Simultaneous Hardcore Bits and Cryptography Against Memory Attacks

Adi Akavia[1⋆], Shafi Goldwasser[2⋆⋆], and Vinod Vaikuntanathan[3⋆⋆⋆]

[1] IAS and DIMACS
[2] MIT and Weizmann Insitute
[3] MIT and IBM Research

Abstract. This paper considers two questions in cryptography.

Cryptography Secure Against Memory Attacks. A particularly devastating side-channel attack against cryptosystems, termed the "memory attack", was proposed recently. In this attack, a significant fraction of the bits of a secret key of a cryptographic algorithm can be measured by an adversary if the secret key is ever *stored* in a part of memory which can be accessed even after power has been turned off for a short amount of time. Such an attack has been shown to completely compromise the security of various cryptosystems in use, including the RSA cryptosystem and AES.
We show that the public-key encryption scheme of Regev (STOC 2005), and the identity-based encryption scheme of Gentry, Peikert and Vaikuntanathan (STOC 2008) are remarkably robust against memory attacks where the adversary can measure a large fraction of the bits of the secret-key, or more generally, can compute an arbitrary function of the secret-key of bounded output length. This is done without increasing the size of the secret-key, and without introducing any complication of the natural encryption and decryption routines.

Simultaneous Hardcore Bits. We say that a block of bits of x are *simultaneously hard-core* for a one-way function $f(x)$, if given $f(x)$ they cannot be distinguished from a random string of the same length. Although any candidate one-way function can be shown to hide one hardcore bit and even a logarithmic number of simultaneously hardcore bits, there are few examples of one-way or trapdoor functions for which a linear number of the input bits have been proved simultaneously hardcore; the ones that are known relate the simultaneous security to the difficulty of factoring integers.
We show that for a lattice-based (injective) trapdoor function which is a variant of function proposed earlier by Gentry, Peikert and Vaikuntanathan, an $N - o(N)$ number of input bits are simultaneously hardcore, where N is the total length of the input.
These two results rely on similar proof techniques.

⋆ Supported in part by NSF grant CCF-0514167, by NSF grant CCF-0832797, and by Israel Science Foundation 700/08.
⋆⋆ Supported in part by NSF grants CCF-0514167, CCF-0635297, NSF-0729011, the Israel Science Foundation 700/08 and the Chais Family Fellows Program.
⋆⋆⋆ Supported in part by NSF grant CCF-0635297 and Israel Science Foundation 700/08.

1 Introduction

The contribution of this paper is two-fold.

First, we define a new class of strong side-channel attacks that we call "memory attacks", generalizing the "cold-boot attack" recently introduced by Halderman et al. [22]. We show that the public-key encryption scheme proposed by Regev [39], and the identity-based encryption scheme proposed by Gentry, Peikert, and Vaikuntanathan [16] can provably withstand these side channel attacks under essentially the same intractability assumptions as the original systems[4].

Second, we study how many bits are simultaneously hardcore for the candidate trapdoor one-way function proposed by [16]. This function family has been proven one-way under the assumption that the learning with error problem (LWE) for certain parameter settings is intractable, or alternatively the assumption that approximating the length of the shortest vector in an integer lattice to within a polynomial factor is hard for quantum algorithms [39]. We first show that for the set of parameters considered by [16], the function family has $O(\frac{N}{\log N})$ simultaneously hardcore bits (where N is the length of the input to the function). Next, we introduce a new parameter regime for which we prove that the function family is still trapdoor one-way and has upto $N - o(N)$ simultaneously hardcore bits[5], under the assumption that approximating the length of the shortest vector in an integer lattice to within a *quasi-polynomial* factor is hard for quantum algorithms running in quasi-polynomial time.

The techniques used to solve both problems are closely related. We elaborate on the two results below.

1.1 Security against Memory Attacks

The absolute privacy of the secret-keys associated with cryptographic algorithms has been the corner-stone of modern cryptography. Still, in practice, keys do get compromised at times for a variety of reasons.

A particularly disturbing loss of secrecy is as a result of side-channel attacks. These attacks exploit the fact that every cryptographic algorithm is ultimately implemented on a physical device and such implementations typically enable 'observations' which can be made and measured, such as the amount of power consumption or the time taken by a particular implementation of a cryptographic algorithm. These side-channel observations lead to information leakage about secret-keys which can (and have) lead to complete breaks of systems which have been proved mathematically secure, without violating any of the underlying mathematical principles or assumptions (see, for example, [28, 29, 12, 1, 2]). Traditionally, such attacks have been followed by ad-hoc 'fixes' which make particular implementations invulnerable to particular attacks, only to potentially be broken anew by new examples of side-channel attacks.

[4] Technically, the assumptions are the same except that they are required to hold for problems of a smaller size, or dimension. See Informal Theorems 1 and 2 for the exact statements.

[5] The statement holds for a particular $o(N)$ function. See Informal Theorem 3.

In their pioneering paper on *physically observable cryptography* [33], Micali and Reyzin set forth the goal of building a general theory of physical security against a large class of side channel attacks which one may call *computational* side-channel attacks. These include *any* side channel attack in which leakage of information on secrets occurs as a result of performing a *computation* on secrets. Some well-known examples of such attacks include Kocher's timing attacks [28], power analysis attacks [29], and electromagnetic radiation attacks [1] (see [32] for a glossary of examples.) A basic defining feature of a computational side-channel attack, as put forth by [33] is that *computation and only computation leaks information*. Namely, the portions of memory which are not involved in computation do not leak any information.

Recently, several works [33, 26, 37, 20, 15] have proposed cryptographic algorithms provably robust against computational side-channel attacks, by limiting in various ways the portions of the secret key which are involved in each step of the computation [26, 37, 20, 15].

In this paper, we consider an entirely different family of side-channel attacks that are not included in the computational side-channel attack family, as they violate the basic premise (or axiom, as they refer to it) of Micali-Reyzin [33] that *only computation* leaks information. The new class of attacks, which we call "memory attacks", are inspired by (although not restricted to) the "cold-boot attack" introduced recently by Halderman et al. [22]. The Halderman et al. paper shows how to measure a significant fraction of the bits of secret keys if the keys were *ever stored* in a part of memory which could be accessed by an adversary (e.g. DRAM), even after the power of the machine has been turned off. They show that uncovering half of the bits of the secret key that is stored in the natural way completely compromises the security of cryptosystems, such as the RSA and Rabin cryptosystems.[6]

A New Family of Side Channel Attacks Generalizing from [22], we define the family of memory attacks to leak a bounded number of bits computed as a result of applying *an arbitrary function* whose output length is bounded by $\alpha(N)$ to the content of the secret-key of the cryptographic algorithm (where N is the size of the the secret-key).[7] Naturally, this family of attacks is inherently parameterized and quantitative in nature. If $\alpha(N) = N$, then the attack could uncover the entire secret key at the outset, and there is no hope for any cryptography. However, it seems that in practice, only a fraction of the secret key is recovered [22]. The question that emerges is how large a fraction of the secret-key can leak without compromising the security of the cryptosystems.

For the public-key case (which is the focus of this paper), we differentiate between two flavors of memory attacks.

The first is *non-adaptive α-memory attacks*. Intuitively, in this case, a function h with output-length $\alpha(N)$ (where N is the length of the secret-key in the system) is first chosen by the adversary, and then the adversary is given $(PK, h(SK))$, where (PK, SK) is a random key-pair produced by the key-generation algorithm. Thus, h is

[6] This follows from the work of Rivest and Shamir, and later Coppersmith [40, 13], and has been demonstrated in practice by [22]: their experiments successfuly recovered RSA and AES keys.

[7] The special case considered in [22] corresponds to a function that outputs a subset of its input bits.

chosen independently of the system parameters and in particular, PK. This definition captures the attack specified in [22] where the bits measured were only a function of the hardware or the storage medium used. In principle, in this case, one could design the decryption algorithm to protect against the particular h which was fixed a-priori. However, this would require the design of new software (i.e, the decryption algorithm) for every possible piece of hardware (e.g, a smart-card implementing the decryption algorithm) which is highly impractical. Moreover, it seems that such a solution will involve artificially expanding the secret-key, which one may wish to avoid. We avoid the aforementioned disadvantages by showing an encryption scheme that protects against all leakage functions h (with output of length at most $\alpha(N)$).

The second, stronger, attack is the *adaptive α-memory attacks*. In this case, a key-pair (PK, SK) is first chosen by running the key generation algorithm with security parameter n, and then the adversary on input PK chooses functions h_i adaptively (depending on the PK and the outputs of $h_j(SK)$, for $j < i$) and the adversary receives $h_i(SK)$. The total number of bits output by $h_i(SK)$ for all i, is bounded by $\alpha(N)$.

Since we deal with public-key encryption (PKE) and identity-based encryption (IBE) schemes in this paper, we tailor our definitions to the case of encryption. However, we remark that similar definitions can be made for other cryptographic tasks such as digital signatures, identification protocols, commitment schemes etc. We defer these to the full version of the paper.

New Results on PKE Security. There are two natural directions to take in desiging schemes which are secure against memory attacks. The first is to look for redundant representations of secret-keys which will enable battling memory attacks. The works of [26, 25, 10] can be construed in this light. Naturally, this entails expansion of the storage required for secret keys and data. The second approach would be to examine natural and existing cryptosystems, and see how vulnerable they are to memory attacks. We take the second approach here.

Following Regev [39], we define the learning with error problem (LWE) in dimension n, to be the task of learning a vector $\mathbf{s} \in \mathbb{Z}_q^n$ (where q is a prime), given m pairs of the form $(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle + x_i \bmod q)$ where $\mathbf{a}_i \in \mathbb{Z}_q^n$ are chosen uniformly and independently and the x_i are chosen from some "error distribution" $\overline{\Psi}_\beta$ (Throughout, we one may think of x_i's as being small in magnitude. See section 2 for precise definition of this error distribution.). We denote the above parameterization by $\mathsf{LWE}_{n,m,q,\beta}$. The hardness of the LWE problem is chiefly parametrized by the dimension n: we say that $\mathsf{LWE}_{n,m,q,\beta}$ is t-hard if no probabilistic algorithm running in time t can solve it.

We prove the following two main theorems.

Informal Theorem 1 *Let the parameters m, q and β be polynomial in the security parameter n. There exist public key encryption schemes with secret-key length $N = n \log q = O(n \log n)$ that are:*

1. *semantically secure against a* non-adaptive *$(N - k)$-memory attack, assuming the $\mathsf{poly}(n)$-hardness of $\mathsf{LWE}_{O(k/\log n),m,q,\beta}$, for any $k > 0$. The encryption scheme corresponds to a slight variant of the public key encryption scheme of [39].*

2. *semantically secure against an* adaptive $O(N/\mathsf{polylog}(N))$*-memory attack, assuming the* $\mathsf{poly}(n)$*-hardness of* $\mathsf{LWE}_{k,m,q,\beta}$ *for* $k = O(n)$*. The encryption scheme is the public-key scheme proposed by [39].*

Informal Theorem 2 *Let the parameters m, q and β be polynomial in the security parameter n. The GPV identity-based encryption scheme [16] with secret-key length $N = n \log q = O(n \log n)$ is:*

1. *semantically secure against a* non-adaptive $(N-k)$*-memory attack, assuming the* $\mathsf{poly}(n)$*-hardness of* $\mathsf{LWE}_{O(k/\log n),m,q,\beta}$ *for any* $k > 0$*.*
2. *semantically secure against an* adaptive $O(N/\mathsf{polylog}(N))$*-memory attack, assuming the* $\mathsf{poly}(n)$*-hardness of* $\mathsf{LWE}_{k,m,q,\beta}$ *for* $k = O(n)$*.*

The parameter settings for these theorems require some elaboration. First, the theorem for the non-adaptive case is fully parametrized. That is, for any k, we prove security in the presence of leakage of $N - k$ bits of information about the secret-key, under a corresponding hardness assumption. The more the leakage we would like to tolerate, the stronger the hardness assumption. In particular, setting the parameter k to be $O(N)$, we prove security against leakage of a constant fraction of the secret-key bits assuming the hardness of LWE for $O(N/\log n) = O(n)$ dimensions. If we set $k = N^\epsilon$ (for some $\epsilon > 0$) we prove security against a leakage of *all but* N^ϵ *bits of the secret-key*, assuming the hardness of LWE for a polynomially smaller dimension $O(N^\epsilon/\log n) = O((n \log n)^\epsilon/\log n)$.

For the adaptive case, we prove security against a leakage of $O(N/\mathsf{polylog}(N))$ bits, assuming the hardness of LWE for $O(n)$ dimensions, where n is the security parameter of the encryption scheme.

Due to lack of space, we describe only the public-key encryption result in this paper, and defer the identity-based encryption result to the full version.

Idea of the Proof. The main idea of the proof is *dimension reduction*. To illustrate the idea, let us outline the proof of the non-adaptive case in which this idea is central.

The hardness of the encryption schemes under a non-adaptive memory attack relies on the hardness of computing $\mathbf{s}$ given $m = \mathsf{poly}(n)$ LWE samples $(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle + x_i \bmod q)$ and the leakage $h(\mathbf{s})$. Let us represent these m samples compactly as $(\mathbf{A}, \mathbf{As} + \mathbf{x})$, where the $\mathbf{a}_i$ are the rows of the matrix $\mathbf{A}$. This is exactly the LWE problem except that the adversary also gets to see $h(\mathbf{s})$. Consider now the mental experiment where $\mathbf{A} = \mathbf{BC}$, where $\mathbf{C} \in \mathbb{Z}_q^{m \times l}$ for some $l < n$. The key observations are that (a) since $h(\mathbf{s})$ is small, $\mathbf{s}$ still has considerable min-entropy given $h(\mathbf{s})$, and (b) matrix multiplication is a strong randomness extractor. In particular, these two observations together mean that $\mathbf{t} = \mathbf{Cs}$ is (statistically close to) random, even given $h(\mathbf{s})$. The resulting expression now looks like $\mathbf{Bt} + \mathbf{x}$, which is exactly the LWE distribution with secret $\mathbf{t}$ (a vector in $l < n$ dimensions). The proof of the adaptive case uses similar ideas in a more complex way: we refer the reader to Section 3.1 for the proof.

A few remarks are in order.

(Arbitrary) Polynomial number of measurements. We find it extremely interesting to construct encryption schemes secure against *repeated* memory attacks, where the combined number of bits leaked can be larger than the size of the secret-key (although any single measurement leaks only a small number of bits). Of course, if the secret-key is unchanged, this is impossible. It seems that to achieve this goal, some off-line (randomized) refreshing of the secret key must be done periodically. We do not deal with these further issues in this paper.

Leaking the content of the entire secret memory. The secret-memory may include more than the secret-keys. For example, results of intermediate computations produced during the execution of the decryption algorithm may compromise the security of the scheme even more than a carefully stored secret-key. Given this, why not allow the definition of memory attacks to measure the entire content of the secret-memory? We have two answers to this issue. First, in the case of the adaptive definition, when the decryption algorithm is deterministic (as is the case for the scheme in question and all schemes in use today), there is no loss of generality in restricting the adversary to measure the leakage from just the secret-key. This is the case because the decryption algorithm is itself only a function of the secret and public keys as well as the ciphertext that it receives, and this can be captured by a leakage function h that the adversary chooses to apply. In the non-adaptive case, the definition does not necessarily generalize this way; however, the constructions we give are secure under a stronger definition which allows leakage from the entire secret-memory. Roughly, the reason is that the decryption algorithm in question can be implemented using a small amount of extra memory, and thus the intermediate computations are an insignificant fraction of memory at any time.

1.2 Simultaneous Hard-Core Bits

The notion of hard-core bits for one-way functions was introduced very early in the developement of the theory of cryptography [42, 21, 8]. Indeed, the existence of hard-core bits for particular proposals of one-way functions (see, for example [8, 4, 23, 27]) and later for any one-way function [17], has been central to the constructions of secure public-key (and private-key) encryption schemes, and strong pseudo-random bit generators, the cornerstones of modern cryptography.

The main questions which remain open in this area concern the generalized notion of "simultaneous hard-core bit security" loosely defined as follows. Let f be a one-way function and h an easy to compute function. We say that h is a *simultaneously hard-core function* for f if given $f(x)$, $h(x)$ is computationally indistinguishable from random. In particular, we say that a block of bits of x are simultaneously hard-core for $f(x)$ if given $f(x)$, they cannot be distinguished from a random string of the same length (this corresponds to a function h that outputs a subset of its input bits).

The question of how many bits of x can be proved simultaneously hard-core has been studied for general one-way functions as well as for particular candidates in [41, 4, 31, 24, 18, 17], but the results obtained are far from satisfactory. For a general one-way function (modified in a similar manner as in their hard-core result), [17] showed the existence of an h that outputs $O(\log N)$ bits (where we let N denote the length of the input to the one-way function throughout) which is a simultaneous hard-core function

for f. For particular candidate one-way functions such as the exponentiation function (modulo a prime p), the RSA function and the Rabin function, [41, 31] have pointed to particular blocks of $O(\log N)$ input bits which are simultaneously hard-core given $f(x)$.

The first example of a one-way function candidate that hides more than $O(\log N)$ simultaneous hardcore bits was shown by Hastad, Schrift and Shamir [24, 18] who proved that the modular exponentiation function $f(x) = g^x \bmod M$ hides *half* the bits of x under the intractability of factoring the modulus M. The first example of a trapdoor function for which many bits were shown simultaneous hardcore was the Pallier function. In particular, Catalano, Gennaro and Howgrave-Graham [11] showed that $N - o(N)$ bits are simulatenously hard-core for the Pallier function, under a stronger assumption than the standard Paillier assumption.

A question raised by [11] was whether it is possible to construct other natural and efficient trapdoor functions with many simultaneous hardcore bits and in particular, functions whose conjectured one-wayness is not based on the difficulty of the factoring problem. In this paper, we present two lattice-based trapdoor functions for which is the case.

First, we consider the following trapdoor function family proposed in [16]. A function $f_{\mathbf{A}}$ in the family is described by a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, where $q = \mathsf{poly}(n)$ is prime and $m = \mathsf{poly}(n)$. $f_{\mathbf{A}}$ takes two inputs $\mathbf{s} \in \mathbb{Z}_q^n$ and a sequence of random bits $\mathbf{r}$; it first uses $\mathbf{r}$ to sample a vector $\mathbf{x}$ from (a discretized form of) the Gaussian distribution over $\mathbb{Z}_q^m$. $f_{\mathbf{A}}$ then outputs $\mathbf{As} + \mathbf{x}$. The one-wayness of this function is based on the learning with error (LWE) problem $\mathsf{LWE}_{n,m,q,\beta}$. Alternatively, the one-wayness can also be based on the worst-case quantum hardness of $\mathsf{poly}(n)$-approximate shortest vector problem ($\mathsf{gapSVP}_{\mathsf{poly}(n)}$), by a reduction of Regev [39] from gapSVP to LWE. We prove that $O(N/\log N)$ bits (where N is the total number of input bits) of $f_{\mathbf{A}}$ are simultaneously hardcore.

Second, for a new setting of the parameters in $f_{\mathbf{A}}$, we show that $N - N/\mathsf{polylog}(N)$ bits (out of the N input bits) are simultaneously hardcore. The new parameter setting is a much larger modulus $q = n^{\mathsf{polylog}(n)}$, a much smaller $m = O(n)$ and a Gaussian noise with a much smaller (inverse superpolynomial) standard deviation. At first glance, it is unclear whether for these new parameter setting, the function is still a trapdoor (injective) function. To this end, we show that the function is injective, is sampleable with an appropriate trapdoor (which can be used to invert the function) and that it is one-way. The one-wayness is based on a much stronger (yet plausible) assumption, namely the quantum hardness of gapSVP with approximation factor $n^{\mathsf{polylog}(n)}$ (For details, see Section 4.2).

We stress that our results (as well as the results of [24, 18, 11]) show that particular sets of *input bits* of these functions are simultaneously hardcore (as opposed to arbitrary hardcore functions that output many bits).

Informal Theorem 3

1. *Let m and q be polynomial in n and let $\beta = 4\sqrt{n}/q$. There exists an injective trapdoor function $\mathcal{F}_{n,m,q,\beta}$ with input length N for which a $1/\log N$ fraction of the input bits are simultaneously hardcore, assuming the $\mathsf{poly}(n)$-hardness of $\mathsf{LWE}_{O(n),m,q,\beta}$.*

2. *Let $m = O(n)$, $q = n^{\mathsf{polylog}(n)}$ and $\beta = 4\sqrt{n}/q$. There exists an injective trapdoor function $\mathcal{F}_{n,m,q,\beta}$ with input length N for which a $1 - 1/\mathsf{polylog}(N)$ fraction of input bits are simultaneously hardcore, assuming the hardness of $\mathsf{LWE}_{n/\mathsf{polylog}(n),m,q,\beta}$.*

Our proof is simple and general: one of the consequences of the proof is that a related one-way function based on the well-studied learning parity with noise problem (LPN) [7] also has $N - o(N)$ simultaneous hardcore bits. We defer the proof of this result to the full version due to lack of space.

Idea of the Proof. In the case of security against non-adaptive memory attacks, the statement we showed (see Section 1.1) is that given $\mathbf{A}$ and $h(\mathbf{s})$, $\mathbf{As} + \mathbf{x}$ looks random. The statement of hardcore bits is that given $\mathbf{A}$ and $\mathbf{As}+\mathbf{x}$, $h(\mathbf{s})$ (where h is the particular function that outputs a subset of bits of $\mathbf{s}$) looks random. Though the statements look different, the main idea in the proof of security against non-adaptive memory attacks, namely dimension reduction, carries over and can be used to prove the simultaneous hardcore bits result also. For details, see Section 4.

1.3 Other Related Work

Brent Waters, in a personal communication, has suggested a possible connection between the recently proposed notion of deterministic encryption [9, 6], and simultaneous hardcore bits. In particular, his observation is that deterministic encryption schemes (which are, informally speaking, trapdoor functions that are uninvertible even if the input comes from a min-entropy source) satisfying the definition of [9] imply trapdoor functions with many simultaneous hardcore bits. Together with the construction of deterministic encryption schemes from lossy trapdoor functions [36] (based on DDH and LWE), this gives us trapdoor functions based on DDH and LWE with many simultaneous hardcore bits. However, it seems that using this approach applied to the LWE instantiation, it is possible to get only $o(N)$ hardcore bits (where N is the total number of input bits); roughly speaking, the bottleneck is the "quality" of lossy trapdoor functions based on LWE. In contrast, in this work, we achieve $N - o(N)$ hardcore bits.

Recently, Peikert [34] has shown a classical reduction from a variant of the worst-case shortest vector problem (with appropriate approximation factors) to the average-case LWE problem. This, in turn, means that our results can be based on the *classical* worst-case hardness of this variant shortest-vector problem as well.

A recent observation of [38] surprisingly shows that any public-key encryption scheme is secure against an adaptive $\alpha(N)$-memory attack, *under (sub-)exponential hardness assumptions* on the security of the public-key encryption scheme. Slightly more precisely, the observation is that any semantically secure public-key encryption scheme that cannot be broken in time roughly $2^{\alpha(N)}$ is secure against an adaptive $\alpha(N)$-memory attack. In contrast, the schemes in this paper make only *polynomial hardness assumptions.* (See Section 3.1 for more details).

2 Preliminaries and Definitions

We will let bold capitals such as $\mathbf{A}$ denote matrices, and bold small letters such as $\mathbf{a}$ denote vectors. $\mathbf{x} \cdot \mathbf{y}$ denotes the inner product of $\mathbf{x}$ and $\mathbf{y}$. If $\mathbf{A}$ is an $m \times n$ matrix and

$S \subseteq [n]$ represents a subset of the columns of $\mathbf{A}$, we let $\mathbf{A}_S$ denote the restriction of $\mathbf{A}$ to the columns in S, namely the $m \times |S|$ matrix consisting of the columns with indices in S. In this case, we will write $\mathbf{A}$ as $[\mathbf{A}_S, \mathbf{A}_{\overline{S}}]$.

A problem is t-hard if no (probabilistic) algorithm running in time t can solve it. When we say that a problem is hard without further qualification, we mean that it is $\mathsf{poly}(n)$-hard, where n is the security parameter of the system (which is usually explicitly specified).

2.1 Cryptographic Assumptions

The cryptographic assumptions we make are related to the hardness of learning-type problems. In particular, we will consider the hardness of learning with error (LWE); this problem was introduced by Regev [39] where he showed a relation between the hardness of LWE and the *worst-case hardness* of certain problems on lattices (see Proposition 1).

We now define a probability distribution $A_{\mathbf{s},\chi}$ that is later used to specify this problem. For positive integers n and $q \geq 2$, a vector $\mathbf{s} \in \mathbb{Z}_q^n$ and a probability distribution χ on $\mathbb{Z}_q$, define $A_{\mathbf{s},\chi}$ to be the distribution obtained by choosing a vector $\mathbf{a}_i \in \mathbb{Z}_q^n$ uniformly at random, a noise-term $x_i \in \mathbb{Z}_q$ according to χ and outputting $(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle + x_i)$, where addition is performed in $\mathbb{Z}_q$.[8]

Learning With Error (LWE). Our notation here follows [39, 35]. The normal (or the Gaussian) distribution with mean 0 and variance σ^2 (or standard deviation σ) is the distribution on $\mathbb{R}$ with density function $\frac{1}{\sigma \cdot \sqrt{2\pi}} \exp(-x^2/2\sigma^2)$.

For $\beta \in \mathbb{R}^+$ we define Ψ_β to be the distribution on $\mathbb{T} = [0, 1)$ of a normal variable with mean 0 and standard deviation $\beta/\sqrt{2\pi}$, reduced modulo 1.[9] For any probability distribution $\phi : \mathbb{T} \to \mathbb{R}^+$ and an integer $q \in \mathbb{Z}^+$ (often implicit) we define its *discretization* $\bar{\phi} : \mathbb{Z}_q \to \mathbb{R}^+$ to be the distribution over $\mathbb{Z}_q$ of the random variable $\lfloor q \cdot X_\phi \rceil \bmod q$, where X_ϕ has distribution ϕ.[10] In our case, the distribution $\overline{\Psi}_\beta$ over $\mathbb{Z}_q$ is defined by choosing a number in $[0, 1)$ from the distribution Ψ_β, multiplying it by q, and rounding the result.

Definition 1. *Let $\mathbf{s} \in \mathbb{Z}_q^n$ be uniformly random. Let $q = q(n)$ and $m = m(n)$ be integers, and let $\chi(n)$ be the distribution $\overline{\Psi}_\beta$ with parameter $\beta = \beta(n)$. The goal of the learning with error problem in n dimensions, denoted $\mathsf{LWE}_{n,m,q,\beta}$, is to find $\mathbf{s}$ (with overwhelming probability) given access to an oracle that outputs m samples from the distribution $A_{\mathbf{s},\chi}$. The goal of the decision variant $\mathsf{LWE\text{-}Dist}_{n,m,q,\beta}$ is to distinguish (with non-negligible probability) between m samples from the distribution $A_{\mathbf{s},\chi}$ and m uniform samples over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. We say that $\mathsf{LWE}_{n,m,q,\beta}$ (resp. $\mathsf{LWE\text{-}Dist}_{n,m,q,\beta}$) is t-hard if no (probabilistic) algorithm running in time t can solve it.*

[8] Here, we think of n as the security parameter, and $q = q(n)$ and $\chi = \chi(n)$ as functions of n. We will sometimes omit the explicit dependence of q and χ on n.

[9] For $x \in \mathbb{R}$, $x \bmod 1$ is simply the fractional part of x.

[10] For a real x, $\lfloor x \rceil$ is the result of rounding x to the nearest integer.

The LWE problem was introduced by Regev [39], where he demonstrated a connection between the LWE problem for certain moduli q and error distributions χ, and worst-case lattice problems. In essence, he showed that LWE is as hard as solving several standard *worst-case* lattice problems *using a quantum algorithm*. We state a version of his result here. Informally, $\mathsf{gapSVP}_{c(n)}$ refers to the (worst-case) promise problem of distinguishing between lattices that have a vector of length at most 1 from ones that have no vector shorter than $c(n)$ (by scaling, this is equivalent to distinguishing between lattices with a vector of length at most k from ones with no vector shorter than $k \cdot c(n)$).

Proposition 1 ([39]). *Let $q = q(n)$ be a prime and $\beta = \beta(n) \in [0,1]$ be such that $\beta q > 2\sqrt{n}$. Assume that we have access to an oracle that solves $\mathsf{LWE}_{n,m,q,\beta}$. Then, there is a polynomial (in n and m) time quantum algorithm to solve $\mathsf{gapSVP}_{200n/\beta}$ for* any *n-dimensional lattice.*

We will use Proposition 1 as a guideline for which parameters are hard for LWE. In particular, the (reasonable) assumption that $\mathsf{gapSVP}_{n^{\mathsf{polylog}(n)}}$ is hard to solve in quasi-polynomial (quantum) time implies that $\mathsf{LWE}_{n,m,q,\beta}$ (as well as $\mathsf{LWE\text{-}Dist}_{n,m,q,\beta}$) where $q = n^{\mathsf{polylog}(n)}$ and $\beta = 2\sqrt{n}/q$ is hard to solve in polynomial time.

Regev [39] also showed that an algorithm that solves the decision version LWE-Dist with m samples implies an algorithm that solves the search version LWE in time $\mathsf{poly}(n, q)$.

Proposition 2. *There is a polynomial (in n and q) time reduction from the search version $\mathsf{LWE}_{n,m,q,\beta}$ to the decision version $\mathsf{LWE\text{-}Dist}_{n,m\cdot\mathsf{poly}(n,q),q,\beta}$, and vice versa (for some polynomial* poly*).*

Sampling $\overline{\Psi}_\beta$. The following proposition gives a way to sample from the distribution $\overline{\Psi}_\beta$ using few random bits. This is done by a simple rejection sampling routine (see, for example, [16]).

Proposition 3. *There is a PPT algorithm that outputs a vector $\mathbf{x}$ whose distribution is statistically close to $\overline{\Psi}_\beta^m$ (namely, m independent samples from $\overline{\Psi}_\beta$) using $O(m \cdot \log(q\beta) \cdot \log^2 n)$ uniformly random bits.*

2.2 Defining Memory Attacks

In this section, we define the semantic security of public-key encryption schemes against memory attacks. The definitions in this section can be extended to other cryptographic primitives as well; these extensions are deferred to the full version. We proceed to define semantic security against two flavors of memory attacks, (the stronger) adaptive memory attacks and (the weaker) non-adaptive memory attacks.

Semantic Security Against Adaptive Memory Attacks. In an adaptive memory attack against a public-key encryption scheme, the adversary, upon seeing the public-key PK, chooses (efficiently computable) functions h_i adaptively (depending on PK and the outputs of $h_j(SK)$ for $j < i$) and receives $h_i(SK)$. This is called the probing phase. The definition is parametrized by a function $\alpha(\cdot)$, and requires that *the total number of bits output by $h_i(SK)$ for all i is bounded by $\alpha(N)$* (where N is the length of the secret-key).

After the probing phase, the adversary plays the semantic security game, namely he chooses two messages (m_0, m_1) of the same length and gets $\text{ENC}_{PK}(m_b)$ for a random $b \in \{0,1\}$ and he tries to guess b. We require that the adversary guesses the bit b with probability at most $\frac{1}{2} + \mathsf{negl}(n)$, where n is the security parameter and negl is a negligible function. We stress that the adversary is allowed to get the measurements $h_i(SK)$ only *before* he sees the challenge ciphertext. The formal definition follows.

Definition 2 (Adaptive Memory Attacks). *Let $\alpha : \mathbb{N} \to \mathbb{N}$ be a function, and let N be the size of the secret-key output by $\text{GEN}(1^n)$. Let H_{SK} be an oracle that takes as input a polynomial-size circuit h and outputs $h(SK)$. A PPT adversary $A = (A_1^{H_{SK}}, A_2)$ is called admissible if the total number of bits that A gets as a result of oracle queries to H_{SK} is at most $\alpha(N)$.*

A public-key encryption scheme $\text{PKE} = (\text{GEN}, \text{ENC}, \text{DEC})$ is semantically secure against adaptive $\alpha(N)$-memory attacks *if for any* admissible *PPT adversary $A = (A_1, A_2)$, the probability that A wins in the following experiment differs from $\frac{1}{2}$ by a negligible function in n.*

$(\text{PK}, \text{SK}) \leftarrow \text{GEN}(1^n)$
$(m_0, m_1, \mathsf{state}) \leftarrow A_1^{H_{SK}}(\text{PK})$ *s.t.* $|m_0| = |m_1|$
$y \leftarrow \text{ENC}_{PK}(m_b)$ *where* $b \in \{0,1\}$ *is a random bit*
$b' \leftarrow A_2(y, \mathsf{state})$

The adversary A wins the experiment if $b' = b$.

The definitions of security for identity-based encryption schemes against memory attacks is similar in spirit, and is deferred to the full version.

Semantic Security Against Non-Adaptive Memory Attacks. Non-adaptive memory attacks capture the scenario in which a polynomial-time computable leakage function h whose output length is bounded by $\alpha(N)$ is fixed in advance (possibly as a function of the encryption scheme, and the underlying hardware). We require that the encryption scheme be semantically secure even if the adversary is given the auxiliary input $h(SK)$. We stress that h is chosen *independently of the public-key PK*. Even though this is much weaker than the adaptive definition, schemes satisfying the non-adaptive definition could be much easier to design and prove (as we will see in Section 3). Moreover, in some practical scenarios, the leakage function is just a characteristic of the hardware and is independent of the parameters of the system, including the public-key. The formal definition follows.

Definition 3 (Non-adaptive Memory Attacks). *Let $\alpha : \mathbb{N} \to \mathbb{N}$ be a function, and let N be the size of the secret-key output by $\text{GEN}(1^n)$. A public-key encryption scheme $\text{PKE} = (\text{GEN}, \text{ENC}, \text{DEC})$ is semantically secure against* non-adaptive $\alpha(N)$-memory attacks *if for any function $h : \{0,1\}^N \to \{0,1\}^{\alpha(N)}$, and any PPT adversary $A = (A_1, A_2)$, the probability that A wins in the following experiment differs from $\frac{1}{2}$ by a negligible function in n:*

$(\text{PK}, \text{SK}) \leftarrow \text{GEN}(1^n)$
$(m_0, m_1, \mathsf{state}) \leftarrow A_1(\text{PK}, h(\text{SK}))$ *s.t.* $|m_0| = |m_1|$
$y \leftarrow \text{ENC}_{PK}(m_b)$ *where* $b \in \{0,1\}$ *is a random bit*
$b' \leftarrow A_2(y, \mathsf{state})$

The adversary A wins the experiment if $b' = b$.

Remarks about the Definitions

A Simpler Definition that is Equivalent to the adaptive definition. We observe that without loss of generality, we can restrict our attention to an adversary that outputs a single function h (whose output length is bounded by $\alpha(N)$) and gets $(PK, h(PK, SK))$ (where $(PK, SK) \leftarrow \text{GEN}(1^n)$) as a result. Informally, the equivalence holds because the adversary can encode all the functions h_i (that depend on PK as well as $h_j(SK)$ for $j < i$) into a *single polynomial-size circuit* h that takes PK as well as SK as inputs. We will use this formulation of Definition 2 later in the paper.

The Dependence of the Leakage Function on the Challenge Ciphertext. In the adaptive definition, the adversary is not allowed to obtain $h(SK)$ *after* he sees the challenge ciphertext. This restriction is necessary: if we allow the adversary to choose h depending on the challenge ciphertext, he can use this ability to decrypt it (by letting h be the decryption circuit and encoding the ciphertext into h), and thus the definition would be unachievable.

A similar issue arises in the definition of CCA2-security of encryption schemes, where the adversary should be prohibited from querying the decryption oracle on the challenge ciphertext. Unfortunately, whereas the solution to this issue in the CCA2-secure encryption case is straightforward (namely, explicity disallow querying the decryption oracle on the challenge ciphertext), it seems far less clear in our case.

The Adaptive Definition and Bounded CCA1-security. It is easy to see that a bit-encryption scheme secure against an adaptive $\alpha(N)$-memory attack is also secure against a CCA1 attack where adversary can make at most $\alpha(N)$ decryption queries (also called an $\alpha(N)$-bounded CCA1 attack).

3 Public-key Encryption Secure Against Memory Attacks

In this section, we construct a public-key encryption scheme that is secure against memory attacks. In Section 3.1, we show that the Regev encryption scheme [39] is secure against *adaptive α-memory attacks*, for $\alpha(N) = O(\frac{N}{\log N})$, under the assumption that $\mathsf{LWE}_{O(n),m,q,\beta}$ is $\mathsf{poly}(n)$-hard (where n is the security parameter and $N = 3n \log q$ is the length of the secret-key). The parameters q, m and β are just as in Regev's encryption scheme, described below.

In Section 3.2, we show that a slight variant of Regev's encryption scheme is secure against *non-adaptive* $(N - k)$-memory attacks, assuming the $\mathsf{poly}(\mathsf{n})$-hardness of $\mathsf{LWE}_{O(k/\log n),m,q,\beta}$. On the one hand, this allows the adversary to obtain more information about the secret-key but on the other hand, achieves a much weaker (namely, non-adaptive) definition of security.

The Regev Encryption Scheme. First, we describe the public-key encryption scheme of Regev, namely $\text{RPKE} = (\text{RGEN}, \text{RENC}, \text{RDEC})$ which works as follows. Let n be the security parameter and let $m(n), q(n), \beta(n) \in \mathbb{N}$ be parameters of the system. For concreteness, we will set $q(n)$ be a prime between n^3 and $2n^3$, $m(n) = 3n \log q$ and $\beta(n) = 4\sqrt{n}/q$.

- $\text{RGEN}(1^n)$ picks a random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, a random vector $\mathbf{s} \in \mathbb{Z}_q^n$ and a vector $\mathbf{x} \leftarrow \overline{\Psi}_\beta^m$ (that is, where each entry x_i is chosen independently from the probability distribution $\overline{\Psi}_\beta$). Output $\text{PK} = (\mathbf{A}, \mathbf{As} + \mathbf{x})$ and $\text{SK} = \mathbf{s}$.
- $\text{RENC}(\text{PK}, b)$, where b is a bit, works as follows. First, pick a vector $\mathbf{r}$ at random from $\{0,1\}^m$. Output $(\mathbf{rA}, \mathbf{r}(\mathbf{As} + \mathbf{x}) + b\lfloor \frac{q}{2} \rceil)$ as the ciphertext.
- $\text{RDEC}(\text{SK}, c)$ first parses $c = (\mathbf{c}_0, c_1)$, computes $b' = c_1 - \mathbf{c}_0 \cdot \mathbf{s}$ and outputs 0 if b' is closer to 0 than to $\frac{q}{2}$, and 1 otherwise.

Decryption is correct because the value $b' = \mathbf{r} \cdot \mathbf{x} + b\lfloor q/2 \rfloor$ computed by the decryption algorithm is very close to $b\lfloor q/2 \rfloor$: this is because the absolute value of $\mathbf{r} \cdot \mathbf{x}$ is much smaller than $q/4$. In particular, since $||\mathbf{r}||_2 \leq \sqrt{m}$ and $||\mathbf{x}||_2 \leq mq\beta = 4m\sqrt{n}$ with high probability, $|\mathbf{r} \cdot \mathbf{x}| \leq ||\mathbf{r}||_2 ||\mathbf{x}||_2 \leq 4m\sqrt{mn} \ll q/4$.

3.1 Security Against Adaptive Memory Attacks

Let $N = 3n \log q$ be the length of the secret-key in the Regev encryption scheme. In this section, we show that the scheme is secure against $\alpha(N)$-adaptive memory attacks for any $\alpha(N) = O(\frac{N}{\log N})$, assuming that $\mathsf{LWE}_{O(n),m,q,\beta}$ is $\mathsf{poly}(n)$-hard, where m, q and β are as in encryption scheme described above.

Theorem 1. *Let the parameters m, q and β be as in* RPKE. *Assuming that $\mathsf{LWE}_{O(n),m,q,\beta}$ is $\mathsf{poly}(n)$-hard, the scheme is semantically secure against adaptive $\alpha(N)$-memory attacks for $\alpha(N) \leq N/10 \log N$.*

Proof. (Sketch.) First, we observe that without loss of generality, we can restrict our attention to an adversary that outputs single function h (whose output length is bounded by $\alpha(N)$) and the adversary gets $(PK, h(PK, SK))$ as a result. Informally, the equivalence holds because the adversary can encode all the functions h_i (that depend on PK as well as $h_j(SK)$ for $j < i$) into a *single polynomial (in n) size circuit* h that takes PK as well as SK as inputs.

Thus, it suffices to show that for any polynomial-size circuit h,

$$(PK, \text{ENC}_{PK}(0), h(PK, SK)) \approx_c (PK, \text{ENC}_{PK}(1), h(PK, SK))$$

In our case, it suffices to show the following statement (which states that the encryption of 0 is computationally indistinguishable from uniform)

$$(\mathbf{A}, \mathbf{As} + \mathbf{x}, \mathbf{rA}, \mathbf{r}(\mathbf{As} + \mathbf{x}), h(\mathbf{A}, \mathbf{s}, \mathbf{x})) \approx_c (\mathbf{A}, \mathbf{As} + \mathbf{x}, \mathbf{u}, u', h(\mathbf{A}, \mathbf{s}, \mathbf{x})) \quad (1)$$

where $\mathbf{u} \in \mathbb{Z}_q^n$ and $u' \in \mathbb{Z}_q$ are uniformly random and independent of all other components. That is, the ciphertext is computationally indistinguishable from uniformly random, given the public-key and the leakage $h(PK, SK)$.

We will in fact show a stronger statement, namely that

$$(\mathbf{A}, \mathbf{As}+\mathbf{x}, \mathbf{rA}, \mathbf{rAs}, h(\mathbf{A},\mathbf{s},\mathbf{x}), \mathbf{rx}) \approx_c (\mathbf{A}, \mathbf{As}+\mathbf{x}, \mathbf{u}, u', h(\mathbf{A},\mathbf{s},\mathbf{x}), \mathbf{rx}) \quad (2)$$

The difference between (1) and (2) is that in the latter, the distributions also contain the additional information $\mathbf{r}\cdot\mathbf{x}$. Clearly, this is stronger than (1). We show (2) in four steps.

Step 1. We show that $\mathbf{rA}$ can be replaced with a uniformly random vector in $\mathbb{Z}_q^n$ while maintaining statistical indistinguishability, even given $\mathbf{A}, \mathbf{As}+\mathbf{x}$, the leakage $h(\mathbf{A},\mathbf{s},\mathbf{x})$ and $\mathbf{r}\cdot\mathbf{x}$. More precisely,

$$(\mathbf{A}, \mathbf{As}+\mathbf{x}, \mathbf{rA}, \mathbf{rAs}, h(\mathbf{A},\mathbf{s},\mathbf{x}), \mathbf{r}\cdot\mathbf{x}) \approx_s (\mathbf{A}, \mathbf{As}+\mathbf{x}, \mathbf{u}, \mathbf{u}\cdot\mathbf{s}, h(\mathbf{A},\mathbf{s},\mathbf{x}), \mathbf{r}\cdot\mathbf{x}) \quad (3)$$

where $\mathbf{u} \in \mathbb{Z}_q^n$ is uniformly random.

Informally, 3 is true because of the leftover hash lemma. (A variant of) leftover hash lemma states that if (a) $\mathbf{r}$ is chosen from a distribution over $\mathbb{Z}_q^n$ with min-entropy $k \geq 2n\log q + \omega(\log n)$, (b) $\mathbf{A}$ is a uniformly random matrix in $\mathbb{Z}_q^{m\times n}$, and (c) the distributions of $\mathbf{r}$ and $\mathbf{A}$ are statistically independent, then $(\mathbf{A}, \mathbf{rA}) \approx_s (\mathbf{A}, \mathbf{u})$ where $\mathbf{u}$ is a uniformly random vector in $\mathbb{Z}_q^n$. Given $\mathbf{r}\cdot\mathbf{x}$ (which has length $\log q = O(\log n)$), the residual min-entropy of $\mathbf{r}$ is at least $m - \log q \geq 2n\log q + \omega(\log n)$. Moreover, the distribution of $\mathbf{r}$ given $\mathbf{r}\cdot\mathbf{x}$ depends only on $\mathbf{x}$, and is statistically independent of $\mathbf{A}$. Thus, leftover hash lemma applies and $\mathbf{rA}$ can be replaced with a random vector $\mathbf{u}$.

Step 2. This is the crucial step in the proof. Here, we replace the (uniformly random) matrix $\mathbf{A}$ with a matrix $\mathbf{A}'$ drawn from another distribution $\mathcal{D}$. Informally, the (efficiently sampleable) distribution $\mathcal{D}$ satisfies two properties: (1) a random matrix drawn from $\mathcal{D}$ is computationally indistinguishable from a uniformly random matrix, assuming the $\mathsf{poly}(n)$-hardness of $\mathsf{LWE}_{O(n),m,q,\beta}$, and (2) given $\mathbf{A}' \leftarrow \mathcal{D}$ and $\mathbf{y} = \mathbf{A}'\mathbf{s}+\mathbf{x}$, the min-entropy of $\mathbf{s}$ is at least n. The existence of such a distribution follows from Lemma 1 below.

The intuition behind this step is the following: Clearly, $\mathbf{As}+\mathbf{x}$ is computationally indistinguishable from $\mathbf{A}'\mathbf{s}+\mathbf{x}$. Moreover, given $\mathbf{A}'\mathbf{s}+\mathbf{x}$, $\mathbf{s}$ has high (information-theoretic) min-entropy. Thus, in some informal sense, $\mathbf{s}$ has high "computational entropy" given $\mathbf{As}+\mathbf{x}$. This is the intuition for the next step.

Summing up, the claim in this step is that

$$(\mathbf{A}, \mathbf{As}+\mathbf{x}, \mathbf{u}, \mathbf{u}\cdot\mathbf{s}, h(\mathbf{A},\mathbf{s},\mathbf{x}), \mathbf{r}\cdot\mathbf{x}) \approx_c (\mathbf{A}', \mathbf{A}'\mathbf{s}+\mathbf{x}, \mathbf{u}, \mathbf{u}\cdot\mathbf{s}, h(\mathbf{A}',\mathbf{s},\mathbf{x}), \mathbf{r}\cdot\mathbf{x}) \quad (4)$$

where $\mathbf{A}' \leftarrow \mathcal{D}$. This follows directly from Lemma 1 below.

Step 3. By Lemma 1, $\mathbf{s}$ has min-entropy at least $n \geq \frac{N}{9\log N}$ given $\mathbf{A}'\mathbf{s}+\mathbf{x}$. Since the output length of h is at most $\frac{N}{10\log N}$ and the length of $\mathbf{r}\cdot\mathbf{x}$ is $\log q = O(\log n)$, $\mathbf{s}$ still has residual min-entropy $\omega(\log n)$ given $\mathbf{A}', \mathbf{A}'\mathbf{s}+\mathbf{x}$, $h(\mathbf{A}',\mathbf{s},\mathbf{x})$ and $\mathbf{r}\cdot\mathbf{x}$. Note also that the vector $\mathbf{u}$ on the left-hand side distribution is independent of $(\mathbf{A}, \mathbf{As}+\mathbf{x}, h(\mathbf{A},\mathbf{s},\mathbf{x}), \mathbf{r}\cdot\mathbf{x})$. This allows us to apply leftover hash lemma again (with $\mathbf{u}$ as the "seed" and $\mathbf{s}$ as the min-entropy source). Thus,

$$(\mathbf{A}', \mathbf{A}'\mathbf{s}+\mathbf{x}, \mathbf{u}, \mathbf{u}\cdot\mathbf{s}, h(\mathbf{A}',\mathbf{s},\mathbf{x}), \mathbf{r}\cdot\mathbf{x}) \approx_s (\mathbf{A}', \mathbf{A}'\mathbf{s}+\mathbf{x}, \mathbf{u}, u', h(\mathbf{A}',\mathbf{s},\mathbf{x}), \mathbf{r}\cdot\mathbf{x}) \quad (5)$$

where $u' \leftarrow \mathbb{Z}_q$ is uniformly random and independent of all the other components in the distribution.

Step 4. In the last step, we switch back to a uniform matrix $\mathbf{A}$. That is,

$$(\mathbf{A}', \mathbf{A}'\mathbf{s} + \mathbf{x}, \mathbf{u}, u', h(\mathbf{A}', \mathbf{s}, \mathbf{x}), \mathbf{r} \cdot \mathbf{x}) \approx_c (\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{x}, \mathbf{u}, u', h(\mathbf{A}, \mathbf{s}, \mathbf{x}), \mathbf{r} \cdot \mathbf{x}) \quad (6)$$

Putting the four steps together proves (2). □

Lemma 1. *There is a distribution $\mathcal{D}$ such that*

- $\mathbf{A} \leftarrow U_{\mathbb{Z}_q^{m \times n}} \approx_c \mathbf{A}' \leftarrow \mathcal{D}$*, assuming the* $\mathsf{poly}(n)$*-hardness of* $\mathsf{LWE}_{O(n),m,q,\beta}$*, where m, q, β are as in Regev's encryption scheme.*
- *The min-entropy of* $\mathbf{s}$ *given* $\mathbf{A}'\mathbf{s} + \mathbf{x}$ *is at least n. That is,* $\mathsf{H}_\infty(\mathbf{s} \mid \mathbf{A}'\mathbf{s} + \mathbf{x}) \geq n$ [11].

Remark: The above lemma is a new lemma proved in [19]; it has other consequences such as security under auxiliary input, which is beyond the scope of this paper.

A Different Proof of Adaptive Security under (Sub-)Exponential Assumptions. Interestingly, [38] observed that *any* public-key encryption scheme that is $2^{\alpha(N)}$-hard can be proven to be secure against $\alpha(N)$ adaptive memory attacks. In contrast, our result (Theorem 1) holds under a standard, polynomial (in the security parameter n) hardness assumption (for a reduced dimension, namely $O(n)$). We sketch the idea of the [38] proof here.

The proof follows from the existence of a simulator that breaks the standard semantic security with probability $\frac{1}{2} + \frac{\epsilon}{2^{\alpha(N)}}$ given an adversary that breaks the adaptive $\alpha(N)$-memory security with probability $\frac{1}{2} + \epsilon$. The simulator simply guesses the (at most $\alpha(N)$) bits of the output of h and runs the adversary with the guess; if the guess is correct, the adversary succeeds in guessing the encrypted bit with probability $\frac{1}{2} + \epsilon$. The key observation that makes this idea work is that there is indeed a way for the simulator to "test" if its guess is correct or wrong: simply produce many encryptions of random bits and check if the adversary succeeds on more than $1/2 + \epsilon$ fraction of these encryptions. We remark that this proof idea carries over to the case of symmetric encryption schemes secure against a chosen plaintext attack (that is, CPA-secure) as well.

3.2 Security Against Non-Adaptive Memory Attacks

In this section, we show that a variant of Regev's encryption scheme is secure against non-adaptive $N - o(N)$ memory attacks (where N is the length of the secret-key), assuming that $\mathsf{LWE}_{o(n),m,q,\beta}$ is $\mathsf{poly}(n)$-hard. The variant encryption scheme differs from Regev's encryption scheme only in the way the public-key is generated.

The key generation algorithm picks the matrix $\mathbf{A}$ as $\mathbf{BC}$ where $\mathbf{B}$ is uniformly random in $\mathbb{Z}_q^{m \times k}$ and $\mathbf{C}$ is uniformly random in $\mathbb{Z}_q^{k \times n}$ (as opposed to uniformly random in $\mathbb{Z}_q^{n \times m}$). We will let $k = n - \frac{\alpha(N)}{3 \log q}$ (note that $k < n$). For this modified key-generation procedure, it is easy to show that the decryption algorithm is still correct. We show:

Theorem 2. *The variant public-key encryption scheme outlined above is secure against a non-adaptive α-memory attack, where $\alpha(N) \leq N - o(N)$ for some $o(N)$ function, assuming that* $\mathsf{LWE}_{o(n),m,q,\beta}$ *is* $\mathsf{poly}(n)$*-hard, where the parameters m, q and β are exactly as in Regev's encryption scheme.*

[11] The precise statement uses the notion of average min-entropy due to Dodis, Reyzin and Smith [14].

We sketch a proof of this theorem below. The proof of semantic security of Regev's encryption is based on the fact that the public-key $(\mathbf{A}, \mathbf{As} + \mathbf{x})$ is computationally indistinguishable from uniform. In order to show security against non-adaptive memory attacks, it is sufficient to show that this computational indistinguishability holds even given $h(\mathbf{s})$, where h is an arbitrary (polynomial-time computable) function whose output length is at most $\alpha(N)$.

The proof of this essentially follows from the leftover hash lemma. First of all, observe that $\mathbf{s}$ has min-entropy at least $N - \alpha(N)$, given $h(\mathbf{s})$ (this is because the output length of h is at most $\alpha(N)$). Furthermore, the distribution of $\mathbf{s}$ given $h(\mathbf{s})$ is independent of $\mathbf{A}$ (since h depends only on $\mathbf{s}$ and is chosen independent of $\mathbf{A}$). By our choice of parameters, $N - \alpha(N) \geq 3k \log q$. Thus, leftover hash lemma implies that $\mathbf{Cs}$ is a vector $\mathbf{t}$ whose distribution is statistically close to uniform (even given $\mathbf{C}$ and $h(\mathbf{s})$). Thus, $\mathbf{As} + \mathbf{x} = \mathbf{BCs} + \mathbf{x} = \mathbf{Bt} + \mathbf{x}$ is distributed exactly like the output of an LWE distribution with dimension k (since $\mathbf{t} \in \mathbb{Z}_q^k$). This is computationally indistinguishable from random, assuming $\mathsf{LWE}_{k,m,q,\beta} = \mathsf{LWE}_{o(n),m,q,\beta}$ (since $k = o(n)$ by our choice).

4 Simultaneous Hardcore Bits

In this section, we show that variants of the trapdoor one-way function proposed by Gentry et al [16] (the GPV trapdoor function) has many simultaneous hardcore bits. For the parameters of [16], we show that a $1/\mathsf{polylog}(N)$ fraction of the input bits are simultaneously hardcore, assuming the $\mathsf{poly}(n)$-hardness of $\mathsf{LWE}_{O(n),m,q,\beta}$ (here, m and q are polynomial in n and β is inverse-polynomial in n, the GPV parameter regime).

More significantly, we show a different (and non-standard) choice of parameters for which the function has $N - N/\mathsf{polylog}(N)$ hardcore bits. The choice of parameters is $m = O(n)$, a modulus $q = n^{\mathsf{polylog}(n)}$ and $\beta = 4\sqrt{n}/q$. This result assumes the $\mathsf{poly}(n)$-hardness of $\mathsf{LWE}_{n/\mathsf{polylog}(n),m,q,\beta}$ for these parameters m, q and β. The parameters are non-standard in two respects: first, the modulus is superpolynomial, and the noise rate is very small (i.e, inverse super-polynomial) which makes the hardness assumption stronger. Secondly, the number of samples m is linear in n (as opposed to roughly $n \log n$ in [16]): this affects the trapdoor properties of the function (for more details, see Section 4.2). Also, note that the hardness assumption here refers to a reduced dimension (namely, $n/\mathsf{polylog}(n)$).

We remark that for any sufficiently large $o(N)$ function, we can show that the GPV function is a trapdoor function with $N - o(N)$ hardcore bits for different choices of parameters. We defer the details to the full version.

4.1 Hardcore Bits for the GPV Trapdoor Function

In this section, we show simultaneous hardcore bits for the GPV trapdoor function. First, we show a general result about hardcore bits that applies to a wide class of parameter settings: then, we show how to apply it to get $O(N/\mathsf{polylog}(N))$ hardcore bits for the GPV parameters, and in Section 4.2, $N - N/\mathsf{polylog}(N)$ hardcore bits for our new setting of parameters.

The collection of (injective) trapdoor functions $\mathcal{F}_{n,m,q,\beta}$ is defined as follows. Let $m = m(n)$ be polynomial in n. Each function $f_{\mathbf{A}} : \mathbb{Z}_q^n \times \{0,1\}^r \rightarrow \mathbb{Z}_q^m$ is indexed

by a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$. It takes as input $(\mathbf{s}, \mathbf{r})$ where $\mathbf{s} \in \mathbb{Z}_q^n$ and $\mathbf{r} \in \{0,1\}^r$, first uses $\mathbf{r}$ to sample a vector $\mathbf{x} \leftarrow \overline{\Psi}_\beta^m$ (that is, a vector each of whose components is independently drawn from the Gaussian error-distribution $\overline{\Psi}_\beta$), and outputs $\mathbf{As} + \mathbf{x}$. Clearly, the one-wayness of this function is equivalent to solving $\mathsf{LWE}_{n,m,q,\beta}$. Gentry et al. [16] show that $\mathcal{F}_{n,m,q,\beta}$ is a *trapdoor* one-way function for the parameters $q = O(n^3)$, $m = 3n \log q$ and $\beta = 4\sqrt{n}/q$ (assuming the hardness of $\mathsf{LWE}_{n,m,q,\beta}$).

Lemma 2. *For any integer $n > 0$, integer $q \geq 2$, an error-distribution $\chi = \overline{\Psi}_\beta$ over $\mathbb{Z}_q$ and any subset $S \subseteq [n]$, the two distributions $(\mathbf{A}, \mathbf{As} + \mathbf{x}, \mathbf{s}|_S)$ and $(\mathbf{A}, \mathbf{As} + \mathbf{x}, U_{\mathbb{Z}_q^{|S|}})$ are computationally indistinguishable assuming the hardness of the decision version $\mathsf{LWE\text{-}Dist}_{n-|S|,m,q,\beta}$.*

Proof. We will show this in two steps.

Step 1. The first and the main step is to show that $(\mathbf{A}, \mathbf{As}+\mathbf{x}, \mathbf{s}|_S) \approx_c (\mathbf{A}, U_{\mathbb{Z}_q^m}, U_{\mathbb{Z}_q^{|S|}})$. The distribution on the right consists of uniformly random and independent elements. This statement is shown by contradiction: Suppose a PPT algorithm D distinguishes between the two distributions. Then, we construct a PPT algorithm E that breaks the decision version $\mathsf{LWE\text{-}Dist}_{n-|S|,m,q,\beta}$. E gets as input $(\mathbf{A}', \mathbf{y}')$ such that $\mathbf{A}' \in \mathbb{Z}_q^{m \times (n-|S|)}$ is uniformly random and $\mathbf{y}'$ is either drawn from the LWE distribution (with dimension $n - |S|$) or is uniformly random. E does the following:

1. Let $\mathbf{A}_{\bar{S}} = \mathbf{A}'$. Choose $\mathbf{A}_S$ uniformly at random from $\mathbb{Z}_q^{m \times |S|}$ and set $\mathbf{A} = [\mathbf{A}_S, \mathbf{A}_{\bar{S}}]$.
2. Choose $\mathbf{s}_S \leftarrow \mathbb{Z}_q^{|S|}$ uniformly at random and compute $\mathbf{y} = \mathbf{y}' + \mathbf{A}_S \mathbf{s}_S$.
3. Run D with input $(\mathbf{A}, \mathbf{y}, \mathbf{s}_S)$, and output whatever D outputs.

First, suppose $(\mathbf{A}', \mathbf{y}')$ is drawn from the LWE distribution $A_{\mathbf{s}',\chi}$ for some $\mathbf{s}'$. Let $\mathbf{s}_{\bar{S}} = \mathbf{s}'$ and let $\mathbf{s} = [\mathbf{s}_S, \mathbf{s}_{\bar{S}}]$. Then, $(\mathbf{A}, \mathbf{y})$ constructed by E is distributed identical to $A_{\mathbf{s},\chi}$. On the other hand, if $(\mathbf{A}', \mathbf{y}')$ is drawn from the uniform distribution, then $(\mathbf{A}, \mathbf{y})$ is uniformly distributed, and independent of $\mathbf{s}|_S$. Thus, if D distinguishes between the two distributions, then E solves $\mathsf{LWE\text{-}Dist}_{n-|S|,m,q,\beta}$.

Step 2. The second step is to show that $(\mathbf{A}, U_{\mathbb{Z}_q^m}, U_{\mathbb{Z}_q^{|S|}}) \approx_c (\mathbf{A}, \mathbf{As} + \mathbf{x}, U_{\mathbb{Z}_q^{|S|}})$. This is equivalent to the hardness of $\mathsf{LWE\text{-}Dist}_{n,m,q,\beta}$. □

The theorem below shows that for the GPV parameter settings, a $1/\mathsf{polylog}(N)$ fraction of the bits are simultaneously hardcore.

Theorem 3. *Let $\gamma = m \log(q\beta) \log^2 n / n \log q$. For any $k > 0$, assuming that $\mathsf{LWE}_{k,m,q,\beta}$ is $\mathsf{poly}(n,q)$-hard, the fraction of simultaneous hardcore bits for the family $\mathcal{F}_{n,m,q,\beta}$ is $\frac{1}{1+\gamma}(1 - \frac{k}{n})$. In particular, for the GPV parameters as above, the number of hardcore bits is $O(N/\mathsf{polylog}(N))$.*

Proof. We first bound the total input length of a function in $\mathcal{F}_{n,m,q,\beta}$, in terms of n, m, q and β. The number of bits r needed to sample $\mathbf{x}$ from $\overline{\Psi}_\beta^m$ is $mH(\beta) =$

$O(m \log(q\beta) \log^2 n)$, by Proposition 3. Thus, the total input length is $n \log q + r = n \log q + O(m \log(q\beta) \log^2 n) = n \log q(1+\gamma)$.

By Lemma 2, assuming the hardness of the decision problem $\mathsf{LWE\text{-}Dist}_{k,m,q,\beta}$ (or, by Proposition 2, assuming the $\mathsf{poly}(n,q)$-hardness of the search problem $\mathsf{LWE}_{k,m,q,\beta}$), the number of simultaneously hardcore bits is at least $(n-k)\log q$. The fraction of hardcore bits, then, is $\frac{(n-k)\log q}{n \log q(1+\gamma)} = \frac{1}{1+\gamma}(1 - \frac{k}{n})$.

For the GPV parameters $\gamma = \mathsf{polylog}(N)$, and with $k = O(n)$, the number of hardcore bits is $O(N/\mathsf{polylog}(N))$ assuming the hardness of $\mathsf{LWE}_{O(n),m,q,\beta}$. □

4.2 A New Setting of Parameters for the GPV Function

In this section, we show a choice of the parameters for the GPV function for which the function remains trapdoor one-way and an $1 - o(1)$ fraction of the input bits are simultaneously hardcore. Although the number of hardcore bits remains the same as in the GPV parametrization (as a function of n and q), namely $(n-k)\log q$ bits assuming the hardness of $\mathsf{LWE}_{k,m,q,\beta}$, the length of the input relative to this number will be much smaller. Overall, this means that the *fraction* of input bits that are simultaneously hardcore is larger.

We choose the parameters so that r (the number of random bits needed to sample the error-vector $\mathbf{x}$) is a subconstant fraction of $n \log q$. This could be done in one (or both) of the following ways. (a) Reduce m relative to n: note that m cannot be too small relative to n, otherwise the function ceases to be injective. (b) Reduce the standard deviation β of the Gaussian noise relative to the modulus q: as β/q gets smaller and smaller, it becomes easier to invert the function and consequently, the one-wayness of the function has to be based on progressively stronger assumptions. Indeed, we will employ both these methods (a) and (b) to achieve our goal.

In addition, we have to show that for our choice of parameters, it is possible to sample a random function in $\mathcal{F}_{n,m,q,\beta}$ (that is, the trapdoor sampling property) and that given the trapdoor, it is possible to invert the function (that is, the trapdoor inversion property). See the proof of Theorem 4 below for more details.

Our choice of parameters is $m(n) = 6n$, $q(n) = n^{\log^3 n}$ and $\beta = 4\sqrt{n}/q$.

Theorem 4. *Let $m(n) = 6n$, $q(n) = n^{\log^3 n}$ and $\beta = 4\sqrt{n}/q$. Then, the family of functions $\mathcal{F}_{n,m,q,\beta}$ is a family of trapdoor injective one-way functions with an $1 - 1/\mathsf{polylog}(N)$ fraction of hardcore bits, assuming the $n^{\mathsf{polylog}(n)}$-hardness of the search problem $\mathsf{LWE}_{n/\mathsf{polylog}(n),m,q,\beta}$. Using Regev's worst-case to average-case connection for LWE, the one-wayness of this function family can also be based on the worst-case $n^{\mathsf{polylog}(n)}$-hardness of $\mathsf{gapSVP}_{n^{\mathsf{polylog}(n)}}$.*

Proof. (Sketch.) Let us first compute the fraction of hardcore bits. By Theorem 3 applied to our parameters, we get a $1 - \frac{1}{\log n}$ fraction of hardcore bits assuming the hardness of $\mathsf{LWE\text{-}Dist}_{O(n/\log n),m,q,\beta}$. By Propositions 2 and 1, this translates to the assumptions claimed in the theorem.

We now outline the proof that for this choice of parameters, $\mathcal{F}_{n,m,q,\beta}$ is an injective trapdoor one-way function. Injectivity[12] follows from the fact that for all but an

[12] In fact, what we prove is a slightly weaker statement. More precisely, we show that for all but an exponentially small fraction of $\mathbf{A}$, there are no two pairs $(\mathbf{s}, \mathbf{x})$ and $(\mathbf{s}', \mathbf{x}')$ such that

exponentially small fraction of $\mathbf{A}$, the minimum distance (in the ℓ_2 norm) of the lattice defined by $\mathbf{A}$ is very large; the proof is by a simple probabilistic argument and is omitted due to lack of space. Inverting the function is identical to solving $\mathsf{LWE}_{n,m,q,\beta}$. By Proposition 1, this implies that inverting the function on the average is as hard as solving $\mathsf{gapSVP}_{n^{log^3 n}}$ in the worst-case.

Trapdoor Sampling. The trapdoor for the function indexed by $\mathbf{A}$ is a short basis for the lattice $\Lambda^{\perp}(\mathbf{A}) = \{\mathbf{y} \in \mathbb{Z}^m : \mathbf{y}\mathbf{A} = 0 \bmod q\}$ defined by $\mathbf{A}$ (in a sense described below). We use here a modification of the procedure due to Ajtai [3] (and its recent improvement due to Alwen and Peikert [5]) which generates a pair $(\mathbf{A}, \mathbf{S})$ such that $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ is statistically close to uniform and $\mathbf{S} \in \mathbb{Z}^{m \times m}$ is a short basis for $\Lambda^{\perp}(\mathbf{A})$.

We outline the main distinction between [3, 5] and our theorem. Both [3] and [5] aim to construct bases for $\Lambda^{\perp}(\mathbf{A})$ that is as short as possible (namely, where each basis vector has length $\mathsf{poly}(n)$). Their proof works for the GPV parameter choices, that is $q = \mathsf{poly}(n)$ and $m = \Omega(n \log q) = \Omega(n \log n)$, for which they construct a basis $\mathbf{S}$ such that each basis vector has length $O(m^3)$ (this was recently improved to $m^{0.5}$ by [5]). In contrast, we deal with a much smaller m (linear in n) and a much larger q (superpolynomial in n). For this choice of parameters, the shortest vectors in $\Lambda^{\perp}(\mathbf{A})$ are quite long: indeed, they are unlikely to be much shorter than $q^{n/m} = q^{O(1)}$ (this follows by a simpler probabilistic argument). What we do is to construct a basis that is nearly as short; it turns out that this suffices for our purposes. Reworking the result of Ajtai for our parameters, we get the following theorem. The proof is omitted from this extended abstract.

Theorem 5. *Let $m = 6n$ and $q = n^{\log^3 n}$. There is a polynomial (in n) time algorithm that outputs a pair $(\mathbf{A}, \mathbf{S})$ such that (a) The distribution of $\mathbf{A}$ is statistically close to the uniform distribution in $\mathbb{Z}_q^{m \times n}$. (b) $\mathbf{S} \in \mathbb{Z}^{m \times m}$ is a full-rank matrix and is a short basis for $\Lambda^{\perp}(\mathbf{A})$. In particular, $\mathbf{S}\mathbf{A} = 0 \bmod q$. (c) Each entry of $\mathbf{S}$ has absolute value at most $q' = q/m^4$.*

Trapdoor Inversion. As in GPV, we use the procedure of Liu, Lyubashevsky and Micciancio [30] for trapdoor inversion. In particular, we show a procedure that, given the basis $\mathbf{S}$ for the lattice $\Lambda^{\perp}(\mathbf{A})$ from above, outputs $(\mathbf{s}, \mathbf{x})$ given $f_{\mathbf{A}}(\mathbf{s}, \mathbf{r})$ (if such a pair $(\mathbf{s}, \mathbf{x})$ exists, and $\perp$ otherwise). Formally, they show the following:

Lemma 3. *Let n, m, q, β be as above, and let L be the length of the basis $\mathbf{S}$ of $\Lambda^{\perp}(\mathbf{A})$ (namely, the sum of the lengths of all the basis vectors). If $\beta \leq 1/Lm$, then there is an algorithm that, with overwhelming probability over the choice of $(\mathbf{A}, \mathbf{S})$ output by the trapdoor sampling algorithm, efficiently computes $\mathbf{s}$ from $f_{\mathbf{A}}(\mathbf{s}, \mathbf{r})$.*

The length L of the basis output by the trapdoor sampling algorithm is at most $m^2 q' \leq q/m^2$. For our choice of parameters, namely $\beta = 4\sqrt{n}/q$, and $m = 6n$, clearly $\beta \leq 1/Lm$. Thus, the inversion algorithm guaranteed by Lemma 3 succeeds with overwhelming probability over the choice of inputs. Note that once we compute $\mathbf{s}$, we can also compute the unique value of $\mathbf{x}$. □

$\mathbf{A}\mathbf{s} + \mathbf{x} = \mathbf{A}\mathbf{s}' + \mathbf{x}'$ where $\mathbf{s}, \mathbf{s}' \in \mathbb{Z}_q^m$ and $||\mathbf{x}||_2, ||\mathbf{x}'||_2 \leq \beta\sqrt{mn}$. This does not affect the applications of injective one-way and trapdoor functions such as commitment and encryption schemes.

5 Open Questions

In this paper, we design public-key and identity-based encryption schemes that are secure against memory attacks. The first question that arises from our work is whether it is possible to (define and) construct other cryptographic primitives such as signature schemes, identification schemes and even protocol tasks that are secure against memory attacks. The second question is whether it is possible to protect against memory attacks that measure an arbitrary polynomial number of bits. Clearly, this requires some form of (randomized) refreshing of the secret-key, and it would be interesting to construct such a mechanism. Finally, it would be interesting to improve the parameters of our construction, as well as the complexity assumptions, and also to design encryption schemes against memory attacks under other cryptographic assumptions.

Acknowledgments. We thank Yael Kalai, Chris Peikert, Omer Reingold, Brent Waters and the TCC program committee for their excellent comments. The third author would like to acknowledge delightful discussions with Rafael Pass about the simultaneous hardcore bits problem in the initial stages of this work.

References

1. Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, and Pankaj Rohatgi. The em side-channel(s). In *CHES*, pages 29–45, 2002.
2. Dakshi Agrawal, Josyula R. Rao, and Pankaj Rohatgi. Multi-channel attacks. In *CHES*, pages 2–16, 2003.
3. Miklós Ajtai. Generating hard instances of the short basis problem. In *ICALP*, pages 1–9, 1999.
4. Werner Alexi, Benny Chor, Oded Goldreich, and Claus-Peter Schnorr. Rsa and rabin functions: Certain parts are as hard as the whole. *SIAM J. Comput.*, 17(2):194–209, 1988.
5. Joel Alwen and Chris Peikert. Generating shorter bases for hard random lattices. Manuscript, 2008.
6. Mihir Bellare, Marc Fischlin, Adam O'Neill, and Thomas Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In *CRYPTO*, pages 360–378, 2008.
7. Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. In *CRYPTO*, pages 278–291, 1993.
8. Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984.
9. Alexandra Boldyreva, Serge Fehr, and Adam O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In *CRYPTO*, pages 335–359, 2008.
10. Ran Canetti, Dror Eiger, Shafi Goldwasser, and Dah-Yoh Lim. How to protect yourself without perfect shredding. In *ICALP (2)*, pages 511–523, 2008.
11. Dario Catalano, Rosario Gennaro, and Nick Howgrave-Graham. Paillier's trapdoor function hides up to $O(n)$ bits. *J. Cryptology*, 15(4):251–269, 2002.
12. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In *CHES*, pages 13–28, 2002.
13. Don Coppersmith. Small solutions to polynomial equations, and low exponent rsa vulnerabilities. *J. Cryptology*, 10(4):233–260, 1997.
14. Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT*, pages 523–540, 2004.
15. Stefan Dziembowski and Krysztof Pietrzak. Leakage-resilient stream ciphers. In *To Appear in the IEEE Foundations of Computer Science*, 2008.

16. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.
17. Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32, 1989.
18. Oded Goldreich and Vered Rosen. On the security of modular exponentiation with application to the construction of pseudorandom generators. *Journal of Cryptology*, 16:2003, 2000.
19. Shafi Goldwasser, Yael Kalai, Chris Peikert, and Vinod Vaikuntanathan. Manuscript, in preparation, 2008.
20. Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. One-time programs. In *CRYPTO*, pages 39–56, 2008.
21. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.
22. Alex Halderman, Seth Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph Calandrino, Ariel Feldman, Jacob Appelbaum, and Edward Felten. Lest we remember: Cold boot attacks on encryption keys. In *Usenix Security Symposium*, 2008.
23. Johan Håstad and Mats Näslund. The security of individual rsa bits. In *FOCS*, pages 510–521, 1998.
24. Johan Håstad, A. W. Schrift, and Adi Shamir. The discrete logarithm modulo a composite hides o(n) bits. *J. Comput. Syst. Sci.*, 47(3):376–404, 1993.
25. Yuval Ishai, Manoj Prabhakaran, Amit Sahai, and David Wagner. Private circuits ii: Keeping secrets in tamperable circuits. In *EUROCRYPT*, pages 308–327, 2006.
26. Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In *CRYPTO*, pages 463–481, 2003.
27. Burton S. Kaliski. A pseudo-random bit generator based on elliptic logarithms. In *CRYPTO*, pages 84–103, 1986.
28. Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *CRYPTO*, pages 104–113, 1996.
29. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *CRYPTO*, pages 388–397, 1999.
30. Yi-Kai Liu, Vadim Lyubashevsky, and Daniele Micciancio. On bounded distance decoding for general lattices. In *APPROX-RANDOM*, pages 450–461, 2006.
31. Douglas L. Long and Avi Wigderson. The discrete logarithm hides o(log n) bits. *SIAM J. Comput.*, 17(2):363–372, 1988.
32. Side-Channel Cryptanalysis Lounge, 2008. http://www.crypto.rub.de/en_sclounge.html.
33. Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In *TCC*, pages 278–296, 2004.
34. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. Cryptology ePrint Archive, Report 2008/481, 2008. http://eprint.iacr.org/.
35. Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO*, pages 554–571, 2008.
36. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *STOC*, pages 187–196, 2008.
37. Christophe Petit, François-Xavier Standaert, Olivier Pereira, Tal Malkin, and Moti Yung. A block cipher based pseudo random number generator secure against side-channel key recovery. In *ASIACCS*, pages 56–65, 2008.
38. Krzysztof Pietrzak and Vinod Vaikuntanathan, 2009. Personal Communication.
39. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.
40. Alon Rosen and Gil Segev. Chosen-ciphertext security via correlated products. Cryptology ePrint Archive, Report 2008/116, 2008.
41. Umesh V. Vazirani and Vijay V. Vazirani. Efficient and secure pseudo-random number generation. In *CRYPTO*, pages 193–202, 1984.
42. Andrew C. Yao. Theory and application of trapdoor functions. *Symposium on Foundations of Computer Science*, 0:80–91, 1982.