Public-Key Cryptosystems Resilient to Key Leakage*

Moni Naor** and Gil Segev***

Department of Computer Science and Applied Mathematics,
Weizmann Institute of Science, Rehovot 76100, Israel
{moni.naor,gil.segev}@weizmann.ac.il

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", 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 proof system. Existing constructions of such 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, is resilient to any leakage of $L(1-o(1))$ bits. These two proposed schemes complement each other in terms of efficiency.

* Due to space limitations we refer the reader to a longer version available as [28].

** Incumbent of the Judith Kleeman Professorial Chair. Research supported in part by a grant from the Israel Science Foundation.

*** Research supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities, and by a grant from the Israel Science Foundation.

S. Halevi (Ed.): CRYPTO 2009, LNCS 5677, pp. 18–35, 2009.

– 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.

1 Introduction

Proving the security of a cryptographic scheme consists of two main ingredients: (1) an *adversarial model* that specifies the adversarial access to the system and the adversary's computational capabilities, and (2) a *notion of security* that specifies 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 [15]), the vast majority of cryptographic schemes are analyzed in abstract adversarial models that do not capture *side-channel attacks*. Such attacks exploit unintended leakage of information which is inherent to almost all physical implementations. Over the years side-channel attacks exposed crucial vulnerabilities of systems that are considered secure in standard adversarial models (see, for example, [3,4,23,24]).

Countermeasures for protecting against side-channel attacks are taken on two complementing levels: the hardware level and the software level. Preventing unintended leakage on the hardware level is typically rather inefficient and expensive, and is even impossible in some cases. It is thus highly desirable to protect, as much as possible, against side-channel attacks on the software level by modeling such attacks using abstract notions of computation.

Physically observable cryptography. In their pioneering work, Micali and Reyzin [27] put forward a powerful and 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, [14,16,27,30,31]).

Memory-leakage attacks. Recently, Halderman et al. [18] 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 improved RSA key reconstruction by Heninger and Shacham [19]).

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 [32] is resilient to such key leakage (to an extent that depends on the amount of leakage) by slightly strengthening the computational assumption that is required by the original scheme.

1.1 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 (see below). Moreover, we demonstrate that our approach leads to encryption schemes that are both resilient to significantly large amounts of leakage, and that are very efficient and can be used in practice (see, in particular, the instantiation in Section 4 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 attacks. 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.

In what follows we present a more elaborated exposition of our results, but first, we briefly describe the framework of 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. Akavia et al. showed that Regev's public-key encryption scheme is resilient to any key leakage of $L/\mathrm{polylog}(L)$ bits, where L is the length of the

[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.

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 [7]. The construction does not rely on additional computational assumptions, and the resulting scheme is as efficient as the underlying proof system. Existing constructions of such proof systems [7,22,34] 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. The natural approach for achieving security against partial key leakage is to add redundancy to the private key, so that every (short) function of it will still keep many possibilities for the "real secret". Hash proof systems yield a convenient method for doing just that.

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.

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. [18]), a scenario in which partial results of the decryption process are leaked, and more.

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

[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.

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. [5] fits into our generic approach using a different hash proof system (that satisfies the same weaker universality property). We then compare our two proposals both conceptually and practically, indicating that they complement each other in terms of efficiency.

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 [12,29] 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 [8] (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 [1,22,25]).

"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 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. [18], 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

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

$L(1 - o(1))$ 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 [33] 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 [26] who showed that ϵn questions are sufficient, for any constant $\epsilon > 0$.

Canetti et al. [6] 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. [10] on adaptive security of such functions). Ishai et al. [20,21] 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 [14] and Pietrzak [31] 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 [11] 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 [35] 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. [5] is resilient to key leakage not only when the secret key has sufficient min-entropy given the leakage function (as also shown in this paper in Section 5.2 as a specific instantiation of our generic approach), but also when the leakage function is exponentially hard to invert. 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 is essentially identical to our construction (see [28, Section 6.1]).

1.3 Paper Organization

The remainder of the paper is organized as follows. In Section 2 we present several notions and tools that are used in our constructions. In Section 3 we formally describe the framework of key-leakage 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, and provide a comparison between them. In Section 6 we present several generalizations of the framework considered in this paper that are satisfied by our schemes. Due to space limitations we refer the reader to [28] for our results in the setting of chosen-ciphertext security and weak key-leakage attacks.

2 Preliminaries and Tools

In this section we present some basic notions and tools that are used in our constructions. Specifically, we present the notions of an average-case strong extractor and hash proof systems.

2.1 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. [9] 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:

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

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 [9]:

Lemma 2.1 ([9]). *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 ([9]). *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*

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

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. 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 2.3 ([9]). *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 $h \leftarrow \mathcal{H}$ and for any $m \leq k - 2\log(1/\epsilon)$, it holds that*

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

2.2 Hash Proof Systems

We present the framework of hash proof systems, introduced by Cramer and Shoup [7]. For simplicity we frame the description by viewing hash proof systems as key-encapsulation mechanisms (using the notation of Kiltz et al. [22]), and refer the reader to [7] 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 $C \in \mathcal{C} \setminus \mathcal{V}$,

$$\mathrm{SD}\left((pk, \Lambda_{sk}(C)), (pk, K)\right) \leq \epsilon \tag{1}$$

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 polynomial-time algorithms. The 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 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 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

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

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

3 Defining Key-Leakage Attacks

In this section we define the notion of a key-leakage attack, as introduced as Akavia et al. [2]. Due to space limitations we refer the reader to the longer version of our paper [28] for the extension to chosen-ciphertext attacks, the definition of a weak key-leakage attack, and a discussion on several other generalizations of this framework: noisy leakage, leakage of intermediate values from the key-generation process, keys that are generated using weak random sources, and leakage of intermediate values from the decryption process.

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

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.

The construction. Let $\mathsf{HPS} = (\mathsf{Param}, \mathsf{Pub}, \mathsf{Priv})$ be an ϵ_1-almost 1-universal hash proof system (see Section 2.2 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. 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 (1) in Section 2.2): 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.*

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 a well-known DDH-based 1-universal hash proof system [7], and as an immediate consequence we obtain the following corollary of Theorem 4.1:

Corollary 4.2. *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

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

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 $d \geq 1$) and satisfies this weaker 1-universality property[4]. The hash proof system is a generalization of the hash proof system underlying the simple instantiation described in Section 4. The resulting encryption scheme is then obtained by instantiating our generic construction with this hash proof system.

Our second proposal is a recent encryption scheme of Boneh et al. [5], 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].

We then provide a comparison between these two proposed schemes, indicating that the two schemes complement each other in terms of efficiency.

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 $v = (g_1, \ldots, g_k) \in \mathbb{G}^k$ and $u = (u_1, \ldots, u_k) \in \mathbb{Z}_q^k$ 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

$$(R\Phi, s, \mathsf{Ext}\,(Ry, s) \oplus M) \quad .$$

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

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

– **Decryption:** On input a ciphertext (Ψ, s, e) output $e \oplus \mathsf{Ext}(\Psi x, s)$.

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.*

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 a $((k-d)\log q, \epsilon)$-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 $\mathsf{Priv}(sk, C) = \Lambda_{sk}(C) = \mathsf{Ext}(Cx, s)$.

5.2 Proposal 2: The BHHO Scheme

We show that a simple setting of the parameters in the BHHO encryption scheme [5] 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):

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

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.

5.3 Comparison

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 the subset-product hash functions. Although the second proposal uses a more direct method to extract the randomness of the secret key, it requires the subset-product functions to operate on the *individual bits* of the secret key, since otherwise the subset-product functions are not pairwise independent. In contrast, in the first proposal the extractor operates on the secret key as *group elements*. This leads to significant differences in performance.

For any leakage parameter λ, the size of the secret keys in the two proposals is essentially the same, whereas the size of the public key in the first proposal is shorter by a factor of $\log q$. When considering the length of the ciphertexts and the number of exponentiations per ciphertext, the first proposal performs better than the second proposal when roughly $\lambda < L(1-1/\log q)$, where L is the length of the secret key (note that such a λ is a considerable amount of leakage). For example, by setting $k = 2$ in the first proposal one obtains the simple instantiation described in Section 4 which is resilient to any leakage of $L(1/2 - o(1))$ bits, and requires only 3 exponentiations per ciphertext. For achieving the same resilience in the second proposal more than $\log q$ exponentiations are required. In Table 1 we present a comparison between the efficiency of the schemes. Since the second proposal scheme is based on DDH and encrypts group elements, we compare it to the first proposal using $d = 1$ (i.e., based on DDH), and $m = \log q$ (i.e., $\log q$-bit plaintexts). For simplicity we assume that λ is a multiple of $\log q$, and note that the table presents asymptotical estimates, and not exact numbers.

Table 1. Comparison between the two proposals

	Proposal 1 (Section 5.1)	Proposal 2 (Section 5.2)
Secret key (bits)	$\lambda + 2\log q + 2\log(1/\epsilon)$	$\lambda + 2\log q + 2\log(1/\epsilon)$
Public key (bits)	$\lambda + 2\log q + 2\log(1/\epsilon)$	$\log q\,(\lambda + 2\log q + 2\log(1/\epsilon))$
Ciphertext (bits)	$\frac{(\lambda+2\log q+2\log(1/\epsilon))^2}{\log q}$	$\log q\,(\lambda + 2\log q + 2\log(1/\epsilon))$
Exponentiations	$\left(\frac{\lambda+2\log q+2\log(1/\epsilon)}{\log q}\right)^2$	$\lambda + 2\log q + 2\log(1/\epsilon)$

6 Generalized Forms of Key-Leakage Attacks

In this section we present several generalizations of the framework considered in this paper that are satisfied by our schemes. Due to space limitations we describe these generalizations very briefly and refer the reader to [28] for more details.

Noisy leakage. In the side-channel attack of Halderman et al. [18] 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. In the information-theoretic setting, this generalization does not necessarily strengthen the definition, since the leakage 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.

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. [18] 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.

Keys generated using weak random sources. When considering leakage of the random bits that are used by the key generation algorithm, then from the adversary's point of view these bits are uniformly distributed subject to the leakage information. A natural generalization 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 [13,17,36].

Leakage of intermediate values from the decryption process. An additional generalization 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 bounded length. However, it is not always guaranteed when the adversary obtains all of the memory in a noisy fashion.

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.

References

1. Abe, M., Gennaro, R., Kurosawa, K., Shoup, V.: Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 128–146. Springer, Heidelberg (2005)
2. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous hardcore bits and cryptography against memory attacks. In: TCC, pp. 474–495 (2009)
3. Biham, E., Shamir, A.: Differential fault analysis of secret key cryptosystems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)
4. Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of checking cryptographic protocols for faults. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 37–51. Springer, Heidelberg (1997)
5. Boneh, D., Halevi, S., Hamburg, M., Ostrovsky, R.: Circular-secure encryption from decision diffie-hellman. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 108–125. Springer, Heidelberg (2008)
6. Canetti, R., Dodis, Y., Halevi, S., Kushilevitz, E., Sahai, A.: Exposure-resilient functions and all-or-nothing transforms. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 453–469. Springer, Heidelberg (2000)
7. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002)
8. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J. Comput. 33(1), 167–226 (2003)
9. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. SIAM J. Comput. 38(1), 97–139 (2008)
10. Dodis, Y., Sahai, A., Smith, A.: On perfect and adaptive security in exposure-resilient cryptography. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 301–324. Springer, Heidelberg (2001)
11. Dodis, Y., Tauman Kalai, Y., Lovett, S.: On cryptography with auxiliary input. To appear in STOC (2009)
12. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography. SIAM J. Comput. 30(2), 391–437 (2000)
13. Dorrendorf, L., Gutterman, Z., Pinkas, B.: Cryptanalysis of the windows random number generator. In: ACM CCS, pp. 476–485 (2007)
14. Dziembowski, S., Pietrzak, K.: Leakage-resilient cryptography. In: FOCS, pp. 293–302 (2008)

15. Goldwasser, S., Micali, S.: Probabilistic encryption. J. Comput. Syst. Sci. 28(2), 270–299 (1984)
16. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: One-time programs. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 39–56. Springer, Heidelberg (2008)
17. Gutterman, Z., Pinkas, B., Reinman, T.: Analysis of the linux random number generator. In: IEEE Symposium on Security and Privacy, pp. 371–385 (2006)
18. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: Cold boot attacks on encryption keys. In: USENIX, pp. 45–60 (2008)
19. Heninger, N., Shacham, H.: Improved RSA private key reconstruction for cold boot attacks. Cryptology ePrint Archive, Report 2008/510 (2008)
20. Ishai, Y., Prabhakaran, M., Sahai, A., Wagner, D.: Private circuits II: Keeping secrets in tamperable circuits. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 308–327. Springer, Heidelberg (2006)
21. Ishai, Y., Sahai, A., Wagner, D.: Private circuits: Securing hardware against probing attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer, Heidelberg (2003)
22. Kiltz, E., Pietrzak, K., Stam, M., Yung, M.: A new randomness extraction paradigm for hybrid encryption. In: EUROCRYPT, pp. 590–609 (2009)
23. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
24. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
25. Kurosawa, K., Desmedt, Y.: A new paradigm of hybrid encryption scheme. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 426–442. Springer, Heidelberg (2004)
26. Maurer, U.M.: On the oracle complexity of factoring integers. Computational Complexity 5(3-4), 237–247 (1995)
27. Micali, S., Reyzin, L.: Physically observable cryptography. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 278–296. Springer, Heidelberg (2004)
28. Naor, M., Segev, G.: Public-key cryptosystems resilient to key leakage. Cryptology ePrint Archive, Report 2009/105 (2009)
29. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC, pp. 427–437 (1990)
30. Petit, C., Standaert, F.-X., Pereira, O., Malkin, T., Yung, M.: A block cipher based pseudo random number generator secure against side-channel key recovery. In: ASIACCS, pp. 56–65 (2008)
31. Pietrzak, K.: A leakage-resilient mode of operation. In: EUROCRYPT, pp. 462–482 (2009)
32. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC, pp. 84–93 (2005)
33. Rivest, R.L., Shamir, A.: Efficient factoring based on partial information. In: Pichler, F. (ed.) EUROCRYPT 1985. LNCS, vol. 219, pp. 31–34. Springer, Heidelberg (1986)
34. Shacham, H.: A Cramer-Shoup encryption scheme from the Linear assumption and from progressively weaker Linear variants. Cryptology ePrint Archive, Report 2007/074 (2007)
35. Tauman Kalai, Y., Vaikuntanathan, V.: Public-key encryption schemes with auxiliary inputs and applications (2009)
36. Yilek, S., Rescorla, E., Shacham, H., Enright, B., Savage, S.: PRNG PR0N: Understanding the Debian OpenSSL debacle (2008)