntanathan (TCC '09) formalized a realistic framework for modeling the security of encryption schemes against a wide class of side-channel attacks in which adversarially chosen functions of the secret key are leaked. In the setting of public-key encryption, Akavia et al. showed that Regev's lattice-based scheme (STOC '05) is resilient to any leakage of $L/\text{polylog}(L)$ bits, where L is the length of the secret key.

In this paper we revisit the above-mentioned framework and our main results are as follows:

- We present a generic construction of a public-key encryption scheme that is resilient to key leakage from any *universal hash proof system*. The construction does not rely on additional computational assumptions, and the resulting scheme is as efficient as the underlying hash proof system. Existing constructions of hash proof systems imply that our construction can be based on a variety of number-theoretic assumptions, including the decisional Diffie-Hellman assumption (and its progressively weaker d-Linear variants), the quadratic residuosity assumption, and Paillier's composite residuosity assumption.
- We construct a new hash proof system based on the decisional Diffie-Hellman assumption (and its d-Linear variants), and show that the resulting scheme is resilient to any leakage of $L(1-o(1))$ bits. In addition, we prove that the recent scheme of Boneh et al. (CRYPTO '08), constructed to be a "circular-secure" encryption scheme, fits our generic approach and is also resilient to any leakage of $L(1-o(1))$ bits.
- We extend the framework of key leakage to the setting of chosen-ciphertext attacks. On the theoretical side, we prove that the Naor-Yung paradigm is applicable in this setting as well, and obtain as a corollary encryption schemes that are CCA2-secure with any leakage of $L(1-o(1))$ bits. On the practical side, we prove that variants of the Cramer-Shoup cryptosystem (along the lines of our generic construction) are CCA1-secure with any leakage of $L/4$ bits, and CCA2-secure with any leakage of $L/6$ bits.

A preliminary version of this work appeared in *Advances in Cryptology – CRYPTO '09*, pages 18–35, 2009.

[*] Incumbent of the Judith Kleeman Professorial Chair, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel. Email: `moni.naor@weizmann.ac.il`. Research supported in part by a grant from the Israel Science Foundation.

[†] Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel. Email: `gil.segev@weizmann.ac.il`. Research supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities, and by a grant from the Israel Science Foundation.

Contents

1 Introduction 2

1.1 Overview of Our Results . . . 3

1.2 Related Work . . . 5

1.3 Paper Organization . . . 6

2 Preliminaries, Assumptions, and Tools 6

2.1 Computational Assumptions . . . 6

2.2 Randomness Extraction . . . 7

2.3 Hash Proof Systems . . . 8

3 Defining Key-Leakage Attacks 9

3.1 Chosen-Plaintext Key-Leakage Attacks . . . 10

3.2 Chosen-Ciphertext Key-Leakage Attacks . . . 10

3.3 Weak Key-Leakage Attacks . . . 11

4 A Generic Construction from Hash Proof Systems 12

4.1 The Construction . . . 12

4.2 Example: A DDH-Based Instantiation . . . 14

5 Improved Resilience Based on DDH and *d*-Linear 14

5.1 Proposal 1: A New Hash Proof System . . . 15

5.2 Proposal 2: The BHHO Scheme . . . 17

6 Protecting Against Chosen-Ciphertext Key-Leakage Attacks 18

6.1 A Generic Construction . . . 18

6.2 An Efficient CCA1-Secure Scheme . . . 22

6.3 An Efficient CCA2-Secure Scheme . . . 25

7 Protecting Against Weak Key-Leakage Attacks 30

8 Protecting Against Generalized Forms of Key-Leakage Attacks 31

8.1 Noisy Leakage . . . 31

8.2 Leakage of Intermediate Values from the Key-Generation Process . . . 32

8.3 Keys Generated using Weak Random Sources . . . 33

8.4 Leakage of Intermediate Values from the Decryption Process . . . 34

A The Matrix *d*-Linear Assumption 38

1 Introduction

Proving the security of a cryptographic scheme consists of two main ingredients: (1) an *adversarial model* specifying the adversarial access to the system and the adversary's computational capabilities, and (2) a *notion of security* specifying what it means to "break" the security of the scheme. Whereas notions of security have significantly evolved over the years (following the seminal work of Goldwasser and Micali [19]), the vast majority of cryptographic schemes are still analyzed in the "standard" adversarial model. This is an abstract model in which the parties are viewed as interactive Turing machines, each party has its own secret memory, private source of random bits, and so on. In this model the adversarial access is well defined, and although the model seems rather limited, it nevertheless captures many realistic attacks. In real life, however, almost any physical implementation leaks additional information. Any such information that is not captured by the standard model is referred to as a *side channel*; typical examples include electromagnetic measurements, detection of internal faults, timing information, and many more. Over the years side-channel attacks exposed crucial vulnerabilities of schemes that are considered secure in the standard model (see, for example, [3, 5, 31, 32]).

Countermeasures for protecting against side-channel attacks follow two complementing approaches. The first approach is to make the physical world similar to the standard model by preventing unintended leakage of information (e.g., building "tamper-proof" devices, minimizing electromagnetic noise, and more). This approach is typically rather inefficient and expensive, and is even impossible in some cases. The second approach is to design systems that are robust to modifications of the standard model. This approach mainly includes protecting against side-channel attacks by modeling them, as much as possible, using abstract notions of computation.

Physically observable cryptography. In their pioneering work, Micali and Reyzin [36] put forward a comprehensive framework for modeling security against side-channel attacks. Their framework captures any such attack in which leakage of information occurs as a result of *computation*. The framework relies on the basic assumption that *computation and only computation leaks information*, that is, there is no leakage of information in the absence of computation. This assumption has led to the construction of various cryptographic primitives that are robust to "computational" leakage (see, for example, [16, 20, 36, 38, 39]).

Key-leakage attacks. Recently, Halderman et al. [22] presented a suite of attacks that violate the basic assumption underlying the framework of Micali and Reyzin. Halderman et al. showed that, contrary to popular assumptions, a computer's memory is not erased when it loses power. They demonstrated that ordinary DRAMs typically lose their contents gradually over a period of seconds, and that residual data can be recovered using simple, non-destructive techniques that require only momentary physical access to the machine. Halderman et al. presented attacks that exploit DRAM remanence effects to recover cryptographic keys held in memory. Specifically, their "cold boot" attacks showed that a significant fraction of the bits of a cryptographic key can be recovered if the key is ever stored in memory. Halderman et al. managed to completely compromise the security of several popular disk encryption systems (including BitLocker, TrueCrypt, and FileVault), and to reconstruct DES, AES, and RSA keys (see also the improvements and further analysis of Heninger and Shacham [24]).

Inspired by the cold boot attacks, Akavia, Goldwasser and Vaikuntanathan [2] formalized a general framework for modeling "memory attacks" in which adversarially chosen functions of the secret key are leaked in an adaptive fashion, with the only restriction that the total amount of leakage is bounded. Akavia et al. showed that the lattice-based public-key encryption scheme of Regev [40] and the identity-based encryption of Gentry, Peikert, and Vaikuntanathan [17] are resilient to such bounded key leakage.

Our contributions. In this work we revisit the framework of key-leakage attacks introduced by Akavia et al. in the setting of public-key encryption. We present a generic construction of a public-key encryption scheme that is resilient to key leakage, and show that the construction can be based on a variety of number-theoretic assumptions. Moreover, we demonstrate that our approach leads to encryption schemes that are both resilient to significantly large amounts of leakage, and that are efficient and can be used in practice (see, in particular, the instantiation in Section 4.2 that is based on the decisional Diffie-Hellman assumption). In addition, we extend the framework of key-leakage attacks to the setting of chosen-ciphertext security. We present both a generic transformation from chosen-plaintext security to chosen-ciphertext security in the context of key-leakage attacks, and efficient schemes that are based on specific number-theoretic assumptions.

1.1 Overview of Our Results

In what follows we present a more elaborated exposition of our results, but first, we briefly describe the framework introduced by Akavia et al. and their results. Informally, an encryption scheme is resilient to key-leakage attacks if it is semantically secure even when the adversary obtains sensitive leakage information. This is modeled by providing the adversary with access to a leakage oracle: the adversary can submit any function f and receive $f(sk)$, where sk is the secret key (we note that the leakage functions can be chosen depending on the public key, which is known to the adversary). The adversary can query the leakage oracle adaptively, with only one restriction: the sum of output lengths of all the leakage functions has to be bounded by a predetermined parameter λ (clearly, λ has to be less than the length of the secret key)[1]. A formal definition is provided in Section 3. In the setting of public-key encryption, Akavia et al. showed that Regev's public-key encryption scheme [40] is resilient to any key leakage of $L/\text{polylog}(L)$ bits, where L is the length of the secret key (see improvements to the allowed amount of leakage in the full version of their paper). We are now ready to state our results more clearly:

A generic construction. We present a generic construction of a public-key encryption scheme that is resilient to key leakage from any *universal hash proof system*, a very useful primitive introduced by Cramer and Shoup [8] for protecting against chosen-ciphertext attacks. The construction does not rely on additional computational assumptions, and the resulting scheme is as efficient as the underlying hash proof system. Existing constructions of hash proof systems (see, for example, [8, 30, 44]) imply that our construction can be based on a variety of number-theoretic assumptions, including the decisional Diffie-Hellman (DDH) assumption and its progressively weaker d-Linear variants, the quadratic residuosity assumption, and Paillier's composite residuosity assumption. Informally, a very natural approach for protecting against bounded key leakage is to add redundancy to the secret key (i.e., any public key corresponds to many secret keys), so that every (short) function of it will still keep many possibilities for the "real secret". In our generic construction we show that hash proof systems yield a convenient method for realizing this approach.

We then emphasize a specific instantiation with a simple and efficient DDH-based hash proof system. The resulting encryption scheme is resilient to any leakage of $L(1/2 - o(1))$ bits, where L is the length of the secret key. Although one can instantiate our construction with any hash proof system, we find this specific instantiation rather elegant (we refer the reader to Section 4.2).

[1] Akavia et al. refer to such attacks as *adaptive memory attacks*. They also define the notion of *non-adaptive memory attacks* which we discuss later on.

The schemes that result from our generic construction satisfy in fact a more general notion of leakage resilience: these schemes are secure even if the leakage functions chosen by the adversary are applied to the random bits used by the key generation algorithm. This clearly generalizes the framework of Akavia et al. and guarantees security even in case that intermediate values from the process of generating the secret and public keys are leaked[2]. In addition, we consider several other generalizations of the framework of Akavia et al. that are satisfied by our schemes. These include a scenario in which the adversary obtains a noisy version of all of the memory as in the attack of Halderman et al. (i.e., the leakage may be as long as the whole memory and not of bounded length), and a scenario in which partial results of the decryption process are leaked.

Improved key-leakage resilience. We propose two public-key encryption schemes that are resilient to any key leakage of $L(1 - o(1))$ bits, where L is the length of the secret key. Our proposals are based on the observation that our generic construction from hash proof systems can in fact be based on hash proof systems with a slightly weaker universality property. When viewing hash proof systems as key-encapsulation mechanisms, relaxing the universality property enables us to achieve essentially the best possible ratio between the length of the secret key and the length of the encapsulated symmetric key. This ratio translates to the relative amount of key leakage to which the encryption schemes are resilient[3].

For our first proposal we construct a new hash proof system based on the decisional Diffie-Hellman assumption (and more generally, on any of the d-Linear assumptions) that satisfies this weaker universality property. The resulting encryption scheme is then obtained by instantiating our generic construction with this hash proof system. For our second proposal, we show the recent "circular-secure" encryption scheme of Boneh et al. [6] fits into our generic approach using a different hash proof system (that satisfies the same weaker universality property).

Chosen-ciphertext security. We extend the framework of key leakage to the setting of chosen-ciphertext security. Technically, this is a very natural extension by providing the adversary with access to both a leakage oracle and a decryption oracle. On the theoretical side, we show that the Naor-Yung "double encryption" paradigm [14, 37] can be used as a general transformation from chosen-plaintext security to chosen-ciphertext security in the presence of key leakage. As an immediate corollary of our above-mentioned results, we obtain a scheme that is CCA2-secure with any leakage of $L(1 - o(1))$ bits, where L is the length of the secret key.

The schemes resulting from the Naor-Yung paradigm are rather inefficient due to the usage of generic non-interactive zero-knowledge proofs. To complement this situation, on the practical side, we prove that variants of the Cramer-Shoup cryptosystem [9] (along the lines of our generic transformation from hash proof systems) are CCA1-secure with any leakage of $L(1/4 - o(1))$ bits, and CCA2-secure with any leakage of $L(1/6 - o(1))$ bits. It is left as an open problem to construct a practical CCA-secure scheme that is resilient to any leakage of $L(1 - o(1))$ bits (where a possible approach is to examine recent refinements of the Cramer-Shoup cryptosystem [1, 30, 33]).

"Weak" key-leakage security. Akavia et al. also considered the following weaker notion of key leakage (which they refer to as "non-adaptive" leakage): a leakage function f with output length

[2] We note that it is not clear that Regev's scheme is resilient to leakage of intermediate key-related values, or at least, the proof of security of Akavia et al. does not seem to generalize to this setting. The main reason is that their proof of security involves an indistinguishability argument over the public key, and an adversary that has access to the randomness of the key generation algorithm (via leakage queries) can identify that the public key was not sampled from its specified distribution.

[3] We do not argue that such a relaxation is in fact necessary for achieving the optimal ratio.

λ is chosen by the adversary ahead of time (without any knowledge of the public key), and then the adversary is given $(pk, f(sk))$. That is, in a "weak" key-leakage attack the leakage function f is chosen independently of pk. Akavia et al. proved that Regev's encryption scheme is resilient to any weak key leakage of $L(1 - o(1))$ bits.

Although this notion of key leakage seems rather limited, it still captures many realistic attacks in which the leakage does not depend on the parameters of the encryption scheme. Specifically, this notion captures the cold boot attack of Halderman et al. [22], in which the leakage depends only on the properties of the hardware devices that are used for storing the secret key.

For weak key-leakage attacks we present a generic construction that transforms any encryption scheme to one that is resilient to any weak leakage of $L(1 - o(1))$ bits, where L is the length of the secret key. The resulting scheme is essentially as efficient as the original one, and does not rely on additional computational assumptions. Our approach crucially relies on the fact that the leakage is independent of the public key. One may interpret our construction as evidence to the deficiency of this weaker notion of key-leakage attacks.

1.2 Related Work

Extensive work has been devoted for protecting against side-channel attacks, and for exploiting side-channels to compromise the security of cryptographic schemes. It is far beyond the scope of this paper to present an exhaustive overview of this ever-growing line of work. We focus here on the results that are most relevant to our work. Already in 1985 Rivest and Shamir [41] introduced a model for leakage attacks in the context of factoring. They considered a scenario in which an adversary is interested in factoring an n-bit modulus $N = PQ$, and is allowed to ask a certain number of arbitrary "Yes/No" questions. Rivest and Shamir asked the following question: how many questions are needed in order to factor N in polynomial time? Clearly, if the adversary is allowed to ask about $n/2$ questions, then the binary representation of P can be fully revealed. Rivest and Shamir showed an attack that requires only $n/3$ questions. Specifically, in their attack the adversary requests the top $n/3$ bits of P. This was later improved by Maurer [35] who showed that ϵn questions are sufficient, for any constant $\epsilon > 0$.

Canetti et al. [7] introduced the notion of *exposure resilient* cryptographic primitives, which remain secure even if an adversary is able to learn almost all of the secret key of the primitive. Most notably, they introduced the notion of an *exposure resilient function*: a deterministic function whose output appears random even if almost all the bits of the input are known (see also the work of Dodis et al. [12] on adaptive security of such functions). Ishai et al. [26, 27] considered the more general problem of protecting privacy in circuits, where the adversary can access a bounded number of wires in the circuit. Ishai et al. proposed several techniques for dealing with this type of attacks.

Dziembowski and Pietrzak [16] and Pietrzak [39] introduced a general framework for leakage-resilient cryptography, following the assumption of Micali and Reyzin that only computation leaks information. Their main contributions are constructions of leakage-resilient stream-ciphers. Informally, their model considers cryptographic primitives that proceed in rounds, and update their internal state after each round. In each round, the adversary can obtain bounded leakage information from the portions of memory that were accessed during that round.

Dodis, Tauman Kalai, and Lovett [13] studied the security of symmetric-key encryption schemes under key leakage attacks. They considered leakage of the form $f(sk)$, where sk is the secret key and f is any exponentially-hard one-way function. On one hand they do not impose any restriction on the min-entropy of the secret key given the leakage, but on the other hand, they require that the leakage is a function that is extremely hard to invert. Dodis et al. introduced a new computational assumption that is a generalization of learning parity with noise, and constructed symmetric-key encryption schemes that are resilient to any key leakage that is exponentially hard to invert.

In a concurrent and independent work, Tauman Kalai and Vaikuntanathan [46] considered leakage of hard-to-invert functions in the setting of public-key encryption. Their main result is that the circular-secure encryption scheme of Boneh et al. [6] is resilient not only to bounded key leakage, but also to any leakage that is a sub-exponentially hard to invert (as a function of the secret key). In addition, they proved that the Naor-Yung paradigm can be used to achieve chosen-ciphertext security in the setting of key leakage, and their construction and proof of security are essentially identical to ours.

1.3 Paper Organization

The remainder of the paper is organized as follows. In Section 2 we present some basic notions, as well as several tools that are used in our constructions. In Section 3 we formally describe the framework of key-leakage attacks, and extend it to the setting of chosen-ciphertext attacks. In Section 4 we present our generic construction from hash proof systems, and provide a simple and efficient instantiation. In Section 5 we present our two proposals that are resilient to any key leakage of $L(1 - o(1))$ bits. In Section 6 we present encryption schemes that are secure against chosen-ciphertext key-leakage attacks. In Section 7 we present a generic construction for protecting against weak key-leakage attacks. In Section 8 we discuss several generalizations of the framework of key leakage that are satisfied by our schemes. In Appendix A we introduce the matrix d-Linear assumption (a generalization of the matrix DDH assumption introduced by Boneh et al. [6]) on which we base our new hash proof system in Section 5, and prove that it is implied by the d-Linear assumption.

2 Preliminaries, Assumptions, and Tools

In this section we present some basic notions, definitions, and tools that are used in our constructions. We formally state the decisional Diffie-Hellman and the d-Linear assumptions, and present the notions of an average-case strong extractor and hash proof systems.

2.1 Computational Assumptions

Let $\mathcal{G}$ be a probabilistic polynomial-time algorithm algorithm that takes as input a security parameter, and outputs a triplet $\mathbb{G} = (G, q, g)$ where G is a group of order q that is generated by $g \in G$.

The decisional Diffie-Hellman assumption. The decisional Diffie-Hellman (DDH) assumption is that the ensembles $\{(\mathbb{G}, g_1, g_2, g_1^r, g_2^r)\}_{n \in \mathbb{N}}$ and $\{(\mathbb{G}, g_1, g_2, g_1^{r_1}, g_2^{r_2})\}_{n \in \mathbb{N}}$ are computationally indistinguishable, where $\mathbb{G} \qquad \mathcal{G}(1^n)$, and the elements $g_1, g_2 \in G$ and $r, r_1, r_2 \in \mathbb{Z}_q$ are chosen independently and uniformly at random.

The *d*-Linear assumption. Boneh, Boyen, and Shacham [4] introduced the Linear assumption, intended to take the place of DDH in groups where DDH is easy (specifically, in bilinear groups). They showed that the hardness of DDH implies that hardness of Linear, but at least in generic groups (see, for example, [28, 45]), Linear remains hard even if DDH is easy. The DDH and Linear assumptions naturally generalize to the family of d-Linear assumptions [29, 44], where for every

$d \geq 1$ the d-Linear assumption is that the ensembles

$$\left\{\left(\mathbb{G}, g_1, \ldots, g_d, g_{d+1}, g_1^{r_1}, \ldots, g_d^{r_d}, g_{d+1}^{\sum_{i=1}^{d} r_i}\right)\right\}_{n\in\mathbb{N}}$$

$$\left\{\left(\mathbb{G}, g_1, \ldots, g_d, g_{d+1}, g_1^{r_1}, \ldots, g_d^{r_d}, g_{d+1}^{r_{d+1}}\right)\right\}_{n\in\mathbb{N}} \quad ,$$

are computationally indistinguishable, where $\mathbb{G} \quad \mathcal{G}(1^n)$, and the elements $g_1, \ldots, g_{d+1} \in G$ and $r_1, \ldots, r_{d+1} \in \mathbb{Z}_q$ are chosen independently and uniformly at random.

Note that DDH is the 1-Linear assumption, and that Linear is the 2-Linear assumption. These assumptions are progressively weaker: the hardness of d-Linear implies the hardness of $(d+1)$-Linear, but in generic groups $(d+1)$-Linear remains hard even if d-Linear is easy.

2.2 Randomness Extraction

The *statistical distance* between two random variables X and Y over a finite domain Ω is $\mathrm{SD}(X,Y) = \frac{1}{2}\sum_{\omega\in\Omega} |\Pr[X=\omega] - \Pr[Y=\omega]|$. We say that two variables are ϵ*-close* if their statistical distance is at most ϵ. The *min-entropy* of a random variable X is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X=x])$.

Dodis et al. [11] formalized the notion of *average min-entropy* that captures the remaining unpredictability of a random variable X conditioned on the value of a random variable Y, formally defined as follows:

$$\widetilde{\mathrm{H}}_\infty(X|Y) = -\log\left(E_{y \quad Y}\left[2^{-\mathrm{H}_\infty(X|Y=y)}\right]\right) \quad .$$

The average min-entropy corresponds exactly to the optimal probability of guessing X, given knowledge of Y. The following bound on average min-entropy was proved in [11]:

Lemma 2.1 ([11]). *If Y has 2^r possible values and Z is any random variable, then* $\widetilde{\mathrm{H}}_\infty(X|(Y,Z)) \geq \mathrm{H}_\infty(X|Z) - r$.

A main tool in our constructions in this paper is a strong randomness extractor. The following definition naturally generalizes the standard definition of a strong extractor to the setting of average min-entropy:

Definition 2.2 ([11]). *A function* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ *is an* average-case (k,ϵ)-strong extractor *if for all pairs of random variables (X, I) such that $X \in \{0,1\}^n$ and $\widetilde{\mathrm{H}}_\infty(X|I) \geq k$ it holds that*

$$\mathrm{SD}\left((\mathsf{Ext}(X,S), S, I), (U_m, S, I)\right) \leq \epsilon \quad ,$$

where S is uniform over $\{0,1\}^t$.

Dodis et al. proved that any strong extractor is in fact an average-case strong extractor, for an appropriate setting of the parameters:

Lemma 2.3 ([11]). *For any $\delta > 0$, if* Ext *is a (worst-case) $(m - \log(1/\delta), \epsilon)$-strong extractor, then* Ext *is also an average-case $(m, \epsilon + \delta)$-strong extractor.*

As a specific example, they proved the following generalized variant of the leftover hash lemma, stating that any family of pairwise independent hash functions is an average-case strong extractor:

Lemma 2.4 ([11]). *Let X, Y be random variables such that $X \in \{0,1\}^n$ and $\widetilde{\mathrm{H}}_\infty(X|Y) \geq k$. Let $\mathcal{H}$ be a family of pairwise independent hash functions from $\{0,1\}^n$ to $\{0,1\}^m$. Then for $h \quad \mathcal{H}$, it holds that*

$$\mathrm{SD}\left((Y, h, h(X)), (Y, h, U_m)\right) \leq \epsilon$$

as long as $m \leq k - 2\log(1/\epsilon)$.

2.3 Hash Proof Systems

We present the framework of hash proof systems, introduced by Cramer and Shoup [8]. For simplicity we frame the description by viewing hash proof systems as key-encapsulation mechanisms (using the notation of Kiltz et al. [30]), and refer the reader to [8] for a more complete description.

A key-encapsulation mechanism is a public-key encryption scheme that is used for encrypting random messages. Typically, these messages are used as encryption keys for a symmetric-key encryption scheme, which in turn encrypts the actual plaintext. In this setting, hash proof systems may be viewed as key-encapsulation mechanisms in which ciphertexts can be generated in two modes. Ciphertexts generated using the first mode are referred to as *valid ciphertexts*, and are indeed encapsulations of symmetric keys. That is, given a public key and a valid ciphertext, the encapsulated key is well defined, and can be decapsulated using the secret key. In addition, the generation process of a valid ciphertext also produces a "witness" to the fact that the ciphertext is indeed valid. Ciphertexts generated using the second mode are referred to as *invalid ciphertexts*, and essentially contain no information on the encapsulated key. That is, given a public key and an invalid ciphertext, the distribution of the encapsulated key (as it will be produced by the decryption process) is almost uniform. This is achieved by introducing redundancy into the secret key: each public key has many corresponding secret keys. The only computational requirement is that the two modes are computational indistinguishable: any efficient adversary that is given a public key cannot distinguish with a noticeable advantage between valid ciphertexts and invalid ciphertexts. We note that the secret and public keys are always generated using the same algorithm, and the indistinguishability requirement is only over the ciphertexts.

Smooth projective hashing. Let $\mathcal{SK}$, $\mathcal{PK}$, and $\mathcal{K}$ be sets where we view $\mathcal{SK}$ as the set of secret keys, $\mathcal{PK}$ as the set of public keys, and $\mathcal{K}$ as the set of encapsulated symmetric keys. Let $\mathcal{C}$ and $\mathcal{V} \subset \mathcal{C}$ be sets, where we view $\mathcal{C}$ as the set of all ciphertexts, $\mathcal{V}$ as the set of all valid ciphertexts (i.e., those generated appropriately with a corresponding witness). We assume that there are efficient algorithms for sampling $sk \in \mathcal{SK}$, $C \in \mathcal{V}$ together with a witness w, and $C \in \mathcal{C} \setminus \mathcal{V}$.

Let $\Lambda_{sk} : \mathcal{C} \to \mathcal{K}$ be a hash function indexed with $sk \in \mathcal{SK}$ that maps ciphertexts to symmetric keys. The hash function $\Lambda_{(\cdot)}$ is *projective* if there exists a projection $\mu : \mathcal{SK} \to \mathcal{PK}$ such that $\mu(sk) \in \mathcal{PK}$ defines the action of Λ_{sk} over the subset $\mathcal{V}$ of valid ciphertexts. That is, for every valid ciphertext $C \in \mathcal{V}$, the value $K = \Lambda_{sk}(C)$ is uniquely determined by $pk = \mu(sk)$ and C. In other words, even though there are many different secret keys sk corresponding to the same public key pk, the action of Λ_{sk} over the subset of valid ciphertexts in completely determined by the public key pk. On the other hand, the action of Λ_{sk} over the subset of invalid ciphertexts should be completely undetermined: A projective hash function is ϵ-*almost* 1-*universal* if for all $C \in \mathcal{C} \setminus \mathcal{V}$,

$$\mathrm{SD}\left((pk, \Lambda_{sk}(C)), (pk, K)\right) \leq \epsilon \tag{2.1}$$

where $sk \in \mathcal{SK}$ and $K \in \mathcal{K}$ are sampled uniformly at random, and $pk = \mu(sk)$.

Hash proof systems. A hash proof system $\mathsf{HPS} = (\mathsf{Param}, \mathsf{Pub}, \mathsf{Priv})$ consists of three algorithms that run in polynomial time. The randomized algorithm $\mathsf{Param}(1^n)$ generates parameterized instances of the form $(\mathsf{group}, \mathcal{K}, \mathcal{C}, \mathcal{V}, \mathcal{SK}, \mathcal{PK}, \Lambda_{(\cdot)}, \mu)$, where group may contain public parameters. The deterministic public evaluation algorithm Pub is used to decapsulate valid ciphertexts $C \in \mathcal{V}$ given a "witness" w of the fact that C is indeed valid (specifically, one can think of w as the random coins used to sample C from the set $\mathcal{V}$). The algorithm Pub receives as input a public key $pk = \mu(sk)$, a valid ciphertext $C \in \mathcal{V}$, and a witness w of the fact that $C \in \mathcal{V}$, and outputs the encapsulated key $K = \Lambda_{sk}(C)$. The deterministic private evaluation algorithm Priv is used to decapsulate valid ciphertexts without knowing a witness w, but by using the secret key sk. That is, the algorithm Priv receives as input a secret key $sk \in \mathcal{SK}$ and a valid ciphertext $C \in \mathcal{V}$, and outputs the encapsulated key $K = \Lambda_{sk}(C)$. We assume that μ and $\Lambda_{(\cdot)}$ are efficiently computable. We say that a hash proof system is 1-universal if for all possible outcomes of $\mathsf{Param}(1^n)$ the underlying projective hash function is $\epsilon(n)$-almost 1-universal for some negligible $\epsilon(n)$.

Subset membership problem. As a computational problem we require that the *subset membership problem* is hard in HPS, which means that for random valid ciphertext $C_0 \in \mathcal{V}$ and random invalid ciphertext $C_1 \in \mathcal{C} \setminus \mathcal{V}$, the two ciphertexts C_0 and C_1 are computationally indistinguishable. This is formally captured by defining the advantage function $\mathsf{Adv}^{\mathsf{SM}}_{\mathsf{HPS},\mathcal{A}}(n)$ of an adversary $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{SM}}_{\mathsf{HPS},\mathcal{A}}(n) = \left|\Pr_{C_0 \quad \mathcal{V}}\left[\mathcal{A}(\mathcal{C},\mathcal{V},C_0) = 1\right] - \Pr_{C_1 \quad \mathcal{C}\setminus\mathcal{V}}\left[\mathcal{A}(\mathcal{C},\mathcal{V},C_1) = 1\right]\right| \ ,$$

where $\mathcal{C}$ and $\mathcal{V}$ are generated using $\mathsf{Param}(1^n)$.

Example: A DDH-based hash proof system. Hash proof systems are known to exist based on a variety of number-theoretic assumptions: the decisional Diffie-Hellman assumption and its progressively weaker d-Linear variants, the quadratic residuosity assumption, and Paillier's composite residuosity assumption [8, 30, 44]. We conclude this section by presenting a simple example of a hash proof system (due to Cramer and Shoup [8]) that is based on the decisional Diffie-Hellman assumption. This hash proof system is used in Section 4.2 to instantiate our generic construction of a public-key encryption scheme that is resilient to key leakage.

Let $\mathbb{G}$ be a group of prime order q. We define a hash proof system $\mathsf{HPS} = (\mathsf{Param}, \mathsf{Pub}, \mathsf{Priv})$ as follows. The algorithm $\mathsf{Param}(1^n)$ generates instances $(\mathsf{group}, \mathcal{K}, \mathcal{C}, \mathcal{V}, \mathcal{SK}, \mathcal{PK}, \Lambda, \mu)$, where:

- $\mathsf{group} = (\mathbb{G}, g_1, g_2)$, where $g_1, g_2 \in \mathbb{G}$ are uniformly chosen generators.
- $\mathcal{C} = \mathbb{G}^2$, $\mathcal{V} = \{(g_1^r, g_2^r) : \ r \in \mathbb{Z}_q\}$, $\mathcal{K} = \mathbb{G}$.
- $\mathcal{SK} = \mathbb{Z}_q^2$, $\mathcal{PK} = \mathbb{G}$.
- For $sk = (x_1, x_2) \in \mathcal{SK}$ we define $\mu(sk) = g_1^{x_1} g_2^{x_2} \in \mathcal{PK}$.
- For $C = (g_1^r, g_2^r) \in \mathcal{V}$ with witness $r \in \mathbb{Z}_q$ we define $\mathsf{Pub}(pk, C, r) = pk^r$.
- For $C = (c_1, c_2) \in \mathcal{V}$ we define $\mathsf{Priv}(sk, C) = \Lambda_{sk}(C) = c_1^{x_1} c_2^{x_2}$.

This hash proof system can be easily shown to be 1-universal based on the DDH assumption (see, for example, [30] for a complete proof).

3 Defining Key-Leakage Attacks

In this section we define the notion of a key-leakage attack, as introduced as Akavia et al. [2]. In addition, we present a natural extension of this notion to the setting of chosen-ciphertext attacks, and define the notion of a weak key-leakage attack. In Section 8 we discuss several generalizations of this framework.

3.1 Chosen-Plaintext Key-Leakage Attacks

Informally, an encryption scheme is resilient to key-leakage attacks if it is semantically secure even when the adversary obtains sensitive leakage information. This is modeled by providing the adversary with access to a leakage oracle: the adversary can submit any function f and receive $f(SK)$, where SK is the secret key. The adversary can query the leakage oracle adaptively, with only one restriction: the sum of output lengths of all the leakage functions has to be bounded by a predetermined parameter λ.

More formally, for a public-key encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ we denote by $\mathcal{SK}_n$ and $\mathcal{PK}_n$ the sets of secret keys and public keys that are produced by $\mathcal{G}(1^n)$. That is, $\mathcal{G}(1^n) : \{0,1\}^* \to \mathcal{SK}_n \times \mathcal{PK}_n$ for every $n \in \mathbb{N}$. The leakage oracle, denoted $\mathsf{Leakage}(SK)$, takes as input a function $f : \mathcal{SK}_n \to \{0,1\}^*$ and outputs $f(SK)$. We say that an oracle machine $\mathcal{A}$ is a λ-key-leakage adversary if the sum of output lengths of all the functions that $\mathcal{A}$ submits to the leakage oracle is at most λ.

Definition 3.1 (key-leakage attacks)**.** *A public-key encryption scheme* $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ *is* semantically secure against $\lambda(n)$-key-leakage attacks *if for any probabilistic polynomial-time* $\lambda(n)$*-key-leakage adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *it holds that*

$$\mathsf{Adv}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(n) \stackrel{\mathsf{def}}{=} \left| \Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(0) = 1 \right] - \Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(1) = 1 \right] \right|$$

is negligible in n*, where* $\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(b)$ *is defined as follows:*

1. $(SK, PK) \quad \mathcal{G}(1^n)$.
2. $(M_0, M_1, \mathsf{state}) \quad \mathcal{A}_1^{\mathsf{Leakage}(SK)}(PK)$ *such that* $|M_0| = |M_1|$.
3. $C \quad \mathcal{E}_{pk}(M_b)$.
4. $b' \quad \mathcal{A}_2(C, \mathsf{state})$
5. *Output* b'.

Challenge-dependent key leakage. Note that the adversary is not allowed to access the leakage oracle after the challenge phase. This restriction is necessary: the adversary can clearly encode the decryption algorithm, the challenge ciphertext, and the two messages M_0 and M_1 into a function that outputs the bit b. It will be very interesting to find an appropriate definition that allows a certain form of challenge-dependent leakage.

Adaptivity. As pointed out by Akavia et al. [2], Definition 3.1 is in fact equivalent to a definition in which the adversary queries the leakage oracle only once. Informally, the adversary can encode its adaptive behavior into a single polynomial-size leakage function. It is not clear, however, that the same equivalence holds when we extend the definition to consider chosen-ciphertext attacks. Therefore, for consistency, we chose to present this adaptive definition.

3.2 Chosen-Ciphertext Key-Leakage Attacks

Extending the above definition to the setting of chosen-ciphertext security is rather natural. In this case the adversary is allowed to adaptively access a decryption oracle $\mathcal{D}(SK, \cdot)$ that receives as input a ciphertext and outputs a decryption using the secret key SK. We denote by $\mathcal{D}_{\neq C}(SK, \cdot)$ a decryption oracle that decrypts any ciphertext other than C. As in the standard definition of chosen-ciphertext attacks we distinguish between *a-priori* chosen-ciphertext attacks (CCA1) and *a-posteriori* chosen-ciphertext attacks (CCA2).

Definition 3.2 (a-priori chosen-ciphertext key-leakage attacks)**.** *A public-key encryption scheme* $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ *is* semantically secure against a-priori chosen-ciphertext $\lambda(n)$-key-leakage attacks *if for any probabilistic polynomial-time $\lambda(n)$-key-leakage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ it holds that*

$$\mathsf{Adv}_{\Pi,\mathcal{A}}^{\mathsf{LeakageCCA1}}(n) \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A}}^{\mathsf{LeakageCCA1}}(0) = 1\right] - \Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A}}^{\mathsf{LeakageCCA1}}(1) = 1\right]\right|$$

is negligible in n, where $\mathsf{Expt}_{\Pi,\mathcal{A}}^{\mathsf{LeakageCCA1}}(b)$ is defined as follows:

1. $(SK, PK) \quad \mathcal{G}(1^n)$.
2. $(M_0, M_1, \mathsf{state}) \quad \mathcal{A}_1^{\mathsf{Leakage}(SK), \mathcal{D}(SK,\cdot)}(PK)$ *such that* $|M_0| = |M_1|$.
3. $C \quad \mathcal{E}_{pk}(M_b)$.
4. $b' \quad \mathcal{A}_2(C, \mathsf{state})$
5. *Output* b'.

Definition 3.3 (a-posteriori chosen-ciphertext key-leakage attacks)**.** *A public-key encryption scheme* $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ *is* semantically secure against a-posteriori chosen-ciphertext $\lambda(n)$-key-leakage attacks *if for any probabilistic polynomial-time $\lambda(n)$-key-leakage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ it holds that*

$$\mathsf{Adv}_{\Pi,\mathcal{A}}^{\mathsf{LeakageCCA2}}(n) \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A}}^{\mathsf{LeakageCCA2}}(0) = 1\right] - \Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A}}^{\mathsf{LeakageCCA2}}(1) = 1\right]\right|$$

is negligible in n, where $\mathsf{Expt}_{\Pi,\mathcal{A}}^{\mathsf{LeakageCCA2}}(b)$ is defined as follows:

1. $(SK, PK) \quad \mathcal{G}(1^n)$.
2. $(M_0, M_1, \mathsf{state}) \quad \mathcal{A}_1^{\mathsf{Leakage}(SK), \mathcal{D}(SK,\cdot)}(PK)$ *such that* $|M_0| = |M_1|$.
3. $C \quad \mathcal{E}_{pk}(M_b)$.
4. $b' \quad \mathcal{A}_2^{\mathcal{D}_{\neq C}(SK,\cdot)}(C, \mathsf{state})$
5. *Output* b'.

3.3 Weak Key-Leakage Attacks

Akavia et al. also considered the following weaker notion of leakage (which they refer to as "non-adaptive" leakage): a leakage function f with output length λ is chosen by the adversary ahead of time (without any knowledge of the public key), and then the adversary is given $(PK, f(SK))$. That is, the leakage function f is chosen independently of the public key. Although this notion seems rather weak, it nevertheless captures realistic attacks in which the leakage depends only on hardware devices (as in [22]) and is independent of the parameters of the system, including the public key.

Definition 3.4 (weak key-leakage attacks)**.** *A public-key encryption scheme* $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ *is* semantically secure against weak $\lambda(n)$-key-leakage attacks *if for any probabilistic polynomial-time adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, and for any ensemble $\mathcal{F} = \left\{f_n : \mathcal{SK}_n \to \{0,1\}^{\lambda(n)}\right\}_{n\in\mathbb{N}}$ of efficiently computable functions, it holds that*

$$\mathsf{Adv}_{\Pi,\mathcal{A},\mathcal{F}}^{\mathsf{WeakLeakage}}(n) \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A},\mathcal{F}}^{\mathsf{WeakLeakage}}(0) = 1\right] - \Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A},\mathcal{F}}^{\mathsf{WeakLeakage}}(1) = 1\right]\right|$$

is negligible in n, where $\mathsf{Expt}_{\Pi,\mathcal{A},\mathcal{F}}^{\mathsf{WeakLeakage}}(b)$ is defined as follows:

1. $(SK, PK) \quad \mathcal{G}(1^n)$.
2. $(M_0, M_1, \mathsf{state}) \quad \mathcal{A}_1(PK, f_n(SK))$ *such that* $|M_0| = |M_1|$.
3. $C \quad \mathcal{E}_{pk}(M_b)$.
4. $b' \quad \mathcal{A}_2(C, \mathsf{state})$
5. *Output* b'.

4 A Generic Construction from Hash Proof Systems

In this section we present a generic construction of a public-key encryption scheme that is resilient to key-leakage attacks. We then present an instantiation of our generic construction with a simple and efficient hash proof system based on the DDH assumption. The resulting encryption scheme is resilient to any leakage of $L(1/2 - o(1))$ bits, where L is the length of the secret key. Although one can instantiate our generic construction with any hash proof system, we find this specific instantiation rather elegant.

4.1 The Construction

Let $\mathsf{HPS} = (\mathsf{Param}, \mathsf{Pub}, \mathsf{Priv})$ be an ϵ_1-almost 1-universal hash proof system (see Section 2.3 for an overview of hash proof systems), where $\mathsf{Param}(1^n)$ generates parameterized instances of $(\mathsf{group}, \mathcal{K}, \mathcal{C}, \mathcal{V}, \mathcal{SK}, \mathcal{PK}, \Lambda_{(\cdot)}, \mu)$ which are used as the public parameters of the encryption scheme. Let $\lambda = \lambda(n)$ be a bound on the amount of leakage, and let $\mathsf{Ext} : \mathcal{K} \times \{0,1\}^t \to \{0,1\}^m$ be an average-case $(\log|\mathcal{K}| - \lambda, \epsilon_2)$-strong extractor (see Lemma 2.3 stating that any strong extractor is in fact also an average-case strong extractor). We assume that ϵ_1 and ϵ_2 are negligible in the security parameter. The following describes the encryption scheme $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$:

- **Key generation:** Choose a random $sk \in \mathcal{SK}$ and let $pk = \mu(sk) \in \mathcal{PK}$. Output the pair (sk, pk).
- **Encryption:** On input a message $M \in \{0,1\}^m$, choose a random $C \in \mathcal{V}$ together with a corresponding witness w, and a random seed $s \in \{0,1\}^t$. Let $\Psi = \mathsf{Ext}\left(\mathsf{Pub}(pk, C, w), s\right) \oplus M$, and output the ciphertext (C, s, Ψ).
- **Decryption:** On input a ciphertext (C, s, Ψ), output the message $M = \Psi \oplus \mathsf{Ext}\left(\Lambda_{sk}(C), s\right)$.

The correctness of the scheme follows from the property that $\Lambda_{sk}(C) = \mathsf{Pub}(pk, C, w)$ for any $C \in \mathcal{V}$ with witness w. Thus, a decryption of an encrypted plaintext is always the original plaintext. The security of the scheme (i.e., its resilience to key leakage) follows from the universality of the proof system (see Equation (2.1) in Section 2.3): for all $C \in \mathcal{C} \setminus \mathcal{V}$ it holds that

$$\mathrm{SD}\left((pk, \Lambda_{sk}(C)), (pk, K)\right) \leq \epsilon_1 \ ,$$

where $sk \in \mathcal{SK}$ and $K \in \mathcal{K}$ are sampled uniformly at random, and $pk = \mu(sk)$. Therefore, even given pk and any leakage of λ bits, the distribution $\Lambda_{sk}(C)$ is ϵ_1-close to a distribution with *average min-entropy* at least $\log|\mathcal{K}| - \lambda$. The strong extractor is then applied to $\Lambda_{sk}(C)$ using a fresh seed (chosen during the challenge phase and thus independent of the leakage), and guarantees that the plaintext is properly hidden. The following theorem establishes the security of the scheme:

Theorem 4.1. *Assuming that* HPS *is a* 1*-universal hash proof system, the encryption scheme* Π *is semantically secure against* $\lambda(n)$*-key-leakage attacks for any* $\lambda(n) \leq \log|\mathcal{K}| - \omega(\log n) - m$*, where* n *is the security parameter and* m *is the length of plaintexts.*

Proof. Assume that HPS is an ϵ_1-almost 1-universal hash proof system, and recall that Ext is chosen to be an average-case $(\log|\mathcal{K}| - \lambda, \epsilon_2)$-strong extractor. We show that for any efficient λ-key-leakage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ there exists an efficient adversary $\mathcal{A}'$ for the subset membership problem of the hash proof system HPS such that

$$\mathsf{Adv}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(n) \leq 2\left(\mathsf{Adv}^{\mathsf{SM}}_{\mathsf{HPS},\mathcal{A}'}(n) + \epsilon_1(n) + \epsilon_2(n)\right) \ .$$

For $\mathcal{S} \in \{\mathcal{V}, \mathcal{C} \setminus \mathcal{V}\}$ and $b \in \{0,1\}$ consider the following experiment denoted $\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{S}, b)$:

1. Generate $(\mathcal{K}, \mathcal{C}, \mathcal{V}, \mathcal{SK}, \mathcal{PK}, \Lambda_{(\cdot)}, \mu) \quad \mathsf{Param}(1^n)$, choose a random $sk \in \mathcal{SK}$ and let $pk = \mu(sk) \in \mathcal{PK}$.
2. $(M_0, M_1, \mathsf{state}) \quad \mathcal{A}_1^{\mathsf{Leakage}(sk,pk)}(pk)$ such that $|M_0| = |M_1|$.
3. Choose a random $C \quad \mathcal{S}$, a random seed $s \in \{0,1\}^t$, and let $\Psi = \mathsf{Ext}\left(\Lambda_{sk}(C), s\right) \oplus M_b$.
4. $b' \quad \mathcal{A}_2((C, s, \Psi), \mathsf{state})$.
5. Output b'.

Using the notation of Definition 3.1 and the triangle inequality, for any adversary $\mathcal{A}$ it holds that

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(n) &= \left|\Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(1) = 1\right]\right| \\
&= \left|\Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{V}, 0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{V}, 1) = 1\right]\right| & (4.1) \\
&\leq \left|\Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{V}, 0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, 0) = 1\right]\right| & (4.2) \\
&\quad + \left|\Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, 0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, 1) = 1\right]\right| & (4.3) \\
&\quad + \left|\Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, 1) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{V}, 1) = 1\right]\right| \ , & (4.4)
\end{aligned}$$

where the equality in (4.1) follows from the property that $\Lambda_{sk}(C) = \mathsf{Pub}(pk, C, w)$ for any $C \in \mathcal{V}$ with witness w. It is straightforward that there exists an adversary $\mathcal{A}'$ for the subset membership problem of the hash proof system HPS such that the terms (4.2) and (4.4) are upper bounded by $\mathsf{Adv}^{\mathsf{SM}}_{\mathsf{HPS},\mathcal{A}'}(n)$. The following claim bounds the term (4.3) and concludes the proof of the theorem.

Claim 4.2. *For any probabilistic polynomial-time adversary* $\mathcal{A}$ *it holds that*

$$\left|\Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, 0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, 1) = 1\right]\right| \leq 2\left(\epsilon_1(n) + \epsilon_2(n)\right) \ .$$

Proof. For each $b \in \{0,1\}$ we claim that in experiment $\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, b)$ the value Ψ in the challenge ciphertext is $(\epsilon_1 + \epsilon_2)$-close to the uniform distribution over m bits from the adversary's point of view. Noting that the experiments $\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, 0)$ and $\mathsf{Expt}^{\mathsf{Leakage}}_{\Pi,\mathcal{A}}(\mathcal{C} \setminus \mathcal{V}, 1)$ differ only on Ψ, this implies that the statistical distance between the distributions of the adversary's view in these experiments is at most $2(\epsilon_1 + \epsilon_2)$.

The hash proof system guarantees that for $C \in \mathcal{C} \setminus \mathcal{V}$ the value $\Lambda_{sk}(C)$ is ϵ_1-close to uniform over the set $\mathcal{K}$ given pk and C. In our setting, however, the adversary learns additional λ bits of information using the leakage oracle. However, Lemma 2.1 states that this can reduce the average min-entropy of $\Lambda_{sk}(C)$ by at most λ. That is, $\Lambda_{sk}(C)$ is ϵ_1-close to a distribution that has average min-entropy at least $\log|\mathcal{K}| - \lambda$ given the adversary's view. The application of the strong extractor with a randomly chosen seed (that is independent of the adversary's view so far) guarantees that $\mathsf{Ext}(\Lambda_{sk}(C), s)$ is ϵ_2-close to the uniform distribution over m bits, and therefore Ψ is $(\epsilon_1 + \epsilon_2)$-close to uniform. ■

■

4.2 Example: A DDH-Based Instantiation

Let $\mathbb{G}$ be a group of prime order q, let $\lambda = \lambda(n)$ be the leakage parameter, and let $\mathsf{Ext} : \mathbb{G} \times \{0,1\}^t \to \{0,1\}^m$ be an average-case $(\log q - \lambda, \epsilon)$-strong extractor for some negligible $\epsilon = \epsilon(n)$.

- **Key generation:** Choose $x_1, x_2 \in \mathbb{Z}_q$ and $g_1, g_2 \in \mathbb{G}$ uniformly at random. Let $h = g_1^{x_1} g_2^{x_2}$, and output the keys
$$SK = (x_1, x_2)\,, \quad PK = (g_1, g_2, h) \quad .$$
- **Encryption:** On input a message M, choose $r \in \mathbb{Z}_q$ and $s \in \{0,1\}^t$ uniformly at random, and output the ciphertext
$$(g_1^r, g_2^r, s, \mathsf{Ext}(h^r, s) \oplus M) \quad .$$
- **Decryption:** On input a ciphertext (u_1, u_2, s, e), output $e \oplus \mathsf{Ext}(u_1^{x_1} u_2^{x_2}, s)$.

The hash proof system underlying the above encryption scheme is the one described in Section 2.3. This hash proof system is 1-universal based on the DDH assumption, and as an immediate consequence we obtain the following corollary of Theorem 4.1:

Corollary 4.3. *Assuming the hardness of DDH, the above encryption scheme is semantically-secure against $(L/2 - \omega(\log n) - m)$-key-leakage attacks, where n denotes the security parameter, $L = L(n)$ denotes the length of the secret key and $m = m(n)$ denotes the length of the plaintext.*

5 Improved Resilience Based on DDH and *d*-Linear

In this section we propose two encryption schemes that are resilient to any key leakage of $L(1-o(1))$ bits, where L is the length of the secret key. These proposals are based on the observation that our generic construction from hash proof systems can in fact be based on hash proof systems with a slightly weaker 1-universality property. Specifically, the 1-universality property asks that for *all* $C \in \mathcal{C} \setminus \mathcal{V}$ it holds that
$$\mathrm{SD}\left((pk, \Lambda_{sk}(C)), (pk, K)\right) \le \epsilon$$
where $sk \in \mathcal{SK}$ and $K \in \mathcal{K}$ are sampled uniformly at random, and $pk = \mu(sk)$. It is rather straightforward that our generic construction only requires this property to hold *with overwhelming probability* over the choice of $C \in \mathcal{C} \setminus \mathcal{V}$.

For our first proposal we construct a new hash proof system that is based on the d-Linear assumption (for any $d \ge 1$) and satisfies this weaker 1-universality property[4]. The hash proof system is a generalization of the hash proof system underlying the simple encryption scheme described in Section 4.2. The resulting encryption scheme is then obtained by instantiating our generic construction with this hash proof system.

[4] Recall that the DDH is the 1-Linear assumption.

Our second proposal is a recent encryption scheme of Boneh et al. [6], that is secure under key cycles (and more generally, under encryptions of linear functions of the secret keys). This is the first and only known encryption scheme with this property. We refer to this scheme as the BHHO scheme, and show that it fits into our generic approach using an appropriate hash proof system (that satisfies the same weaker universality property). As a corollary we derive that the BHHO scheme is resilient to any leakage of $L(1-o(1))$ bits[5].

The main difference between the two schemes proposed in this section is in their method of extracting randomness from the secret key. In the first proposal an invertible function is applied to the secret key (thus preserving its min-entropy), and then a strong extractor is applied to the resulting value. In the second proposal, the entropy of the secret key is extracted directly using a strong extractor that is implicit in the construction.

5.1 Proposal 1: A New Hash Proof System

We begin by presenting the encryption scheme, and then turn to describe the underlying hash proof system and its properties.

Notation. Let $\mathbb{G} = (G, q, g)$ where G a group of order q that is generated by g. For two vectors $v = (g_1, \dots, g_k) \in \mathbb{G}^k$ and $u = (u_1, \dots, u_k) \in \mathbb{Z}_q^k$ we define $v \cdot u^{\mathsf{T}} = \prod_{i=1}^{k} g_i^{u_i}$, and note the notation naturally extends to matrix-vector and matrix-matrix multiplications.

The encryption scheme. Let $k = k(n) \geq d + 1$ be any polynomial, let $\lambda = \lambda(n)$ be the leakage parameter, and let $\mathsf{Ext} : \mathbb{G}^{k-d} \times \{0,1\}^t \to \{0,1\}^m$ be an average-case $((k-d)\log q - \lambda, \epsilon)$-strong extractor for some negligible $\epsilon = \epsilon(n)$.

The following encryption scheme has a secret key of size essentially $k \log q$ bits (k group elements), and is resilient to any leakage of $\lambda \leq (k-d)\log q - \omega(\log n) - m$ bits, where m is the length of plaintexts. That is, the scheme is resilient to any leakage of essentially a $(1 - d/k)$-fraction of the length of the secret key.

- **Key generation:** Choose $x \in \mathbb{Z}_q^k$ and $\Phi \in \mathbb{G}^{d \times k}$ uniformly at random. Let $y = \Phi x \in \mathbb{G}^d$, and output the keys
$$SK = x, \quad PK = (\Phi, y) \quad .$$
- **Encryption:** On input a message M, choose $R \in \mathbb{Z}_q^{(k-d)\times d}$ and $s \in \{0,1\}^t$ uniformly at random, and output the ciphertext
$$(R\Phi, s, \mathsf{Ext}\left(Ry, s\right) \oplus M) \quad .$$
- **Decryption:** On input a ciphertext (Ψ, s, e) output $e \oplus \mathsf{Ext}\left(\Psi x, s\right)$.

The following theorem establishes the security of the scheme:

Theorem 5.1. *Assuming the hardness of d-Linear, for any polynomial $k = k(n) \geq d + 1$ the above encryption scheme is semantically-secure against a $((1-d/k)L - \omega(\log n) - m)$-key-leakage attack, where n denotes the security parameter, $L = L(n)$ denotes the length of the secret key and $m = m(n)$ denotes the length of the plaintext.*

[5] We note that not every circular-secure scheme is also resilient to key leakage.

The hash proof system. Let $k = k(n) \geq d+1$ be any polynomial, and let $\mathsf{Ext} : \mathbb{G}^{k-d} \times \{0,1\}^t \to \{0,1\}^m$ be an average-case $((k-d)\log q, \epsilon)$-strong extractor for some negligible $\epsilon = \epsilon(n)$.

We define a hash proof system $\mathsf{HPS} = (\mathsf{Param}, \mathsf{Pub}, \mathsf{Priv})$ as follows. The algorithm $\mathsf{Param}(1^n)$ generates instances $(\mathsf{group}, \mathcal{K}, \mathcal{C}, \mathcal{V}, \mathcal{SK}, \mathcal{PK}, \Lambda, \mu)$, where:

- $\mathsf{group} = (\mathbb{G}, \Phi, s)$, where $\Phi \in \mathbb{G}^{d \times k}$ and $s \in \{0,1\}^t$ are chosen uniformly at random.
- $\mathcal{C} = \mathbb{G}^{(k-d) \times k}$, $\mathcal{V} = \left\{ R\Phi \ : \ R \in \mathbb{Z}_q^{(k-d)\times d} \right\}$, $\mathcal{K} = \{0,1\}^m$.
- $\mathcal{SK} = \mathbb{Z}_q^k$, $\mathcal{PK} = \mathbb{G}^d$.
- For $sk = x \in \mathcal{SK}$ we define $\mu(sk) = \Phi x \in \mathcal{PK}$.
- For $C \in \mathcal{V}$ with witness $R \in \mathbb{Z}_q^{(k-d)\times d}$ we define $\mathsf{Pub}(pk, C, R) = \mathsf{Ext}(Ry, s)$.
- For $C \in \mathcal{V}$ we define $\mathsf{Priv}(sk, C) = \Lambda_{sk}(C) = \mathsf{Ext}(Cx, s)$.

Before analyzing the hash proof system, we note that in the above encryption scheme we did not include the strong extractor that is a part of the hash proof system. The reason is that the generic transformation in Section 4 utilizes a strong extractor (using a fresh seed for every ciphertext), and in this specific case there is no need to apply two extractors.

We now analyze the hash proof system. It is straightforward that for every $C \in \mathcal{V}$ with witness $R \in \mathbb{Z}_q^{(k-d)\times d}$ it holds that

$$\Lambda_{sk}(C) = \mathsf{Ext}(Cx, s) = \mathsf{Ext}(R\Phi x, s) = \mathsf{Ext}(Ry, s) = \mathsf{Pub}(pk, C, R) \ .$$

For the remainder of the analysis we introduce for following additional notation. We denote by $\mathrm{Rk}_i(\mathbb{Z}_q^{a\times b})$ the set of all matrices in $\mathbb{Z}_q^{a\times b}$ with rank i. For a matrix $R = \{r_{i,j}\}_{i\in[a],j\in[b]} \in \mathbb{Z}_q^{a\times b}$ we denote by g^R the matrix $\{g_{i,j}\}_{i\in[a],j\in[b]} = \{g^{r_{i,j}}\}_{i\in[a],j\in[b]} \in G^{a\times b}$. We denote by $\mathrm{Rk}_i(\mathbb{G}^{a\times b})$ the set of all matrices $g^R \in \mathbb{G}^{a\times b}$ for $R \in \mathrm{Rk}_i(\mathbb{Z}_q^{a\times b})$.

In Appendix A we show that the d-Linear assumption implies the hardness of the subset membership problem in HPS. Specifically, we introduce the *matrix d-Linear* assumption stating that in $\mathbb{G}^{k\times k}$ a random matrix of rank d is computationally indistinguishable from a random matrix of rank at least $d+1$. We prove that the d-Linear assumption implies the matrix d-Linear assumption[6]. Note that in the hash proof system, the matrix Φ has rank d with overwhelming probability, and in this case the matrix

$$\Psi = \left(\frac{\Phi}{C} \right) \in G^{k\times k}$$

has rank d for every $C \in \mathcal{V}$, and has rank at least $d+1$ for every $C \in \mathcal{C} \setminus \mathcal{V}$.

In the following claim we prove the universality of the proof system. As noted above, it suffices in our setting to argue 1-universality with overwhelming probability over the choice of $C \in \mathcal{C} \setminus \mathcal{V}$.

Claim 5.2. *With overwhelming probability over the choice of $C \in \mathcal{C} \setminus \mathcal{V}$ it holds that*

$$\mathrm{SD}\left((pk, \Lambda_{sk}(C)), (pk, K) \right) \leq \epsilon \ ,$$

where $sk \in \mathcal{SK}$ and $K \in \mathcal{K}$ are sampled uniformly at random, and $pk = \mu(sk)$.

[6] We note that the case $d = 1$ was proved by Boneh et al. [6].

Proof. With overwhelming probability the matrix $\Phi \in \mathbb{G}^{d \times k}$ has rank d (alternatively, it is possible to just sample Φ from the set of rank d matrices to begin with). In this case, with overwhelming probability over the choice over of $C \in \mathcal{C} \setminus \mathcal{V}$ the matrix

$$\Psi = \left(\frac{\Phi}{C} \right) \in G^{k \times k}$$

is invertible. Therefore, the mapping $x \to Cx$ is injective given Φ and Φx, and this implies that

$$\mathrm{H}_\infty \left(Cx \mid \Phi, \Phi x \right) = \mathrm{H}_\infty \left(x \mid \Phi, \Phi x \right) \geq (k - d) \log q \ \ .$$

The strong extractor then guarantees that $\Lambda_{sk}(C) = \mathsf{Ext}(Cx, s)$ is ϵ-close to the uniform distribution over $\mathcal{K}$ given pk. ■

5.2 Proposal 2: The BHHO Scheme

We show that a simple setting of the parameters in the BHHO encryption scheme [6] results in an encryption scheme that is resilient any key leakage of $L(1 - o(1))$ bits, where L is the length of the secret key. Let $\mathbb{G} = (G, q, g)$ where G a group of order q that is generated by g, and set $\ell = \lambda + 2 \log q + 2 \log(1/\epsilon)$ for some negligible $\epsilon = \epsilon(n)$.

- **Key generation:** Choose $s_1, \ldots, s_\ell \in \{0, 1\}$ and $g_1, \ldots, g_\ell \in \mathbb{G}$ uniformly at random. Let $h = \prod_{i=1}^{\ell} g_i^{s_i}$, and output the keys
$$SK = (s_1, \ldots, s_\ell), \quad PK = (g_1, \ldots, g_\ell, h) \ \ .$$
- **Encryption:** On input a message $M \in G$, choose $r \in \mathbb{Z}_q$ uniformly at random, and output the ciphertext
$$(g_1^r, \ldots, g_\ell^r, h^r \cdot M) \ \ .$$
- **Decryption:** On input a ciphertext $(u_1, \ldots, u_k, e)$ output $e \cdot \left(\prod_{i=1}^{\ell} u_i^{s_i} \right)^{-1}$.

The encryption scheme can be viewed as based on a hash proof system with the following subset membership problem (whose hardness follows from DDH):

$$\begin{aligned} \mathcal{C} &= \left\{ \left(g_1^{r_1}, \ldots, g_\ell^{r_\ell} \right) : r_1, \ldots, r_\ell \in \mathbb{Z}_q \right\} \\ \mathcal{V} &= \left\{ \left(g_1^{r}, \ldots, g_\ell^{r} \right) : r \in \mathbb{Z}_q \right\} \ \ . \end{aligned}$$

The leftover hash lemma guarantees that with overwhelming probability over the choice of $C = (u_1, \ldots, u_\ell) \in \mathcal{C} \setminus \mathcal{V}$ it holds that $\Lambda_{sk}(C) = \prod_{i=1}^{\ell} u_i^{s_i}$ is ϵ-close to the uniform distribution over G, even given $h = \prod_{i=1}^{\ell} g_i^{s_i}$ and any leakage of length λ bits.

Improved efficiency. A rather straightforward improvement to the efficiency of the above scheme is choosing the values $s_1, \ldots, s_\ell$ as elements of $\mathbb{Z}_q$ instead of binary values as in the original scheme. In turn, this allows to decrease the value of ℓ: all we need is that even given the public key and any λ bits of leakage, the remaining average-min entropy in the secret key is at least $\log q + \omega(\log n)$ (note that this suffices for applying the leftover hash lemma). That is, we need to set ℓ such that $\mathrm{H}_\infty \left(sk \right) - \log q - \lambda \geq \log q + \omega(\log n)$, which implies that $\ell = 2 + \frac{\lambda + \omega(\log n)}{\log q}$ suffices.

6 Protecting Against Chosen-Ciphertext Key-Leakage Attacks

In this section we present public-key encryption schemes that are resilient to chosen-ciphertext key-leakage attacks (see Definitions 3.2 and 3.3).

6.1 A Generic Construction

We prove that the Naor-Yung "double encryption" paradigm [14, 37] is applicable also in the setting of key leakage attacks. That is, we show that any public-key encryption scheme that is semantically secure against *chosen-plaintext* key-leakage attacks can be transformed into a scheme that is semantically secure against *a-posteriori chosen-ciphertext* key-leakage attacks using non-interactive zero-knowledge proofs.

The key property of the transformation is that the size of the secret key in the resulting CCA-secure scheme is exactly the same as in the underlying CPA-secure scheme. This enables us to prove that the resulting CCA-secure scheme is resilient to exactly the same amount of leakage as the underlying CPA-secure scheme.

Our construction in this section are based on the refinements of Sahai [43] and Lindell [34] to the encryption schemes of Naor and Yung [37] and Dolev, Dwork and Naor [14]. These refinements enable us to achieve CCA2 security without increasing the size of the secret key. We refer the reader to [34, 43] for comprehensive overviews of the construction.

The construction. Let $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a public-key encryption scheme that is semantically secure against chosen-plaintext λ-key-leakage attacks, and let $(\mathcal{P}, \mathcal{V})$ be a one-time simulation-sound adaptive NIZK proof system[7] for the following NP-language:

$$L = \{(c_0, c_1, pk_0, pk_1) \mid \exists m, r_0, r_1 \text{ s.t. } c_0 = \mathcal{E}_{pk_0}(m; r_0) \text{ and } c_1 = \mathcal{E}_{pk_1}(m; r_1)\} \ .$$

The encryption scheme $\Pi' = (\mathcal{G}', \mathcal{E}', \mathcal{D}')$ is defined as follows:

- **Key generation:** Sample $(sk_0, pk_0), (sk_1, pk_1) \quad \mathcal{G}(1^n)$ independently, and a reference string σ for the NIZK proof system. Output $SK = sk_0$ and $PK = (pk_0, pk_1, \sigma)$.
- **Encryption:** On input a message M choose $r_0, r_1 \in \{0,1\}^*$, and compute $c_0 = \mathcal{E}_{pk_0}(M; r_0)$ and $c_1 = \mathcal{E}_{pk_1}(M; r_1)$. Then, invoke the NIZK prover $\mathcal{P}$ to obtain a proof π for the statement $(c_0, c_1, pk_0, pk_1) \in L$ with respect to the reference string σ. Output the ciphertext (c_0, c_1, π).
- **Decryption:** On input a ciphertext (c_0, c_1, π), invoke the NIZK verifier $\mathcal{V}$ to verify that π is an accepting proof with respect to the reference string σ. If $\mathcal{V}$ accepts then output $\mathcal{D}_{sk_0}(c_0)$, and otherwise output $\perp$.

Proof of security. In the remainder of the section we prove the security of the scheme Π' by adapting the ideas of Lindell [34] and Sahai [43] to key leakage attacks. Specifically, we show that any adversary that breaks the security of the scheme can be used to either break the simulation soundness of the proof system $(\mathcal{P}, \mathcal{V})$ or to break the encryption scheme Π. The main point in our setting, is that we can always simulate both the leakage oracle and the decryption oracle to such an adversary:

- When attacking the proof system $(\mathcal{P}, \mathcal{V})$ we know the key sk_0. We use this key to simulate the leakage oracle, and use this key together with the verifier of the proof system to simulate the decryption oracle.

[7] We refer the reader to [34, 43] for the definitions of a one-time simulation-sound adaptive NIZK proof system.

- When attacking the public key pk_1 we again know the key sk_0 as in the previous case. In fact, pk_1 can even be sampled from a CPA-secure encryption scheme that is not resilient to key leakage (both in the actual scheme and in the proof of security the adversary never obtains leakage information from sk_1).
- When attacking the public key pk_0 we know the key sk_1. We simulate the leakage oracle by forwarding all queries to the leakage oracle of sk_0, and we simulate the decryption oracle using sk_1 and the verifier of the proof system.

We prove the following theorem:

Theorem 6.1. *Assume that Π is semantically secure against chosen-plaintext λ-key-leakage attacks, and that $(\mathcal{P}, \mathcal{V})$ is a one-time simulation-sound NIZK proof system. Then, Π' is semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attacks.*

Proof. Given a probabilistic polynomial-time λ-key-leakage adversary $\mathcal{A}$ we begin by describing a mental experiment in which $\mathcal{A}$ runs its attack. The difference between the mental experiment and a real attack is that the challenge ciphertext received by $\mathcal{A}$ is not properly generated. Rather, the NIZK proof is generated by the NIZK simulator $S = (S_1, S_2)$ and the two encryptions may not be to the same message.

Experiment $\mathsf{Expt}^{S}_{\mathcal{A}}(b_0, b_1)$:

1. Key generation:
   - $(\sigma, s) \quad S_1(1^n)$ (a simulated reference string σ is used for the public key).
   - $(sk_0, pk_0), (sk_1, pk_1) \quad \mathcal{G}(1^n)$.
   - Set $SK = sk_0$ and $PK = (pk_0, pk_1, r)$.
2. $(M_0, M_1, \mathsf{state}) \quad \mathcal{A}_1^{\mathsf{Leakage}(SK), \mathcal{D}'(SK, \cdot)}(PK)$.
3. Set up the challenge ciphertext $c = (c_0, c_1, \pi)$:
   - $c_0 \quad \mathcal{E}_{pk_0}(M_{b_0})$.
   - $c_1 \quad \mathcal{E}_{pk_1}(M_{b_1})$.
   - $\pi \quad S_2((c_0, c_1, pk_0, pk_1), \sigma, s)$ (a simulated proof π is used for the challenge).
4. $b' \quad \mathcal{A}_2^{\mathcal{D}'_{\neq c}(SK, \cdot)}(c, \mathsf{state})$.

Given the above experiment, the advantage of $\mathcal{A}$ in attacking the scheme Π' can be bounded as follows:

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{LeakageCCA2}}_{\Pi', \mathcal{A}}(n) &= \left|\Pr\left[\mathsf{Expt}^{\mathsf{LeakageCCA2}}_{\Pi', \mathcal{A}}(1) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{LeakageCCA2}}_{\Pi', \mathcal{A}}(0) = 1\right]\right| \\
&\leq \left|\Pr\left[\mathsf{Expt}^{\mathsf{LeakageCCA2}}_{\Pi', \mathcal{A}}(1) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(1,1) = 1\right]\right| \qquad (6.1) \\
&\quad + \left|\Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(1,1) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,1) = 1\right]\right| \qquad (6.2) \\
&\quad + \left|\Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,1) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,0) = 1\right]\right| \qquad (6.3) \\
&\quad + \left|\Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,0) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{LeakageCCA2}}_{\Pi', \mathcal{A}}(0) = 1\right]\right| . \qquad (6.4)
\end{aligned}$$

The following claim shows that the terms (6.1) and (6.4) are negligible.

Claim 6.2. *For every probabilistic polynomial-time $\mathcal{A}$ and $b \in \{0,1\}$ it holds that*

$$\left|\Pr\left[\mathsf{Expt}^{\mathsf{LeakageCCA2}}_{\Pi',\mathcal{A}}(b) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(b,b) = 1\right]\right| \leq \epsilon(n)$$

for some negligible function $\epsilon(n)$.

Proof. The correctness of the claim follows from the zero-knowledge property of the NIZK proof system $(\mathcal{P},\mathcal{V})$. Given a probabilistic polynomial-time λ-key-leakage adversary $\mathcal{A}$ and a bit $b \in \{0,1\}$ for which $\left|\Pr\left[\mathsf{Expt}^{\mathsf{LeakageCCA2}}_{\Pi',\mathcal{A}}(b) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(b,b) = 1\right]\right| \geq \epsilon(n)$, we construct a probabilistic polynomial-time distinguisher $\mathcal{A}'$ that distinguishes a simulated proof from a real proof with advantage $\epsilon(n)$. Thus, the adaptive zero-knowledge property of $(\mathcal{P},\mathcal{V})$ implies that $\epsilon(n)$ is negligible.

The distinguisher $\mathcal{A}'$ works as follows: it receives a reference string σ, samples two pairs of keys $(sk_0, pk_0), (sk_1, pk_1) \quad \mathcal{G}(1^n)$ and invokes $\mathcal{A}$ on the public-key $PK = (pk_0, pk_1, \sigma)$ while emulating the decryption oracle and the leakage oracle for $\mathcal{A}$ using $SK = sk_0$ (note that anyone can verify the NIZK proofs for the decryption process). When $\mathcal{A}$ outputs two messages (M_0, M_1) for the challenge, $\mathcal{A}'$ computes $c_0 = \mathcal{E}_{pk_0}(M_b)$ and $c_1 = \mathcal{E}_{pk_1}(M_b)$, and asks for a proof π of the (correct) statement $(c_0, c_1, pk_0, pk_1) \in L$. The distinguisher $\mathcal{A}'$ then gives $\mathcal{A}$ the challenge ciphertext (c_0, c_1, π) and outputs whatever $\mathcal{A}$ does. Notice that if $\mathcal{A}'$ receives a real proof then it perfectly simulates $\mathsf{Expt}^{\mathsf{LeakageCCA2}}_{\Pi',\mathcal{A}}(b)$, and if $\mathcal{A}'$ receives a simulated proof then it perfectly simulates $\mathsf{Expt}^{S}_{\mathcal{A}}(b,b)$. Thus, the advantage of $\mathcal{A}'$ is exactly $\epsilon(n)$. ■

Before proceeding, we prove a claim stating that in the experiment $\mathsf{Expt}^{S}_{\mathcal{A}}(b_0, b_1)$, with overwhelming probability all ciphertext queries to the decryption oracle that contain accepting NIZK proofs, are of valid ciphertexts. Formally, we say that a ciphertext $c = (c_0, c_1, \pi)$ is *invalid with an accepting NIZK proof* if $D_{sk_0}(c_0) \neq D_{sk_1}(c_1)$ and $V((c_0, c_1, pk_0, pk_1), \sigma, \pi) = 1$. The following claim shows that if any of $\mathcal{A}$'s ciphertext queries are of the above form, then $\mathcal{A}$ can be used to contradict the one-time simulation-soundness of the NIZK proof system.

Claim 6.3. *For every probabilistic polynomial-time adversary $\mathcal{A}$ and $b_0, b_1 \in \{0,1\}$, the probability that in the experiment $\mathsf{Expt}^{S}_{\mathcal{A}}(b_0, b_1)$ the adversary $\mathcal{A}$ queries the decryption oracle with an invalid ciphertext that has an accepting NIZK proof is negligible.*

Proof. The correctness of the claim follows from the one-time simulation-soundness of the NIZK proof system $(\mathcal{P},\mathcal{V})$. Given a probabilistic polynomial-time adversary $\mathcal{A}$ and $b_0, b_1 \in \{0,1\}$ for which with a non-negligible probability $\mathcal{A}$ queries the decryption oracle with an invalid ciphertext that has an accepting NIZK proof, we construct a probabilistic polynomial-time $\mathcal{A}'$ that contradicts the one-time simulation-soundness of $(\mathcal{P},\mathcal{V})$.

$\mathcal{A}'$ receives a simulator-generated reference string σ and runs the rest of experiment $\mathsf{Expt}^{S}_{\mathcal{A}}(b_0, b_1)$ as in the proof of Claim 6.2 (the only difference is that $\mathcal{A}'$ computes $c_0 = \mathcal{E}_{pk_0}(M_{b_0})$ and $c_1 = \mathcal{E}_{pk_1}(M_{b_1})$). If during the simulation $\mathcal{A}$ queries the decryption oracle with an invalid ciphertext that has an accepting NIZK proof, then $\mathcal{A}'$ outputs this ciphertext and halt ($\mathcal{A}'$ can check this because it knows both decryption keys and because it can verify the validity of the NIZK proofs). The simulation by $\mathcal{A}'$ is perfect, and therefore if $\mathcal{A}$ outputs an invalid ciphertext with an accepting proof in $\mathsf{Expt}^{S}_{\mathcal{A}}(b_0, b_1)$, then $\mathcal{A}'$ outputs a false statement with an accepting proof. ■

We are now ready to prove that the terms (6.2) and (6.3) are negligible.

Claim 6.4. *For every probabilistic polynomial-time $\mathcal{A}$ it holds that*

$$\left|\Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(1,1) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,1) = 1\right]\right| \leq \epsilon(n)$$

for some negligible function $\epsilon(n)$.

Proof. The correctness of the claim follows from the security of the scheme $\Pi = (G, E, D)$. Given a probabilistic polynomial-time adversary $\mathcal{A}$ for which $\left|\Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(1,1) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,1) = 1\right]\right| \geq \epsilon(n)$, we construct a probabilistic polynomial-time adversary $\mathcal{A}'$ that breaks the security of Π with essentially the same advantage.

The adversary $\mathcal{A}'$ receives as input a public-key pk generated by $\mathcal{G}(1^n)$, and simulates the view of $\mathcal{A}$ as follows:

1. Key generation:
   - $\mathcal{A}'$ sets $pk_0 = pk$, and samples $(sk_1, pk_1) \quad \mathcal{G}(1^n)$ and $(\sigma, s) \quad S_1(1^n)$.
   - $\mathcal{A}'$ sets $PK = (pk_0, pk_1, \sigma)$.
2. Leakage queries: Any leakage query f is forwarded to the leakage oracle corresponding to pk.
3. Decryption queries: Given a ciphertext $c = (c_0, c_1, \pi)$ the adversary $\mathcal{A}'$ invokes the NIZK verifier $\mathcal{V}$ to verify that π is an accepting proof with respect to the reference string σ. If $\mathcal{V}$ accepts then $\mathcal{A}'$ outputs $\mathcal{D}_{sk_1}(c_1)$, and otherwise $\mathcal{A}'$ outputs $\perp$.
4. When $\mathcal{A}$ outputs two messages M_0 and M_1, the adversary $\mathcal{A}'$ obtains $c_0 = \mathcal{E}_{pk_0}(M_b)$ for a random $b \in \{0,1\}$, and outputs the challenge ciphertext $c = (c_0, c_1, \pi)$, where:
   - $c_1 \quad \mathcal{E}_{pk_1}(M_1)$.
   - $\pi \quad S_2((c_0, c_1, pk_0, pk_1), \sigma, s)$ (a simulated NIZK proof).
5. $\mathcal{A}'$ outputs the output of $\mathcal{A}$.

From the point of view of $\mathcal{A}$, the only difference between the view simulated by $\mathcal{A}'$ and the experiment $\mathsf{Expt}^{S}_{\mathcal{A}}(b,1)$ is that $\mathcal{A}'$ performs the decryption using sk_1 and not sk_0. However, these views are identical as long as $\mathcal{A}$ does not submit an invalid ciphertext that has an accepting NIZK proof. Claim 6.3 guarantees that the latter event has only a negligible probability. ■

Claim 6.5. *For every probabilistic polynomial-time $\mathcal{A}$ it holds that*

$$\left|\Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,1) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,0) = 1\right]\right| \leq \epsilon(n)$$

for some negligible function $\epsilon(n)$.

Proof. The correctness of the claim follows from the security of the scheme $\Pi = (G, E, D)$. Given a probabilistic polynomial-time adversary $\mathcal{A}$ for which $\left|\Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,1) = 1\right] - \Pr\left[\mathsf{Expt}^{S}_{\mathcal{A}}(0,0) = 1\right]\right| \geq \epsilon(n)$, we construct a probabilistic polynomial-time adversary $\mathcal{A}'$ that breaks the security of Π with essentially the same advantage.

The adversary $\mathcal{A}'$ receives as input a public-key pk generated by $\mathcal{G}(1^n)$, and simulates the view of $\mathcal{A}$ as follows:

1. Key generation:
   - $\mathcal{A}'$ sets $pk_1 = pk$, and samples $(sk_0, pk_0) \quad \mathcal{G}(1^n)$ and $(\sigma, s) \quad S_1(1^n)$.
   - $\mathcal{A}'$ sets $PK = (pk_0, pk_1, \sigma)$.
2. Leakage queries: Given a function f the simulator $\mathcal{A}'$ outputs $f(sk_0)$.

3. Decryption queries: Given a ciphertext $c = (c_0, c_1, \pi)$ the simulator $\mathcal{A}'$ invokes the NIZK verifier $\mathcal{V}$ to verify that π is an accepting proof with respect to the reference string σ. If $\mathcal{V}$ accepts then $\mathcal{A}$ outputs $\mathcal{D}_{sk_0}(c_0)$, and otherwise $\mathcal{A}$ outputs $\perp$.

4. When $\mathcal{A}$ outputs two messages M_0 and M_1, the simulator $\mathcal{A}'$ obtains $c_1 = \mathcal{E}_{pk_1}(M_b)$ for a random $b \in \{0,1\}$, and outputs the challenge ciphertext $c = (c_0, c_1, \pi)$, where:
   - $c_0 \quad \mathcal{E}_{pk_0}(M_0)$.
   - $\pi \quad S_2((c_0, c_1, pk_0, pk_1), \sigma, s)$ (a simulated NIZK proof).

5. $\mathcal{A}'$ outputs the output of $\mathcal{A}$.

The claim now follows by noting that the simulated view of $\mathcal{A}$ is in fact identical to the experiment $\mathsf{Expt}^S_{\mathcal{A}}(0, b)$. ■

This concludes the proof of the theorem. ■

6.2 An Efficient CCA1-Secure Scheme

In this section we prove that a variant of the Cramer-Shoup "Lite" cryptosystem [9] (along the lines of our generic construction in Section 4) is secure against a-priori chosen-ciphertext key-leakage attacks.

Let $\mathbb{G}$ be a group of prime order q, let $\lambda = \lambda(n)$ be the leakage parameter, and let $\mathsf{Ext} : \mathbb{G} \times \{0,1\}^t \to \{0,1\}^m$ be an average-case $(\log q - \lambda, \epsilon)$-strong extractor for some negligible $\epsilon = \epsilon(n)$.

We present an encryption scheme with a secret key of size essentially $4 \log q$ bits (four group elements), and show that the scheme is secure for any leakage of length $\lambda \leq \log q - \omega(\log n) - m$, where m is the length of plaintexts. The following describes the encryption scheme:

- **Key generation:** Choose $x_1, x_2, z_1, z_2 \in \mathbb{Z}_q$ and $g_1, g_2 \in \mathbb{G}$ uniformly at random. Let $c = g_1^{x_1} g_2^{x_2}$, $h = g_1^{z_1} g_2^{z_2}$, and output the keys

$$SK = (x_1, x_2, z_1, z_2), \quad PK = (g_1, g_2, c, h) \ .$$

- **Encryption:** On input a message $M \in \{0,1\}^m$, choose $r \in \mathbb{Z}_q$ and $s \in \{0,1\}^t$ uniformly at random, and output the ciphertext

$$(g_1^r, g_2^r, c^r, s, \mathsf{Ext}(h^r, s) \oplus M) \ .$$

- **Decryption:** On input a ciphertext (u_1, u_2, v, s, e), if $v \neq u_1^{x_1} u_2^{x_2}$ then output $\perp$ and otherwise output $e \oplus \mathsf{Ext}(u_1^{z_1} u_2^{z_2}, s)$.

Correctness. For any sequence of coin tosses of the key generation and encryption algorithms it holds that $u_1^{x_1} u_2^{x_2} = (g_1^{x_1} g_2^{x_2})^r = c^r = v$ and that $u_1^{z_1} u_2^{z_2} = (g_1^{z_1} g_2^{z_2})^r = h^r$, and therefore the decryption algorithm is always correct.

Proof of security. The proof of security is based on augmenting the proof of Cramer and Shoup with the ideas presented in Section 4. We prove the following theorem:

Theorem 6.6. *Assuming the hardness of DDH, the above encryption scheme is semantically-secure against a-priori chosen-ciphertext $(L/4 - \omega(\log n) - m)$-key-leakage attacks, where n denotes the security parameter, $L = L(n)$ denotes the length of the secret key and $m = m(n)$ denotes the length of the plaintext.*

Proof. We show that any efficient adversary $\mathcal{A}$ that breaks the security of the scheme (recall Definition 3.2) can be used to construct an efficient algorithm $\mathcal{A}'$ that distinguishes between a DH instance and a non-DH instance with a non-negligible advantage. On input $(g_1, g_2, u_1, u_2) \in \mathbb{G}^4$ the algorithm $\mathcal{A}'$ acts as follows:

1. $\mathcal{A}'$ chooses $x_1, x_2, z_1, z_2 \in \mathbb{Z}_q$ uniformly at random, and sets $c = g_1^{x_1} g_2^{x_2}$, $h = g_1^{z_1} g_2^{z_2}$, $SK = (x_1, x_2, z_1, z_2)$, and $PK = (g_1, g_2, c, h)$. Then $\mathcal{A}'$ invokes $\mathcal{A}$ with input PK.

2. $\mathcal{A}'$ simulates the leakage oracle and the decryption oracle to $\mathcal{A}$ using SK.

3. When $\mathcal{A}$ outputs two messages M_0 and M_1, the simulator $\mathcal{A}'$ chooses $b \in \{0,1\}$ and $s \in \{0,1\}^t$ uniformly at random, and sends $\mathcal{A}$ the challenge ciphertext $(u_1, u_2, u_1^{x_1} u_2^{x_2}, s, \mathsf{Ext}(u_1^{z_1} u_2^{z_2}, s) \oplus M_b)$.

4. If $\mathcal{A}$ outputs b then $\mathcal{A}'$ outputs 1, and otherwise $\mathcal{A}'$ outputs 0.

In the following we say that a ciphertext (u_1, u_2, v, s, e) is *invalid* if $\log_{g_1}(u_1) \neq \log_{g_2}(u_2)$. The remainder of the proof proceeds as follows. First, we prove that if (g_1, g_1, u_1, u_2) is a DH instance then $\mathcal{A}$'s view is identical to the actual attack. Then, we prove that in both the actual attack and in the simulated attack the decryption oracle rejects all invalid ciphertexts except with a negligible probability. Finally, we prove that if (g_1, g_1, u_1, u_2) is a non-DH instance and the decryption oracle rejects all invalid ciphertexts then $\mathcal{A}$ has only a negligible advantage in outputting the bit b. Thus, if $\mathcal{A}$ has a non-negligible advantage in the actual attack, then $\mathcal{A}'$ has a non-negligible advantage in distinguishing between DH instances and non-DH instances.

Claim 6.7. *If (g_1, g_2, u_1, u_2) is a DH instance then $\mathcal{A}$'s view is identical to the actual attack.*

Proof. The actual attack and the simulated attack are identical up to the challenge phase. It remains to prove that the challenge ciphertext has the correct distribution when (g_1, g_2, u_1, u_2) is a DH instance. Indeed, in this case $u_1 = g_1^r$ and $u_2 = g_2^r$ for some randomly chosen $r \in \mathbb{Z}_q$, and therefore $u_1^{x_1} u_2^{x_2} = c^r$ and $u_1^{z_1} u_2^{z_2} = h^r$ as it should be. ■

Claim 6.8. *In both the actual attack and the simulated attack, the decryption algorithm rejects all invalid ciphertexts except with a negligible probability.*

Proof. The actual attack and the simulated attack are identical up to the challenge phase. Therefore, the probability that the decryption algorithm rejects all invalid ciphertexts is the same in both attacks.

We prove the claim by considering the distribution of the point $(x_1, x_2) \in \mathbb{Z}_q^2$ from the adversary's point of view. The adversary is given the public key (g_1, g_2, c, h) and therefore from the adversary's point of view the point (x_1, x_2) is uniformly random subject to $\log_{g_1}(c) = x_1 + \gamma x_2$, where $\gamma = \log_{g_1}(g_2)$. Moreover, by submitting valid ciphertexts to the decryption oracle the adversary does not learn any more information on (x_1, x_2) (in fact, by submitting a valid ciphertext the adversary only learns a linear combination of the constraint $\log_{g_1}(h) = z_1 + \gamma z_2$ that is already known from the public key).

Denote by (u'_1, u'_2, v', s', e') the first invalid ciphertext submitted by the adversary, where $u'_1 = g_1^{r'_1}$, $u'_2 = g_2^{r'_2}$, and $r'_1 \neq r'_2$. Denote by aux the output of all leakage functions that were submitted by the adversary prior to submitting the invalid ciphertext. The value aux obtains at most 2^λ values, and therefore Lemma 2.1 implies that from the adversary's point of view prior to submitting the invalid ciphertext it holds that

$$\widetilde{\mathrm{H}}_\infty\left((x_1, x_2)|PK, \mathsf{aux}\right) \geq \mathrm{H}_\infty\left((x_1, x_2)|PK\right) - \lambda \geq \log q - \lambda \ .$$

In particular, the definition of average min-entropy implies that prior to submitting the invalid ciphertext the probability of $\mathcal{A}$ in guessing (x_1, x_2) is at most $2^{-\widetilde{\mathrm{H}}_\infty((x_1,x_2)|PK,\mathsf{aux})} \leq 2^\lambda/q$. However, note that if the decryption algorithm accepts the invalid ciphertext then we obtain the following linear equations:

$$\begin{cases} \log_{g_1}(v') &= r'_1 x_1 + \gamma r'_2 x_2 \\ \log_{g_1}(c) &= x_1 + \gamma x_2 \end{cases}$$

As long as $\gamma(r'_1 - r'_2) \neq 0$ these equations are linearly independent, and therefore the adversary can be used to guess (x_1, x_2). Thus, the probability that the decryption algorithm accepts the first invalid ciphertext is at most $2^\lambda/q$.

An almost identical argument holds for all the subsequent invalid decryption queries. The only difference is that each time the decryption oracle rejects an invalid ciphertext the adversary can rule out one more value of (x_1, x_2) from the set $\{(x_1, x_2) \in \mathbb{Z}_q^2 : \log_{g_1}(c) = x_1 + \gamma x_2\}$. This shows that the decryption algorithm accepts the i-th invalid ciphertext with probability at most $2^\lambda/(q - i + 1)$. The claim now follows from the fact that the number of decryption queries is polynomial, and from the restriction $\lambda \leq \log q - \omega(\log n)$. ■

Claim 6.9. *If (g_1, g_2, u_1, u_2) is a non-DH instance and the decryption algorithm rejects all invalid ciphertexts, then $\mathcal{A}$ has only a negligible advantage in outputting the bit b.*

Proof. We show that if (g_1, g_2, u_1, u_2) is a non-DH instance and the decryption algorithm rejects all invalid ciphertexts then with overwhelming probability the value $u_1^{z_1} u_2^{z_2}$ has average min-entropy at least $\log q - \lambda \geq m + \omega(\log n)$ given all the other values in the adversary's view. The strong extractor then guarantees that the part of the challenge ciphertext that depends on the bit b is ϵ-close to uniform given the adversary's view, for a negligible $\epsilon = \epsilon(n)$.

We prove the claim by considering the distribution of the point $(z_1, z_2) \in \mathbb{Z}_q$ from the adversary's point of view. The adversary is given the public key (g_1, g_2, c, h) and therefore from the adversary's point of view the point (z_1, z_2) is uniformly random subject to $\log_{g_1}(h) = z_1 + \gamma z_2$, where $\gamma = \log_{g_1}(g_2)$. We assume that the decryption algorithm rejects all invalid ciphertext, and note that by submitting valid ciphertexts to the decryption oracle the adversary does not learn any more information (the adversary only learns a linear combination of the constraint $\log_{g_1}(h) = z_1 + \gamma z_2$). Thus, the adversary does not learn any information on (z_1, z_2) via decryption queries.

Let $u_1 = g_1^{r_1}$, $u_2 = g_2^{r_2}$, and denote by aux the output of all leakage functions chosen by the adversary. Then from the adversary's point of view in the challenge phase it holds that

$$\widetilde{\mathrm{H}}_\infty\left(u_1^{z_1} u_2^{z_2} \mid g_1, g_2, c, h, \mathsf{aux}, u_1, u_2\right) = \widetilde{\mathrm{H}}_\infty\left(r_1 z_1 + \gamma r_2 z_2 \mid \gamma, c, h, \mathsf{aux}, r_1, r_2\right) \ .$$

Note that as long as $\gamma(r_1 - r_2) \neq 0$, then for any $w = u_1^{z_1} u_2^{z_2}$ the following two equations (in z_1 and z_2) are linearly independent:

$$\begin{cases} \log_{g_1}(h) &= z_1 + \gamma z_2 \\ \log_{g_1}(w) &= r_1 z_1 + \gamma r_2 z_2 \end{cases}$$

which implies that given (γ, h, r_1, r_2) the function $(z_1, z_2) \to r_1 z_1 + \gamma r_2 z_2$ is injective. Applying an injective function to a distribution preserves its min-entropy, and therefore

$$\begin{aligned} \widetilde{\mathrm{H}}_\infty\left(r_1 z_1 + \gamma r_2 z_2 \mid \gamma, c, h, \mathsf{aux}, r_1, r_2\right) &= \widetilde{\mathrm{H}}_\infty\left((z_1, z_2) \mid \gamma, c, h, \mathsf{aux}, r_1, r_2\right) \\ &= \widetilde{\mathrm{H}}_\infty\left((z_1, z_2) \mid \gamma, c, h, \mathsf{aux}\right) \ , \end{aligned}$$

where the second equality follows from the fact that $(z_1, z_2, \gamma, c, h, \mathsf{aux})$ are all chosen independently of (r_1, r_2). Finally, the leakage aux obtains at most 2^λ possible values, and therefore (see Lemma 2.1)

$$\begin{aligned} \widetilde{\mathrm{H}}_\infty\left((z_1, z_2) \mid \gamma, c, h, \mathsf{aux}\right) &\geq \mathrm{H}_\infty\left((z_1, z_2) \mid \gamma, c, h\right) - \lambda \\ &= \mathrm{H}_\infty\left((z_1, z_2) \mid \gamma, z_1 + \gamma z_2\right) - \lambda \\ &\geq \log q - \lambda \ . \end{aligned}$$

Combining all of the above, with overwhelming probability it holds that

$$\widetilde{\mathrm{H}}_\infty\left(u_1^{z_1} u_2^{z_2} \mid g_1, g_2, c, h, \mathsf{aux}, u_1, u_2\right) \geq \log q - \lambda \ .$$

■

This conclude the proof of Theorem 6.6. ■

6.3 An Efficient CCA2-Secure Scheme

In this section we prove that a variant of the Cramer-Shoup cryptosystem [9] (along the lines of our generic construction in Section 4) is secure against a-posteriori chosen-ciphertext key-leakage attacks.

Let $\mathbb{G}$ be a group of prime order q, let $\lambda = \lambda(n)$ be the leakage parameter, let $\mathsf{Ext} : \mathbb{G} \times \{0,1\}^t \to \{0,1\}^m$ be an average-case $(\log q - \lambda, \epsilon)$-strong extractor for some negligible $\epsilon = \epsilon(n)$, and let $\mathcal{H}$ be a family of universal one-way hash functions $H : \mathbb{G}^3 \to \mathbb{Z}_q$. Rompel [42] showed that universal one-way hash functions can be constructed from one-way functions, and in particular, such functions exist based on the hardness of DDH.

We present an encryption scheme with a secret key of size essentially $6 \log q$ bits (six group elements), and show that the scheme is secure for any leakage of length $\lambda \leq \log q - \omega(\log n) - m$, where m is the length of the plaintext. The following describes the encryption scheme:

- **Key generation:** Choose $x_1, x_2, y_1, y_2, z_1, z_2 \in \mathbb{Z}_q$, $g_1, g_2 \in \mathbb{G}$, and $H \in \mathcal{H}$ uniformly at random. Let $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, $h = g_1^{z_1} g_2^{z_2}$, and output the keys

$$SK = (x_1, x_2, y_1, y_2, z_1, z_2) \ , \quad PK = (g_1, g_2, c, d, h, H) \ .$$

- **Encryption:** On input a message $M \in \{0,1\}^m$, choose $r \in \mathbb{Z}_q$ and $s \in \{0,1\}^t$ uniformly at random, and compute

$$u_1 = g_1^r, \quad u_2 = g_2^r, \quad e = \mathsf{Ext}(h^r, s) \oplus M, \quad \alpha = H(u_1, u_2, s, e), \quad v = c^r d^{r\alpha} \ .$$

  Output the ciphertext (u_1, u_2, v, s, e).

- **Decryption:** On input a ciphertext (u_1, u_2, v, s, e) compute $\alpha = H(u_1, u_2, s, e)$, and if $v = u_1^{x_1+y_1\alpha} u_2^{x_2+y_2\alpha}$ then output the message

  $$M = e \oplus \mathsf{Ext}(u_1^{z_1} u_2^{z_2}, s) \ .$$

  Otherwise output $\perp$.

Correctness. For any sequence of coin tosses of the key generation and encryption algorithms it holds that $u_1^{x_1+y_1\alpha} u_2^{x_2+y_2\alpha} = c^r d^{r\alpha} = v$ and that $u_1^{z_1} u_2^{z_2} = h^r$, and therefore the decryption algorithm is always correct.

Proof of security. As in Section 6.2, the proof of security is based on augmenting the proof of Cramer and Shoup with the ideas presented in Section 4. Specifically, we show that any efficient adversary that breaks the security of the scheme can be used to either distinguish between a DH instance and a non-DH instance or to break the security of the universal one-way hash functions. We prove the following theorem:

Theorem 6.10. *Assuming the hardness of DDH, the above encryption scheme is semantically-secure against a-posteriori chosen-ciphertext* $(L/6 - \omega(\log n) - m)$*-key-leakage attacks, where* n *denotes the security parameter,* $L = L(n)$ *denotes the length of the secret key and* $m = m(n)$ *denotes the length of the plaintext.*

Proof. Given an adversary $\mathcal{A}$ consider the algorithm $\mathcal{A}'$ that on input $(g_1, g_2, u_1, u_2) \in \mathbb{G}^4$ acts as follows:

1. $\mathcal{A}'$ chooses $x_1, x_2, y_1, y_2, z_1, z_2 \in \mathbb{Z}_q$ and $H \in \mathcal{H}$ uniformly at random, and sets $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, $h = g_1^{z_1} g_2^{z_2}$, $SK = (x_1, x_2, y_1, y_2, z_1, z_2)$, and $PK = (g_1, g_2, c, d, h, H)$. Then $\mathcal{A}'$ invokes $\mathcal{A}$ with input PK.

2. $\mathcal{A}'$ simulates the leakage oracle and the decryption oracle to $\mathcal{A}$ using SK.

3. When $\mathcal{A}$ outputs two messages M_0 and M_1, the simulator $\mathcal{A}'$ chooses $b \in \{0,1\}$ and $s \in \{0,1\}^t$ independently and uniformly at random, computes

   $$e = \mathsf{Ext}(u_1^{z_1} u_2^{z_2}, s) \oplus M, \quad \alpha = H(u_1, u_2, s, e), \quad v = u_1^{x_1+y_1\alpha} u_2^{x_2+y_2\alpha} \ ,$$

   and sends $\mathcal{A}$ the challenge ciphertext (u_1, u_2, v, s, e).

4. If $\mathcal{A}$ outputs b then $\mathcal{A}'$ outputs 1, and otherwise $\mathcal{A}'$ outputs 0.

An immediate observation is that if (g_1, g_2, u_1, u_2) is a DH instance, then the simulation is identical to the actual attack (and therefore $\mathcal{A}$ should have a non-negligble advantage in outputting the bit b).

Claim 6.11. *If* (g_1, g_2, u_1, u_2) *is a DH instance then* $\mathcal{A}$*'s view is identical to the actual attack.*

Proof. The actual attack and the simulated attack are identical except for the challenge ciphertext. It remains to prove that the challenge ciphertext has the correct distribution when (g_1, g_2, u_1, u_2) is a DH instance. Indeed, in this case $u_1 = g_1^r$ and $u_2 = g_2^r$ for some randomly chosen $r \in \mathbb{Z}_q$, and therefore $u_1^{x_1+y_1\alpha} u_2^{x_2 y_1 \alpha} = c^r d^{r\alpha}$ and $u_1^{z_1} u_2^{z_2} = h^r$ as it should be. ■

The more interesting part of the proof is devoted for showing that if (g_1, g_2, u_1, u_2) is a non-DH instance then $\mathcal{A}$ has only a negligible advantage in outputting the bit b. From now on we assume that (g_1, g_2, u_1, u_2) is a non-DH instance, where $\log_{g_1}(u_1) = r_1$, $\log_{g_2}(u_2) = r_2$, and $r_1 \neq r_2$.

We denote by $(u_1^*, u_2^*, v^*, s^*, e^*)$ the challenge ciphertext given to $\mathcal{A}$, and denote by $\mathsf{Collision}$ the event in which for one of $\mathcal{A}$'s decryption queries (u_1, u_2, v, s, e) it holds that $(u_1, u_2, s, e) \neq (u_1^*, u_2^*, s^*, e^*)$ and $H(u_1, u_2, s, e) = H(u_1^*, u_2^*, s^*, e^*)$. We say that a ciphertext (u_1', u_2', v', s', e') is *invalid* if $\log_{g_1}(u_1') \neq \log_{g_2}(u_2')$.

In what follows we prove that if the event $\mathsf{Collision}$ does not occur then $\mathcal{A}$ has only a negligible advantage in outputting the bit b. Specifically, we prove that: (1) if the event $\mathsf{Collision}$ does not occur then the decryption oracle rejects all invalid ciphertexts except with a negligible probability, and (2) if the decryption oracle rejects all invalid ciphertexts then $\mathcal{A}$ has only a negligible advantage in outputting the bit b. We conclude by proving that the event $\mathsf{Collision}$ has a negligible probability due to the security of the family $\mathcal{H}$ of universal one-way hash functions.

Claim 6.12. *If (g_1, g_2, u_1, u_2) is a non-DH instance and the event $\mathsf{Collision}$ does not occur, then the decryption algorithm rejects all invalid ciphertext except with a negligible probability.*

Proof. We prove the claim by considering the distribution of the point $(x_1, x_2, y_1, y_2) \in \mathbb{Z}_q^4$ from the adversary's point of view. Ignoring the leakage functions for now, the adversary is given the public key (g_1, g_2, c, d, h, H) and the challenge ciphertext (u_1, u_2, v, s, e), and therefore from the adversary's point of view the point (x_1, x_2, y_1, y_2) is uniformly random subject to

$$\begin{cases} \log_{g_1}(c) &= x_1 + \gamma x_2 \\ \log_{g_1}(d) &= y_1 + \gamma y_2 \\ \log_{g_1}(v) &= r_1 x_1 + r_2 \gamma x_2 + \alpha r_1 y_1 + \alpha r_2 \gamma y_2 \end{cases} ,$$

where $\gamma = \log_{g_1}(g_2)$. Note that by submitting valid ciphertexts to the decryption oracle the adversary does not learn any more information on (x_1, x_2, y_1, y_2) (in fact, by submitting a valid ciphertext the adversary only learns a linear combination of the constraint $\log_{g_1}(h) = z_1 + \gamma z_2$ that is already known from the public key).

Denote by $(u_1', u_2', v', s', e') \neq (u_1, u_2, v, s, e)$ the first invalid ciphertext submitted by the adversary, where $u_1' = g_1^{r_1'}$, $u_2' = g_2^{r_2'}$, $r_1' \neq r_2'$, and $\alpha' = H(u_1', u_2', s', e')$. Denote by view the view of the adversary prior to submitting the invalid ciphertext. The adversary learns at most λ bits of leakage, and therefore

$$\widetilde{\mathrm{H}}_\infty\left((x_1, x_2, y_1, y_2) | \mathsf{view}\right) \geq \log q - \lambda \ .$$

In particular, the definition of average min-entropy implies that prior to submitting the invalid ciphertext the probability of $\mathcal{A}$ in guessing (x_1, x_2, y_1, y_2) is at most $2^{-\widetilde{\mathrm{H}}_\infty((x_1,x_2,y_1,y_2)|\mathsf{view})} \leq 2^\lambda / q$. There are three cases to consider:

1. $(u_1', u_2', s', e') = (u_1, u_2, s, e)$. In this case $\alpha' = \alpha$ but $v \neq v'$ and therefore the decryption algorithm rejects.

2. $(u_1', u_2', s', e') \neq (u_1, u_2, s, e)$ and $\alpha' = \alpha$. This is impossible since we assume that the event $\mathsf{Collision}$ does not occur.

3. $(u_1', u_2', s', e') \neq (u_1, u_2, s, e)$ and $\alpha' \neq \alpha$. In this case, if the decryption algorithm accepts the invalid ciphertext then we obtain the following linear equations:

$$\begin{cases} \log_{g_1}(c) &= x_1 + \gamma x_2 \\ \log_{g_1}(d) &= y_1 + \gamma y_2 \\ \log_{g_1}(v) &= r_1 x_1 + r_2 \gamma x_2 + \alpha r_1 y_1 + \alpha r_2 \gamma y_2 \\ \log_{g_1}(v') &= r_1' x_1 + r_2' \gamma x_2 + \alpha' r_1' y_1 + \alpha' r_2' \gamma y_2 \end{cases} ,$$

As long as $\gamma^2(r_1 - r_2)(r'_1 - r'_2)(\alpha - \alpha') \neq 0$ these equations are linearly independent, and therefore the adversary can be used to guess (x_1, x_2, y_1, y_2). Thus, the probability that the decryption algorithm accepts the first invalid ciphertext is at most $2^\lambda/q$.

An almost identical argument holds for all the subsequent invalid decryption queries. The only difference is that each time the decryption oracle rejects an invalid ciphertext the adversary can rule out one more value of (x_1, x_2, y_1, y_2). This shows that the decryption algorithm accepts the i-th invalid ciphertext with probability at most $2^\lambda/(q - i + 1)$. The claim now follows from the fact that the number of decryption queries is polynomial, and from the restriction $\lambda \leq \log q - \omega(\log n)$. ■

Claim 6.13. *If (g_1, g_2, u_1, u_2) is a non-DH instance and the decryption algorithm rejects all invalid ciphertexts, then $\mathcal{A}$ has only a negligible advantage in outputting the bit b.*

Proof. We show that if (g_1, g_2, u_1, u_2) is a non-DH instance and the decryption algorithm rejects all invalid ciphertexts then with overwhelming probability the value $u_1^{z_1} u_2^{z_2}$ has average min-entropy at least $\log q - \lambda \geq m + \omega(\log n)$ given all the other values in the adversary's view. The strong extractor then guarantees that the part of the challenge ciphertext that depends on the bit b is ϵ-close to uniform given the adversary's view, for a negligible $\epsilon = \epsilon(n)$.

We prove the claim by considering the distribution of the point $(z_1, z_2) \in \mathbb{Z}_q$ from the adversary's point of view. The adversary is given the public key (g_1, g_2, c, d, h, H) and therefore from the adversary's point of view the point (z_1, z_2) is uniformly random subject to $\log_{g_1}(h) = z_1 + \gamma z_2$, where $\gamma = \log_{g_1}(g_2)$. We assume that the decryption algorithm rejects all invalid ciphertext, and note that by submitting valid ciphertexts to the decryption oracle the adversary does not learn any more information (the adversary only learns a linear combination of the constraint $\log_{g_1}(h) = z_1 + \gamma z_2$). Thus, the adversary does not learn any information on (z_1, z_2) via decryption queries.

Let $u_1 = g_1^{r_1}$, $u_2 = g_2^{r_2}$, and denote by aux the output of all leakage functions chosen by the adversary. Then from the adversary's point of view it holds that

$$\widetilde{\mathrm{H}}_\infty \left(u_1^{z_1} u_2^{z_2} \mid g_1, g_2, c, d, h, H, \mathsf{aux}, u_1, u_2 \right) = \widetilde{\mathrm{H}}_\infty \left(r_1 z_1 + \gamma r_2 z_2 \mid \gamma, c, d, h, H, \mathsf{aux}, r_1, r_2 \right) \quad .$$

Note that as long as $\gamma(r_1 - r_2) \neq 0$, then for any $w = u_1^{z_1} u_2^{z_2}$ the following two equations (in z_1 and z_2) are linearly independent:

$$\begin{cases} \log_{g_1}(h) &= z_1 + \gamma z_2 \\ \log_{g_1}(w) &= r_1 z_1 + \gamma r_2 z_2 \end{cases}$$

which implies that given (γ, h, r_1, r_2) the function $(z_1, z_2) \to r_1 z_1 + \gamma r_2 z_2$ is injective. Applying an injective function to a distribution preserves its min-entropy, and therefore

$$\begin{aligned} \widetilde{\mathrm{H}}_\infty \left(r_1 z_1 + \gamma r_2 z_2 \mid \gamma, c, d, h, H, \mathsf{aux}, r_1, r_2 \right) &= \widetilde{\mathrm{H}}_\infty \left((z_1, z_2) \mid \gamma, c, d, h, H, \mathsf{aux}, r_1, r_2 \right) \\ &= \widetilde{\mathrm{H}}_\infty \left((x_1, x_2) \mid \gamma, c, d, h, H, \mathsf{aux} \right) \quad , \end{aligned}$$

where the second equality follows from the fact that $(z_1, z_2, \gamma, c, d, h, H, \mathsf{aux})$ are all chosen independently of (r_1, r_2). Finally, the leakage aux obtains at most 2^λ possible output values, and therefore (see Lemma 2.1)

$$\begin{aligned} \widetilde{\mathrm{H}}_\infty \left((z_1, z_2) \mid \gamma, c, d, h, H, \mathsf{aux} \right) &\geq \mathrm{H}_\infty \left((z_1, z_2) \mid \gamma, c, d, h, H \right) - \lambda \\ &= \mathrm{H}_\infty \left((z_1, z_2) \mid \gamma, z_1 + \gamma z_2 \right) - \lambda \\ &\geq \log q - \lambda \ . \end{aligned}$$

Combining all of the above, with overwhelming probability it holds that

$$\widetilde{\mathrm{H}}_\infty\left(u_1^{z_1}u_2^{z_2} \mid g_1, g_2, c, d, h, H, \mathsf{aux}, u_1, u_2\right) \geq \log q - \lambda \enspace .$$

■

Claim 6.14. *If (g_1, g_2, u_1, u_2) is a non-DH instance then the event* Collision *occurs with only a negligible probability.*

Proof. Given an adversary $\mathcal{A}$ for which the event Collision occurs with only a negligible probability, we construct an algorithm $\mathcal{A}''$ that breaks the security of the universal one-way hash functions. $\mathcal{A}''$ is essentially identical to $\mathcal{A}'$ that was described above, except for choosing (u_1, u_2, s, e) before the function H is chosen, where $e \in \{0,1\}^m$ is chosen uniformly at random instead of as a function of the messages M_0 and M_1. We claim, however, that as long as the event Collision does not occur then $\mathcal{A}$ cannot distinguish between $\mathcal{A}'$ and $\mathcal{A}''$ (the argument is identical to the proofs of Claims 6.12 and 6.13). Therefore, $\mathcal{A}$ helps $\mathcal{A}''$ to find a collision with $H(u_1, u_2, s, e)$ with a non-negligible probability.

Formally, consider the following algorithm $\mathcal{A}''$ that attacks the family $\mathcal{H}$ of universal one-way hash functions:

1. $\mathcal{A}''$ chooses $(g_1, g_2, u_1, u_2) \in \mathbb{G}^4$, $s \in \{0,1\}^t$ and $e \in \{0,1\}^m$ uniformly at random, and announces (u_1, u_2, s, e).

2. $\mathcal{A}''$ is given as input a randomly chosen function $H \in \mathcal{H}$.

3. $\mathcal{A}''$ chooses $x_1, x_2, y_1, y_2, z_1, z_2 \in \mathbb{Z}_q$ independently and uniformly at random, and sets $c = g_1^{x_1}g_2^{x_2}$, $d = g_1^{y_1}g_2^{y_2}$, $h = g_1^{z_1}g_2^{z_2}$, $SK = (x_1, x_2, y_1, y_2, z_1, z_2)$, and $PK = (g_1, g_2, c, d, h, H)$. Then $\mathcal{A}'$ sends PK to $\mathcal{A}$.

4. $\mathcal{A}'$ simulates the leakage oracle and the decryption oracle to $\mathcal{A}$ using SK.

5. In the challenge phase $\mathcal{A}'$ ignores the two messages $M_0, M_1 \in \{0,1\}^m$, computes

$$\alpha = H(u_1, u_2, s, e), \quad v = u_1^{x_1+y_1\alpha}u_2^{x_2+y_2\alpha} \enspace ,$$

   and sends $\mathcal{A}$ the challenge ciphertext (u_1, u_2, v, s, e).

6. If at some point $\mathcal{A}$ submits a decryption query (u_1', u_2', v', s', e') such that $(u_1', u_2', s', e') \neq (u_1, u_2, s, e)$ and $H(u_1', u_2', s', e') = H(u_1, u_2, s, e)$ then $\mathcal{A}''$ outputs (u_1', u_2', s', e'). Otherwise $\mathcal{A}''$ outputs $\perp$.

Claim 6.12 guarantees that as long as the event Collision does not occur, then the decryption algorithm rejects all invalid ciphertexts except with a negligible probability. Claim 6.13 then guarantees that as long as the decryption algorithm rejects all invalid ciphertexts then $\mathcal{A}$ cannot distinguish between $\mathcal{A}'$ and $\mathcal{A}''$. Specifically, both in the execution with $\mathcal{A}'$ and in the execution with $\mathcal{A}''$ the component in the challenge ciphertext that depends on the bit b is ϵ-close to random given the adversary's view (for some negligible $\epsilon = \epsilon(n)$). Therefore, with a non-negligible probability $\mathcal{A}$ submits a ciphertext (u_1', u_2', v', s', e') such that $(u_1', u_2', s', e') \neq (u_1, u_2, s, e)$ and $H(u_1', u_2', s', e') = H(u_1, u_2, s, e)$, and in this case $\mathcal{A}''$ finds a collision. ■

This concludes the proof of Theorem 6.10. ■

7 Protecting Against Weak Key-Leakage Attacks

In this section we consider weak key-leakage attacks (see Definition 3.4), in which a leakage function with output length λ is chosen by the adversary ahead of time without any knowledge of the public key. We present a generic construction that transforms any encryption scheme to one that is resilient to any weak leakage of $L(1 - o(1))$ bits, where L is the length of the secret key. The resulting scheme is essentially as efficient as the original one, and does not rely on additional computational assumptions.

As discussed in Section 1.1, although this notion of key leakage seems very limited, it still captures many realistic attacks in which the leakage does not depend on the parameters of the encryption scheme. Specifically, this notion captures the cold boot attack of Halderman et al. [22], in which the leakage depends only on the properties of the hardware devices that are used for storing the secret key.

Let $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ be any public-key encryption scheme, and denote by $m = m(n)$ the length of the random string used by $\mathcal{G}(1^n)$. Given a leakage parameter $\lambda = \lambda(n)$, let $\mathsf{Ext} : \{0,1\}^{k(n)} \times \{0,1\}^{t(n)} \to \{0,1\}^{m(n)}$ be an average-case $(k - \lambda, \epsilon)$-strong extractor for some negligible $\epsilon = \epsilon(n)$. Consider the encryption scheme $\Pi_\lambda = (\mathcal{G}_\lambda, \mathcal{E}_\lambda, \mathcal{D}_\lambda)$ defined as follows:

- **Key generation:** On input 1^n choose $x \in \{0,1\}^{k(n)}$ and $s \in \{0,1\}^{t(n)}$ uniformly at random, and compute $(pk, sk) = \mathcal{G}(\mathsf{Ext}(x, s))$. Output $PK = (pk, s)$ and $SK = x$.

- **Encryption:** On input a message M and a public-key $PK = (pk, s)$, choose $r \in \{0,1\}^*$ uniformly at random and output $(\mathcal{E}(pk, M; r), s)$.

- **Decryption:** On input a ciphertext (c, s) and a secret key $SK = x$, compute $(pk, sk) = \mathcal{G}(\mathsf{Ext}(x, s))$ and output $\mathcal{D}(sk, c)$.

The following theorem states that if Π is semantically secure, then Π_λ is resilient to any weak key leakage of λ bits.

Theorem 7.1. *Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a semantically secure public-key encryption scheme. Then, for any polynomial $\lambda = \lambda(n)$ the scheme $(\mathcal{G}_\lambda, \mathcal{E}_\lambda, \mathcal{D}_\lambda)$ is semantically secure against weak λ-key-leakage attacks.*

Proof. We show that for any efficient adversary $\mathcal{A}$ and ensemble $\mathcal{F}$ of efficiently computable leakage functions, there exists an efficient adversary $\mathcal{A}'$ such that

$$\mathsf{Adv}^{\mathsf{WeakLeakage}}_{\Pi_\lambda, \mathcal{A}, \mathcal{F}}(n) \leq \mathsf{Adv}^{\mathsf{CPA}}_{\Pi, \mathcal{A}'}(n) + 2\epsilon(n) \ .$$

For $b \in \{0,1\}$ consider the following experiment denoted $\mathsf{Expt}_{\Pi, \mathcal{A}, \mathcal{F}}(b)$:

1. Choose $x \in \{0,1\}^{k(n)}$, $s \in \{0,1\}^{t(n)}$, and $y \in \{0,1\}^{m(n)}$ uniformly at random. Compute $(pk, sk) = \mathcal{G}(y)$. Let $PK = (pk, s)$ and $SK = sk$.

2. $(M_0, M_1, \mathsf{state}) \quad \mathcal{A}_1(PK, f_n(x))$ such that $|M_0| = |M_1|$.

3. $C \quad \mathcal{E}_{pk}(M_b)$.

4. $b' \quad \mathcal{A}_2(C, \mathsf{state})$

5. Output b'.

Using the notation of Definition 3.4 and the triangle inequality, for any adversary $\mathcal{A}$ it holds that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{WeakLeakage}}_{\Pi_\lambda,\mathcal{A},\mathcal{F}}(n) &= \left|\Pr\left[\mathsf{Expt}^{\mathsf{WeakLeakage}}_{\Pi_\lambda,\mathcal{A},\mathcal{F}}(0)=1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{WeakLeakage}}_{\Pi_\lambda,\mathcal{A},\mathcal{F}}(1)=1\right]\right| \\ &\leq \left|\Pr\left[\mathsf{Expt}^{\mathsf{WeakLeakage}}_{\Pi_\lambda,\mathcal{A},\mathcal{F}}(0)=1\right] - \Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A},\mathcal{F}}(0)=1\right]\right| \qquad (7.1) \\ &\quad + \left|\Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A},\mathcal{F}}(0)=1\right] - \Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A},\mathcal{F}}(1)=1\right]\right| \qquad (7.2) \\ &\quad + \left|\Pr\left[\mathsf{Expt}_{\Pi,\mathcal{A},\mathcal{F}}(1)=1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{WeakLeakage}}_{\Pi_\lambda,\mathcal{A},\mathcal{F}}(1)=1\right]\right| \ . \qquad (7.3)\end{aligned}$$

The experiment $\mathsf{Expt}_{\Pi,\mathcal{A},\mathcal{F}}(b)$ is identical to the experiment $\mathsf{Expt}^{\mathsf{WeakLeakage}}_{\Pi_\lambda,\mathcal{A},\mathcal{F}}(b)$ other than the fact that the key generation algorithm $\mathcal{G}$ is invoked on a truly random input y and not on input $\mathsf{Ext}(x, s)$. Note, however, that the average min-entropy of x given $f_n(x)$ is at least $k - \lambda$, and therefore the strong extractor guarantees that the statistical distance between the views of the adversary in these two experiments is at most ϵ. A crucial point here is that the leakage function is independent of the public key, and specifically, independent of the seed s. This implies that each of the terms (7.1) and (7.3) is upper bounded by ϵ.

In addition, it is rather straightforward that there exists a CPA-adversary $\mathcal{A}'$ for which the term (7.2) is upper bounded by $\mathsf{Adv}^{\mathsf{CPA}}_{\Pi,\mathcal{A}'}(n)$, and this concludes the proof. ■

Generalization to hard-to-invert leakage. So far in this paper we considered leakage attacks in which the secret key is information-theoretically unpredictable given the leakage information. One can also consider leakage attacks in which the secret key is only *computationally* unpredictable. That is, the leakage may completely determine the secret key, but it is computationally hard to actually recover the secret key. Such a notion of key leakage was studied by Dodis, Tauman Kalai, and Lovett [13] in the setting of symmetric-key encryption. They introduced a new computational assumption that is a generalization of learning parity with noise, and constructed symmetric-key encryption schemes that are resilient to leakage of the form $f(sk)$, where f is any exponentially-hard one-way function.

The above construction for protecting against weak key-leakage attacks can be generalized in a straightforward manner to provide protection against any leakage that is sufficiently hard to invert. Specifically, the only modification to the above construction is that the strong extractor $\mathsf{Ext}(x, s)$ is replaced by Goldreich-Levin hard-core bits $\mathsf{GL}(x, s)$ [18] (recall that x is the secret key, and s is part of the public key). Thus, the resulting encryption scheme is secure with respect to any leakage of the form $f(x)$, where $\mathsf{GL}(x, s)$ is a string of m^ϵ bits (for some constant $\epsilon > 0$) that is hard core for f (note that a pseudorandom generator can be applied to $\mathsf{GL}(x, s)$ to obtain a pseudorandom string of length m to be used by the key-generation algorithm). In particular, this is known to be satisfied by any $2^{\Omega(m^\epsilon)}$-hard one-way function $f : \{0, 1\}^m \to \{0, 1\}^{m'}$ (see, for example, [18, Corollary 1]).

8 Protecting Against Generalized Forms of Key-Leakage Attacks

In this section we present several generalizations of key-leakage attacks that extend the framework presented in Section 3, and show that they are satisfied by our schemes.

8.1 Noisy Leakage

In the side-channel attack of Halderman et al. [22] the adversary learns a noisy version of all of the memory. This is a more general scenario than the scenario captured by Definition 3.1: The leakage is not of bounded length, but it is guaranteed that the secret key is still somewhat unpredictable given the leakage. This motivates a realistic generalization that allows the adversary to learn any random variable W (representing the leakage information) for which the conditional mutual information (adapted to average min-entropy[8]) of the secret key and W given the public key is at most λ. That is, the adversary can learn any random variable W for which

$$\widetilde{\mathrm{I}}_{\infty}\left(SK; W|PK\right) \stackrel{\mathsf{def}}{=} \widetilde{\mathrm{H}}_{\infty}\left(SK|PK\right) - \widetilde{\mathrm{H}}_{\infty}\left(SK|PK, W\right) \leq \lambda$$

In the information-theoretic setting, this generalization does not necessarily strengthen the definition, since W may be compressed to essentially λ bits. However, in the computational setting (which is the setting we consider in this work) we can conjecture that this notion is stronger[9].

Our generic construction from hash proof systems satisfies this notion of security assuming an additional property of the underlying hash proof system (that is satisfied by all the constructions in this paper). Informally, using the notation introduced in Section 2.3, the required property is that with overwhelming probability over the choice of an invalid ciphertext $C \in \mathcal{C} \setminus \mathcal{V}$, there is no substantial gap between the min-entropy of the secret key $sk \in \mathcal{SK}$ and the min-entropy of the encapsulated key $\Lambda_{sk}(C) \in \mathcal{K}$, both conditioned on the public key $pk \in \mathcal{PK}$. For example, for the hash proof systems underlying the schemes in Sections 4.2 and 5.1 it holds that $\widetilde{\mathrm{I}}_{\infty}\left(\Lambda_{sk}(C); W|pk\right) = \widetilde{\mathrm{I}}_{\infty}\left(sk; W|pk\right)$, and this suffices for the same proofs of security to go through (a slightly more subtle property holds for the hash proof system underlying the scheme in Section 5.2). In addition, it is rather straightforward to verify that our CCA-secure constructions (both the generic construction and the two specific constructions) and the generic construction for protecting against weak leakage attacks satisfy this notion of security.

8.2 Leakage of Intermediate Values from the Key-Generation Process

Definition 3.1 assumes that the adversary does not learn any of the intermediate values that occur during the generation of the secret and public keys. In practice, however, this is not always a valid assumption. Specifically, in the attack of Halderman et al. [22] the adversary learns a noisy version of all of the memory, and it is rather likely that intermediate values from the generation of the keys are not always completely erased. This motivates a natural generalization that allows the adversary to learn functions of the random bits that are used by the key generation algorithm. Encryption schemes that satisfy this notion of security are more robust to leakage in the sense that the key generation algorithm does not have to make sure that all intermediate key-related values have been deleted. In addition, this generalization is especially important to security under composition of cryptographic primitives. For example, the key generation algorithm may use random bits (or pseudorandom bits) that are the output of another primitive (say, a pseudorandom generator) which may also suffer from unintended leakage of sensitive information.

All of our constructions in this paper, except for the generic CCA-secure construction in Section 6.1 (see comment below), are secure according to this generalization. Informally, the proofs of security of our schemes (other than the generic CCA-secure scheme) rely on an indistinguishability argument over the challenge ciphertext and not over the public key. As a result, the simulators in the proofs of security always generate the secret key and public key by themselves, and therefore they know the random bits used by the key generation algorithm and can simulate the leakage oracle to the adversary. A minor comment is that the simulators do not always generate the

[8] For Shannon entropy, the conditional mutual information of X and Y given Z is defined as $I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$.

[9] See, as an *analogy*, the trade-off presented by Harsha et al. [23] between communication complexity and computation.

public parameters of the hash proof system by themselves. These public parameters, however, can be shared among all users and therefore they do not have to be included in the key generation algorithm. For example, when instantiating the generic construction presented in Section 4.1 with the Paillier-based hash proof system of Cramer and Shoup [8], the public parameters include a modulus $N = PQ$. In this case, as demonstrated by Rivest and Shamir [41], an adversary that obtains leakage information on the random bits used to generate N can efficiently compute its factorization and completely compromise the security of the scheme. However, the factorization of N is not needed for the encryption scheme (i.e., it is not part of the secret key), and therefore it is possible to assume that N is publicly known and is not part of the key generation algorithm (exactly as suggested in [8]).

The generic CCA-secure construction in Section 6.1 can be made secure against such attacks by requiring two additional simple properties from its building blocks. First, the public key pk_1 should be sampled without producing a corresponding secret key (note that the secret key of the resulting scheme does not include sk_1 anyway). That is, there should be an efficient algorithm that generates pk_1, such that the scheme is secure even if the adversary knows the random bits used for generate pk_1. This property is satisfied by the schemes in Sections 4.2, 5.1, and 5.2. Moreover, we note that in fact pk_1 does not have to come from an encryption scheme that is resilient to key-leakage attacks – only pk_0 has to be resilient to key-leakage attacks (as shown by the proofs of Claims 6.4 and 6.5), and therefore any public-key encryption scheme with such an oblivious sampling algorithm can be used for pk_1. Second, the NIZK proof system should also have such a property with respect to the common reference string. That is, the NIZK proof system should maintain its security even if the adversary knows the random bits used for generating the common reference string (note that this should hold only for the "real" reference string, and not for the "simulated" one). In particular, this is satisfied by any NIZK proof system in which the common reference string is a uniformly random string. A similar comment is in place also for the CCA2-secure construction in Section 6.3: the universal one-way hash functions should be secure even if the adversary knows the random bits used for generating the description of a function. In general, this is a seemingly stronger requirement (see [25]), but in practice this is not a concern since the universal one-way hash functions can be instantiated with a fixed hash function, such as SHA-256.

8.3 Keys Generated using Weak Random Sources

When considering leakage of the random bits that are used by the key generation algorithm (as in Section 8.2), then from the adversary's point of view these bits are uniformly distributed subject to the leakage information. A natural generalization in this setting is to consider cases in which the keys are generated using a weak source of random bits. This is relevant, in particular, in light of crucial security vulnerabilities that were recently identified in pseudorandom generators that are used by many systems [15, 21, 47].

Formally, we consider the following security experiment. First, a global algorithm generates public parameters for the encryption scheme. These parameters are generated using truly random (or pseudorandom) bits, and are shared among all users. Next, we allow the adversary to specify a distribution of random bits that will be used to generate the secret key and public key of a specific user. The adversary can choose this distribution as a function of the public parameters, and the only restriction is a predetermined lower bound on the min-entropy of the distribution. Then, the experiment continues as in the standard indistinguishability of encryptions experiment.

Our schemes in Sections 4.2, 5.1, 5.2, and 6.2 satisfy this notion of security. For simplicity, we exemplify this fact using the scheme from Section 4.2, and note that ideas we present here easily extend to the other schemes[10]. The public parameters of the scheme consist of a description of a group $\mathbb{G}$ of prime order q, and two generators $g_1, g_2 \in \mathbb{G}$ that are chosen uniformly at random (these are the public parameters of the corresponding hash proof system). The secret key of the scheme is a pair $(x_1, x_2) \in \mathbb{Z}_q^2$, and we allow the adversary to specify the distribution from which the secret key is sampled. The adversary can choose this distribution as a function of the public parameters, and we only require that it has min-entropy at least $\log q + \omega(\log n) + m$, where m is the length of plaintexts (note that this is rather far from the uniform distribution over $\mathbb{Z}_q^2$ for a wide range of parameters). The public key is then computed as $h = g_1^{x_1} g_2^{x_2}$. For the proof of security, we observe that if (g_1, g_2, u_1, u_2) is not a Diffie-Hellman tuple (i.e., $u_1 = g_1^{r_1}$, $u_2 = g_2^{r_2}$, and $r_1 \neq r_2$), then it holds that

$$\widetilde{\mathrm{H}}_\infty \left(u_1^{x_1} u_2^{x_2} | h\right) = \widetilde{\mathrm{H}}_\infty \left((x_1, x_2) | h\right) \geq \mathrm{H}_\infty \left(x_1, x_2\right) - \log q = m + \omega(\log n)$$

Thus, the average-case strong-extractor Ext guarantees that $\mathsf{Ext}(u_1^{x_1} u_2^{x_2}, s)$ statistically masks the m-bit challenge plaintext.

8.4 Leakage of Intermediate Values from the Decryption Process

An additional strengthening of Definition 3.1 is to consider leakage that may occur during computation, and not only leakage from the stored key. Specifically, an invocation of the decryption algorithm may produce various *intermediate* values, whose leakage may compromise the security of the scheme even if the scheme is robust against leakage from the stored key. Such a notion of security is generically guaranteed when considering leakage of fixed bounded length (as in Definition 3.1). However, it is not always guaranteed when the adversary obtains all of the memory in a noisy fashion (as discussed in Section 8.1).

Consider the seemingly contrived example of a decryption algorithm that first encodes the secret key using a good error-correcting code, and then performs the actual decryption. In this case, an adversary that obtains a noisy variant of the memory can clearly recover the secret key. This example, however, is not so contrived, since as demonstrated by Halderman et al., encryption schemes typically compute intermediate key-related values whose representation is rather redundant, and this can be used to attack the scheme. Moreover, even if the encryption scheme itself does not explicitly instructs to compute intermediate values, it may be the case that such values are computed by a specific implementation of the encryption scheme.

Although when presenting our encryption schemes in this paper we do not specify exact implementations of their decryption algorithms, we point out that they can be implemented while introducing only minimal redundancy. That is, the key-related computation has “low bandwidth”: at any point in time the (natural) decryption algorithm works on a limited part of the key and the intermediate results are also succinct. For example, the natural implementation of the decryption algorithms of the schemes in Section 5 require space that is significantly smaller than the size of their corresponding secret keys: for the scheme in Section 5.2, the length of the secret key is ℓ bits, and the decryption algorithm requires only $O(\log q)$ bits where ℓ is significantly larger than $\log q$, and a similar property is satisfied by the scheme in Section 5.1.

[10] We note that these feasibility results do not contradict the negative result of Dodis et al. [10] on the possibility of public-key encryption with imperfect randomness. Specifically, our generalization circumvents their impossibility result by assuming that truly random bits are available for the encryption algorithm, and that only the keys are generated using imperfect randomness.

References

[1] M. Abe, R. Gennaro, K. Kurosawa, and V. Shoup. Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In *Advances in Cryptology – EUROCRYPT '05*, pages 128–146, 2005.

[2] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *Proceedings of the 6th Theory of Cryptography Conference*, pages 474–495, 2009.

[3] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In *Advances in Cryptology – CRYPTO '97*, pages 513–525, 1997.

[4] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology – CRYPTO '04*, pages 41–55, 2004.

[5] D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of checking cryptographic protocols for faults. In *Advances in Cryptology – EUROCRYPT '97*, pages 37–51, 1997.

[6] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision Diffie-Hellman. In *Advances in Cryptology – CRYPTO '08*, pages 108–125, 2008.

[7] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai. Exposure-resilient functions and all-or-nothing transforms. In *Advances in Cryptology – EUROCRYPT '00*, pages 453–469, 2000.

[8] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *Advances in Cryptology – EUROCRYPT '02*, pages 45–64, 2002.

[9] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

[10] Y. Dodis, S. J. Ong, M. Prabhakaran, and A. Sahai. On the (im)possibility of cryptography with imperfect randomness. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 196–205, 2004.

[11] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.

[12] Y. Dodis, A. Sahai, and A. Smith. On perfect and adaptive security in exposure-resilient cryptography. In *Advances in Cryptology – EUROCRYPT '01*, pages 301–324, 2001.

[13] Y. Dodis, Y. Tauman Kalai, and S. Lovett. On cryptography with auxiliary input. To appear in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, 2009.

[14] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.

[15] L. Dorrendorf, Z. Gutterman, and B. Pinkas. Cryptanalysis of the windows random number generator. In *Proceedings of the 2007 ACM Conference on Computer and Communications Security*, pages 476–485, 2007.

[16] S. Dziembowski and K. Pietrzak. Leakage-resilient cryptography. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 293–302, 2008.

[17] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 197–206, 2008.

[18] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 25–32, 1989.

[19] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[20] S. Goldwasser, Y. Tauman Kalai, and G. N. Rothblum. One-time programs. In *Advances in Cryptology – CRYPTO '08*, pages 39–56, 2008.

[21] Z. Gutterman, B. Pinkas, and T. Reinman. Analysis of the linux random number generator. In *IEEE Symposium on Security and Privacy*, pages 371–385, 2006.

[22] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. Lest we remember: Cold boot attacks on encryption keys. In *Proceedings of the 17th USENIX Security Symposium*, pages 45–60, 2008.

[23] P. Harsha, Y. Ishai, J. Kilian, K. Nissim, and S. Venkatesh. Communication versus computation. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming*, pages 745–756, 2004.

[24] N. Heninger and H. Shacham. Reconstructing RSA private keys from random key bits. In *Advances in Cryptology – CRYPTO '09*, pages 1–17, 2009.

[25] C.-Y. Hsiao and L. Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In *Advances in Cryptology – CRYPTO '04*, pages 92–105, 2004.

[26] Y. Ishai, M. Prabhakaran, A. Sahai, and D. Wagner. Private circuits II: Keeping secrets in tamperable circuits. In *Advances in Cryptology – EUROCRYPT '06*, pages 308–327, 2006.

[27] Y. Ishai, A. Sahai, and D. Wagner. Private circuits: Securing hardware against probing attacks. In *Advances in Cryptology – CRYPTO '03*, pages 463–481, 2003.

[28] A. Joux and K. Nguyen. Separating decision Diffie-Hellman from computational Diffie-Hellman in cryptographic groups. *Journal of Cryptology*, 16(4):239–247, 2003.

[29] E. Kiltz. Chosen-ciphertext secure key-encapsulation based on gap hashed Diffie-Hellman. In *Proceedings of the 10th International Conference on Practice and Theory in Public-Key Cryptography*, pages 282–297, 2007.

[30] E. Kiltz, K. Pietrzak, M. Stam, and M. Yung. A new randomness extraction paradigm for hybrid encryption. In *Advances in Cryptology – EUROCRYPT '09*, pages 590–609, 2009.

[31] P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Advances in Cryptology – CRYPTO '96*, pages 104–113, 1996.

[32] P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Advances in Cryptology – CRYPTO '99*, pages 388–397, 1999.

[33] K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In *Advances in Cryptology – CRYPTO '04*, pages 426–442, 2004.

[34] Y. Lindell. A simpler construction of CCA2-secure public-key encryption under general assumptions. *Journal of Cryptology*, 19(3):359–377, 2006.

[35] U. M. Maurer. On the oracle complexity of factoring integers. *Computational Complexity*, 5(3-4):237–247, 1995.

[36] S. Micali and L. Reyzin. Physically observable cryptography. In *Proceedings of the 1st Theory of Cryptography Conference*, pages 278–296, 2004.

[37] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 427–437, 1990.

[38] C. Petit, F.-X. Standaert, O. Pereira, T. Malkin, and M. Yung. A block cipher based pseudo random number generator secure against side-channel key recovery. In *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, pages 56–65, 2008.

[39] K. Pietrzak. A leakage-resilient mode of operation. In *Advances in Cryptology – EUROCRYPT '09*, pages 462–482, 2009.

[40] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 84–93, 2005.

[41] R. L. Rivest and A. Shamir. Efficient factoring based on partial information. In *Advances in Cryptology – EUROCRYPT '85*, pages 31–34, 1985.

[42] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 387–394, 1990.

[43] A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 543–553, 1999.

[44] H. Shacham. A Cramer-Shoup encryption scheme from the Linear assumption and from progressively weaker Linear variants. Cryptology ePrint Archive, Report 2007/074, 2007.

[45] V. Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology – EUROCRYPT '97*, pages 256–266, 1997.

[46] Y. Tauman Kalai and V. Vaikuntanathan. Public-key encryption schemes with auxiliary inputs and applications, 2009.

[47] S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage. PRNG PR0N: Understanding the Debian OpenSSL debacle, 2008.

A The Matrix *d*-Linear Assumption

Let $\mathcal{G}$ be an algorithm that takes as input a security parameter and outputs a triplet $\mathbb{G} = (G, q, g)$ where G a group of order q that is generated by g. We denote by $\mathrm{Rk}_i(\mathbb{Z}_q^{a\times b})$ the set of all $a \times b$ matrices over $\mathbb{Z}_q$ with rank i. For a matrix $R = \{r_{i,j}\}_{i\in[a],j\in[b]} \in \mathbb{Z}_q^{a\times b}$ we denote by g^R the matrix $\{g_{i,j}\}_{i\in[a],j\in[b]} = \{g^{r_{i,j}}\}_{i\in[a],j\in[b]} \in G^{a\times b}$.

The matrix d-Linear assumption is that for any integers a and b, and for any $d \leq i < j \leq \min\{a, b\}$ the ensembles $\{(\mathbb{G}, g^R)\}_{R\in\mathrm{Rk}_i(\mathbb{Z}_q^{a\times b}),n\in\mathbb{N}}$ and $\{(\mathbb{G}, g^R)\}_{R\in\mathrm{Rk}_j(\mathbb{Z}_q^{a\times b}),n\in\mathbb{N}}$ are computationally indistinguishable, where $\mathbb{G} = (G, q, g) \quad \mathcal{G}(1^n)$. We prove the following lemma (noting that Boneh et al. [6] proved it for $d = 1$):

Lemma A.1. *The d-Linear assumption implies the matrix d-Linear assumption.*

Proof. We show that for any $d \leq i < \min\{a, b\}$ an efficient distinguisher $\mathcal{A}$ that distinguishes between the ensembles $\{(\mathbb{G}, g^R)\}_{R\in\mathrm{Rk}_i(\mathbb{Z}_q^{a\times b}),n\in\mathbb{N}}$ and $\{(\mathbb{G}, g^R)\}_{R\in\mathrm{Rk}_{i+1}(\mathbb{Z}_q^{a\times b}),n\in\mathbb{N}}$ with a non-negligible advantage can be used to construct an efficient distinguisher $\mathcal{A}'$ for the d-Linear problem.

On input $\left(\mathbb{G}, g_1, \ldots, g_{d+1}, g_1^{r_1}, \ldots, g_{d+1}^{r_{d+1}}\right)$ where $\mathbb{G} = (G, q, g) \quad \mathcal{G}(1^n)$, the distinguisher $\mathcal{A}'$ constructs the following matrices:

$$\Phi_1 = \begin{pmatrix} g_1 & g^0 & \cdots & \cdots & g^0 & g_{d+1} \\ g^0 & g_2 & g^0 & \cdots & g^0 & g_{d+1} \\ \vdots & \vdots & \ddots & & \vdots & \vdots \\ \vdots & \vdots & & \ddots & \vdots & \vdots \\ g^0 & \cdots & \cdots & g^0 & g_d & g_{d+1} \\ g_1^{r_1} & g_2^{r_2} & \cdots & g_{d-1}^{r_{d-1}} & g_d^{r_d} & g_{d+1}^{r_{d+1}} \end{pmatrix} \in G^{(d+1)\times(d+1)}$$

$$\Phi_2 = \left(\begin{array}{c|c|c} \Phi_1 & & \\ \hline & \mathsf{Id}^{i-d} & \\ \hline & & 0^{a-i,b-i} \end{array}\right) \in G^{a\times b}$$

where Id^{i-d} is the identity matrix of size $i - d$ of G (i.e., g on the main diagonal and g^0 all other entries), $0^{a-i,b-i}$ is the zero matrix of G of size $(a - i) \times (b - i)$ (i.e., g^0 all entries), and all the other blocks in Φ_2 are zero matrices.

Then, $\mathcal{A}'$ chooses invertible matrices $L \in \mathbb{Z}_q^{a\times a}$ and $R \in \mathbb{Z}_q^{b\times b}$ independently and uniform at random, invokes $\mathcal{A}$ on input $\Phi_3 = L \otimes \Phi_2 \otimes R \in G^{a\times b}$ (where $\otimes$ is naturally defined using $\mathbb{Z}_q$ operations in the exponent), and outputs whatever $\mathcal{A}$ outputs.

In what follows we prove that with overwhelming probability if $r_{d+1} = \sum_{i=1}^{d} r_i$ then $\Phi_3 = g^{\Psi_3}$ for a randomly chosen $\Psi_3 \in \mathrm{Rk}_i(\mathbb{Z}_q^{a\times b})$, and if $r_{d+1} \neq \sum_{i=1}^{d} r_i$ then $\Phi_3 = g^{\Psi_3}$ for a randomly chosen $\Psi_3 \in \mathrm{Rk}_{i+1}(\mathbb{Z}_q^{a\times b})$. For every $1 \leq i \leq d + 1$ let $w_i = \log_g(g_i)$. Then $\Phi_1 = g^{\Psi_1}$ where

$$\Psi_1 = \begin{pmatrix} w_1 & 0 & \cdots & \cdots & 0 & w_{d+1} \\ 0 & w_2 & 0 & \cdots & 0 & w_{d+1} \\ \vdots & \vdots & \ddots & & \vdots & \vdots \\ \vdots & \vdots & & \ddots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & w_d & w_{d+1} \\ w_1 r_1 & w_2 r_2 & \cdots & w_{d-1} r_{d-1} & w_d r_d & w_{d+1} r_{d+1} \end{pmatrix} \in \mathbb{Z}_q^{(d+1)\times(d+1)} .$$

The elements $g_1, \ldots, g_{d+1}$ are chosen uniformly at random, and therefore with overwhelming probability it holds that $\prod_{i=1}^{d+1} w_i \neq 0$. In this case, if $r_{d+1} = \sum_{i=1}^{d} r_i$ then Ψ_1 has rank d, and if $r_{d+1} \neq \sum_{i=1}^{d} r_i$ then Ψ_1 has rank $d+1$. There, the matrix

$$\Psi_2 = \left(\begin{array}{c|c|c} R_1 & & \\ \hline & \mathsf{Id}^{i-d} & \\ \hline & & 0^{a-i,b-i} \end{array}\right) \in \mathbb{Z}_q^{a \times b}$$

either has rank i if $r_{d+1} = \sum_{i=1}^{d} r_i$ or has rank $i+1$ if $r_{d+1} \neq \sum_{i=1}^{d} r_i$. This implies that $\Psi_3 = L\Psi_2 R \in \mathbb{Z}_q^{a \times b}$ is either a uniformly distributed matrix with rank i or a uniformly distributed matrix with rank $i+1$. ■