On Cryptography with Auxiliary Input

Yevgeniy Dodis[*]
New York University
dodis@cs.nyu.edu

Yael Tauman Kalai
Microsoft Research
yael@microsoft.com

Shachar Lovett[†]
Weizmann Institute
shachar.lovett@weizmann.ac.il

ABSTRACT

We study the question of designing cryptographic schemes which are secure even if an *arbitrary* function $f(sk)$ of the secret key is leaked, as long as the secret key sk is still (exponentially) hard to compute from this auxiliary input. This setting of auxiliary input is more general than the more traditional setting, which assumes that some of information about the secret key sk may be leaked, but sk still has high min-entropy left. In particular, we deal with situations where $f(sk)$ *information-theoretically determines* the entire secret key sk.

As our main result, we construct CPA/CCA secure symmetric encryption schemes that remain secure with exponentially hard-to-invert auxiliary input. We give several applications of such schemes.

- We construct an average-case obfuscator for the class of point functions, which remains secure with exponentially hard-to-invert auxiliary input, and is reusable.
- We construct a *reusable* and *robust* extractor that remains secure with exponentially hard-to-invert auxiliary input.

Our results rely on a new cryptographic assumption, Learning Subspace-with-Noise (LSN), which is related to the well known Learning Parity-with-Noise (LPN) assumption.

Categories and Subject Descriptors

E.3 [**Data**]: Data Encryption; E.4 [**Data**]: Coding and Information Theory; H.3.2 [**Information Systems**]: Information Storage

General Terms

Theory, Security, Algorithms

[*]Supported by NSF Grants 0831299, 0716690, 0515121.

[†]Supported by ISF grant 1300/05. Work done while visiting Microsoft Research.

STOC'09, May 31–June 2, 2009, Bethesda, Maryland, USA.

Keywords

Encryption Schemes, Error-Correcting Codes, Learning Parity with Noise, Auxiliary Information, Code Obfuscation, Randomness Extractors

1. INTRODUCTION

In cryptography, most traditional security definitions assume that no information about the secret-key is leaked, beyond a well defined "interface" between the challenger and the attacker. In reality, however, the adversary may obtain additional information about the secret key. This could be due to unexpected attacks, such as side-channel attacks, or, alternatively, users may *prefer* to use the same secret key for multiple tasks. For example, some users may want to use the same biometric data as a secret key for various applications [8, 22]; or some users may want to use identity based cryptography [52], once again using the same key (corresponding to their identity) for various applications.

Motivated by these considerations, in this work we study the question of designing cryptographic schemes that are secure even with respect to *arbitrary auxiliary input* $f(sk)$ about the secret key sk, as long as sk remains (exponentially) hard to compute given the auxiliary input $f(sk)$. We stress that we consider even auxiliary input that *information-theoretically determines* the entire secret key sk. However, as a special case, our results also apply to the setting where the secret key sk still has (linear) min-entropy given $f(sk)$, which is already interesting and highly non-trivial.

The study of cryptography with auxiliary input was initiated by Canetti [10] in the context of perfect one-way functions, and was later studied by [29] in the context of obfuscation. However, the latter work presents mainly negative results, demonstrating the *impossibility* of obfuscation with auxiliary input. The former shows a positive result; i.e., presents a perfect one-way function that is secure with auxiliary input. However, the assumption that it relies on quantifies over all (polynomially) hard-to-invert auxiliary inputs $f(sk)$.[1] Thus, instead of dealing with the auxiliary input directly, the auxiliary input was pushed into the assumption. Additionally, the assumption is not necessarily "efficiently falsifiable" [44], since refuting it might require a proof that a given leakage function f is hard-to-invert. In contrast, our goal in this work is to *construct cryptographic schemes that remain secure with (exponentially) hard-to-invert aux-*

[1]Loosely speaking, the assumption is that it is hard to invert the (randomized) function $H(x,r) = (r, r^x \mod p)$, even given *any* hard-to-invert auxiliary input $f(x)$.

iliary input, under an efficiently falsifiable assumption [44] which does not quantify over all auxiliary functions f.

We note that there is vast literature on cryptography with respect to different types of leakage, all of which assumes that the secret key still has min-entropy, conditioned on the leakage. We discuss this in Section 1.2.

1.1 Our Main Results

We argue that a natural assumption to rely on, in this context, is the (decisional)[2] *Learning Parity with Noise* (LPN) assumption, which states that any polynomial size set of "noisy parities" of a random secret x is computationally indistinguishable from a sequence of truly random bits. More precisely, for a random vector $x \in \{0,1\}^n$ and any polynomial $t = \text{poly}(n)$, the distributions $(A, Ax + e)$ and (A, U) are computationally indistinguishable, where A is a random $t \times n$ boolean matrix, $e \in \{0,1\}^t$ is a "small", randomly generated error vector, and $U \in \{0,1\}^t$ is uniform.

The LPN assumption, beyond its simplicity and efficiency, has several additional advantageous properties. For example, it is "composable" (see Section 2) and is easily seen to be resistant to the leakage of any constant fraction of the bits of x. In fact, we believe that it is also robust against *arbitrary* auxiliary input $f(x)$ which is exponentially hard-to-invert, at least when the noise e is relatively high. Let us call this assumption *auxiliary-input LPN*. However, simply making the auxiliary-input LPN assumption, without any justification, would defeat our goal of basing our schemes on an efficiently falsifiable assumption which does not quantify over all auxiliary functions f.

Instead, our schemes rely on a generalization of the LPN assumption, which we call the *learning subspace with noise* (LSN) assumption. The LSN assumption (see Section 2) is efficiently falsifiable and does not quantify over all auxiliary functions f. Nevertheless, we show (Theorem 1) that *the LSN assumption implies the auxiliary-input LPN assumption* (for high-enough noise). Moreover, the proof of Theorem 1 implies that, under the LSN assumption, the LPN assumption holds also w.r.t. min-entropy, rather than truly random, secrets. We note that the proof of Theorem 1 constitutes the bulk of the technical difficulties of this work.

Thus, the LSN assumption formally demonstrates the appeal of basing cryptographic schemes on the *auxiliary-input* LPN assumption. Indeed, we use the auxiliary-input LPN assumption (implied by the LSN assumption) to build a variety of cryptographic schemes secure with respect to exponentially hard-to-invert auxiliary input. These applications are described below.

We note that in all our applications, security w.r.t. auxiliary input can be replaced with the (incomparable) assumption that the secret key is taken from an arbitrary distribution with linear min-entropy, rather than being uniform.

Symmetric Encryption Schemes. As our main application, under the auxiliary-input LPN assumption (or the standard LSN assumption), we construct CPA/CCA secure symmetric encryption schemes that remain secure w.r.t. exponentially hard-to-invert auxiliary input (where the CCA scheme also requires the existence of trapdoor permutations).

[2]Typically, a computational form of the LPN assumption is used. However, since the decisional variant we use is known to be equivalent to the computational variant [50], and is much more convenient for cryptographic applications, this is the only variant we use in this paper.

Our CPA scheme is very simple. To encrypt m, output $(A, c) = (A, Ax + e + \text{ECC}(m))$, where x is the secret key, $\text{ECC}(m)$ is an appropriate error-correcting encoding of m, A is a random matrix and e is a random noise vector. Given the secret key x, one can recover $c+Ax = e+\text{ECC}(m)$, from which one can recover m. The CPA security of this scheme follows very easily from the LPN assumption. Similarly, the CPA security w.r.t. exponentially hard-to-invert auxiliary input follows from the auxiliary-input LPN assumption.

Our CCA scheme is a bit more complicated, and follows a variant of the Naor-Yung "double encryption" paradigm [45], applied to our simple CPA encryption scheme and a *public-key CCA-secure* encryption scheme. The public-key scheme does not need to have any security w.r.t. auxiliary input, since its secret key is never stored. It is essentially used to encrypt the secret key x as part of the ciphertext. By coupling this encryption with an appropriate simulation-sound non-interactive zero-knowledge proof [51], we overcome the need to store the secret key corresponding to the public-key scheme. Moreover, we argue that the decryption oracle is "useless" to the attacker. Intuitively, producing a valid ciphertext would allow one to recover the secret key x from the public-key encryption of x, which we argue is impossible. In particular, our resulting CCA scheme is actually a symmetric-key *authenticated encryption*, which is a stricter notion than CCA security [5]. See Section 4.2 for details.

We use our encryption schemes to construct *average-case obfuscators with auxiliary input*, and to construct *robust and reusable extractors*.

Average-Case Obfuscation with Auxiliary Input. An obfuscator is a compiler that takes any program and transforms it into a garbled program, that has the same input-output functionality, but is "unintelligible". Barak *et al.* [3] initiated a theoretical study of obfuscation, and demonstrated the impossibility of obfuscation by presenting a family of functions that cannot be obfuscated. Thereafter, several results on obfuscation have emerged [33, 42, 53, 35, 34, 31, 11]. We stress that the original definition of obfuscation did not consider security in the presence of auxiliary input, nor did all of the results mentioned above. Obfuscation w.r.t. auxiliary input was defined in [29], who exhibited mainly negative results.

We use our CPA encryption scheme (and thus rely on the LSN or the auxiliary-input LPN assumption) to construct an average-case obfuscator w.r.t. auxiliary input for the class of extended point functions (such point function $I_{x,m}$ outputs m if the input equals x and outputs $\perp$ otherwise). The construction is very simple: the obfuscator for $I_{x,m}$ contains the description of the ciphertext $c = \mathcal{E}_x(m)$. Given c and any input y, the obfuscated program returns the decryption $\mathcal{D}_y(c)$. We note that to prove correctness and security of this obfuscator we rely on two additional properties of our CPA-secure encryption scheme: the *unique-key* property and the *key-hiding* property, which are defined in Section 4 (Definitions 3 and 4, respectively). See Section 5 for details.

We stress that obfuscator is only secure *on average*, which is a useful notion of obfuscation considered by [34] (and, implicitly, earlier by [14, 23]). In other words, instead of requiring that an efficient adversary cannot gain any information from a garbled circuit (beyond black-box access) *for every* function in the family we are trying to obfuscate, we require that this holds only for a *random* function in the family. So, in that respect our definition is weaker. On the positive side, our definition allows the adversary to have auxiliary input (that can depend on the function being obfuscated).

Similarly to [34], we argue that for many cryptographic applications average-case obfuscation suffices, whereas the assumption of lack of auxiliary input may actually be problematic. Also, we did not know of any obfuscator (even for the class of point functions) that is secure with auxiliary input (except for the one due to Canetti [10], which was given in the context of perfect-one-way functions, and relies on an assumption quantifying over all auxiliary inputs).

We also stress that our obfuscator is *reusable*, in the sense that even if the adversary gets many obfuscations of the same function, all he learns is what he could have learned from black-box access to the function. Finally, we note that our obfuscator is closely related to a perfect one-way function [10, 14]. See Section 5 for more details.

Reusable Extractors. Standard randomness extractors [47] allow one to extract nearly uniform (statistically secure) randomness Y from any distribution X having some min-entropy, by using an additional random seed R. Strong extractors require that Y is nearly uniform even given the seed R, and are very appealing for many cryptographic applications. For example, consider the scenario where two users, Alice and Bob, share a secret key X, but believe that some information about their secret-key may be leaked (say due to side-channel attacks). Using strong extractors, they can "renew" this key as follows: Alice chooses a random seed R, computes $Y = \mathrm{Ext}(X, R)$, and sends the seed R to Bob (over a public channel), while Bob can reconstruct the "new secret" Y by running $Y = \mathrm{Ext}(X, R)$.

Unfortunately, standard extractors do not allow one to extract many (computationally) random secrets Y_i from the same X. In a *reusable* extractor we would like to ensure that, for any polynomial q, the independently extracted keys $Y_1, \ldots, Y_q$ all look (computationally) random and independent even conditioned on $R_1, \ldots, R_q$. We argue that reusable extractors are important for cryptographic applications, since the users do not need to remember anything beyond X in order to derive a fresh secret. The problem of designing such reusable extractors was implicit as early as [14], and is explicitly mentioned in [8, 22].

Reusable extractors are known to exist in the following settings: (a) in the random oracle model [8]; (b) in the CRS model where the distribution of X is *independent* of the CRS (trivial application of extractors and weak pseudorandom functions); or (c) in the standard model under a strong variant of the DDH, which assumes that DDH remains secure even if the secret only has min-entropy [10] (i.e., the assumption quantifies over all min-entropy sources, and assumes that *for every* min-entropy source $\mathcal{X}$ the DDH remains secure when the secret is distributed according $\mathcal{X}$). Finally, we mention the recent independent work of Pietrzak [48], which also implies such a reusable extractor, assuming the min-entropy rate of the sources is greater than 1/2 [54], and under the assumption that exponentially-secure weak pseudorandom functions exist.

In this work we construct a reusable extractor under the auxiliary-input LPN (or standard LSN) assumption. We use our CPA-secure scheme to construct a reusable extractor, for the case that X has linear min-entropy. More generally, our extractor remains secure even in the case of auxiliary input, as long as an efficient adversary can guess X only with exponentially small probability. We note that our extractor is not a strong extractor, but has the property that instead of publishing the randomness R, one can publish a "helper information" P such that the extracted randomness Y looks random even given P, and Y can be reconstructed given P and X. This is enough for all our applications (note, strong extractors correspond to $P = R$).

The actual extractor is very simple: pick a random key Y of desired length, and set P to be the CPA-secure encryption $\mathcal{E}_X(Y)$ of Y, keyed by X. The security and reusability of this extractor follows immediately from the CPA-security of our encryption w.r.t. weak keys. Further details are given in the full version [19].

Robust Extractors. We also construct *robust extractors*, a notion defined by [9, 20] in the context of biometrics. Robust extractors address the following concern regarding the authenticity of the helper information P. Since the user only wishes to remember X, he needs to retrieve the helper information P from somewhere. What if the user does not trust the party providing this information, or this information was changed in transit? Robust extractors solve this problem: it is infeasible to change P into some P', without the user noticing this with high probability. Moreover, the attacker should not succeed to produce a valid $P' \neq P$, even when given the pair (P, Y), as opposed to just P (since the user may have already used the extracted key Y).

Prior to this work, robust extractors were constructed in the following settings: (a) in the random oracle model [9]; (b) in the plain model for X with min-entropy rate greater than 1/2 [20]; and (c) in the CRS model for any min-entropy, but assuming that the distribution of X is independent of the CRS (trivial application of extractors).

In this work, we give a construction of robust extractors in the CRS model, but *allowing the distribution of X to depend of the CRS*, which is important for applications to side-channel attacks and exposure-resilient cryptography. Our extractor works for linear min-entropy; or more generally, we allow the attacker to learn arbitrary auxiliary information $f(X, CRS)$ as long as a PPT adversary cannot guess X with more than exponentially small probability.

We remark that our robust extractor is *also reusable*. In fact, our robust extractor is the same as our reusable extractor above, except we use our CCA-secure encryption scheme (which uses a CRS) instead of our CPA-secure scheme. Intuitively, the non-malleability of the encryption will ensure the robustness of the corresponding reusable extractor. See the full version [19] for more details.

Polynomially-hard-to-invert Leakage? In this work we only deal with exponentially hard-to-invert auxiliary input. It is natural to ask if there exist cryptographic schemes that remain secure even when f is polynomially hard-to-invert? In the full version [19] we argue that constructing such cryptographic schemes appears to be significantly more difficult, and seems to require new techniques. However, dealing with functions f that are sub-exponentially (or even quasi-polynomially) hard-to-invert may be achievable, and we leave these as open problems for future work.

Adaptive Leakage. So far we mainly talked about what we call *static* auxiliary input, where the leaked function f is specified before the secret gets used (e.g., in the context of encryption, before the attacker starts asking encryption or decryption queries). In Section 4.3, we extend our results to the more general case of *adaptive* leakage, concentrating on the case of symmetric encryption schemes.

Intuitively, before the scheme is used, we allow the attacker to learn any "static" function $f(sk)$, such that sk is still hard to compute with probability better than $2^{-\alpha n}$ (where $n = |sk|$ and $\alpha > 0$). Then, as the encryption scheme starts to be used, and the adversary starts getting valid ciphertexts (or even access to the decryption oracle), we allow the adversary to adaptively learn at most βn additional *bits of information* about sk, as long as $\beta + \alpha < 1$. Then the adversary gets the challenge ciphertext, as usual, but cannot learn any more information about sk. In Section 4.3, we argue that our particular CPA/CCA-secure encryption schemes satisfy this extended security w.r.t. adaptive leakage, and also that it is unclear how to define a stronger, yet meaningful form of adaptive leakage.

1.2 Related Work

The question of designing cryptographic schemes resistant to leakages of arbitrary (hard-to-invert) auxiliary information, was studied, in different contexts, by Canetti [10] and Goldwasser and Kalai [29]. However, prior to this work, there were no schemes resisting such attacks whose security was based on an "efficiently falsifiable" assumption which did not quantify over all auxiliary leakage functions f.

On the other hand, there is a large body of work (surveyed below) which constructs various cryptographic primitives under the assumption that the *secret key has min-entropy*: namely, that it is *information-theoretically* hard for the attacker to guess the secret.

This study was initiated in the context of *exposure-resilient cryptography* [18, 12],where the attacker is restricted to learn individual bits of the secret. In a more advanced setting, Ishai *et al.* [39, 38] consider the scenario where the adversary can read (or even tamper with) a bounded number of wires in some secret circuit.

Recently, Dziembowski and Pietrzak [27, 48] considered the question of building stream ciphers secure against *arbitrary* side channel attacks, where the attacker can learn an arbitrary shrinking function of the secret key, as long as this function satisfies the axiom "only computation leaks information" of Micali and Reyzin [43]. However, they require that the secret key has at least $n/2$ min-entropy left [54].

Very recently, Akavia, Goldwasser, and Vaikuntanathan [1] considered security against memory attacks: these are more general than side channel attacks, since they allow leakage even of information that was not used (but only stored). They present a public-key encryption scheme that is semantically secure against arbitrary memory attacks, as long as the secret key has at least $n(1 - \frac{1}{\log n})$ min-entropy left.

Finally, independent of this work, is the recent result of Pietrzak [48], which implicitly gives an alternative construction of a CPA-secure encryption (and a reusable extractor) which is secure assuming the secret key has at least $n/2$ min-entropy left, and assuming the existence of exponentially secure weak pseudorandom functions.

To summarize, all the results mentioned above require the secret key to have at least $n/2$ min-entropy left, and do not deal with the (more general) case of auxiliary input. In contrast, our work can tolerate any constant fraction of leakage (even 99% leakage), and more generally can deal with situations where the secret key doesn't have any min-entropy, but is (computationally) *unpredictable*.

Another related "min-entropy" setting is that of *bounded retrieval model* [17, 24, 25, 15, 26], where the secrets are made intentionally huge and the attacker is assumed not to have enough bandwidth to read all of the secret upon break-in. Yet another related setting was recently considered by Canetti et al. [13], who assume that most memory can only be *partially* erased, giving the attacker some side information about the secrets. However, their schemes still require a (very small) part of the memory to be perfectly erasable.

Imperfect secrets naturally come up in the context of biometrics [21, 22], where users wish to use biometrics as keys for various cryptographic applications. One issue here is to deal with noise or "fuzziness": repeated readings of the same biometric are likely to be different (although "close"). However, even ignoring this "fuzziness" issue, a major drawback of the existing solutions, first pointed out by Boyen [8], comes from the fact that it was not known how to reuse the same biometric secret X for multiple applications. In other words, how to construct a *reusable* (fuzzy) extractor capable of extracting an unbounded number of computationally uncorrelated keys from the same biometric X? In this work, we propose a solution to this problem (in the "non-fuzzy" scenario), even in the more complicated setting of auxiliary information as opposed to min-entropy.

In terms of reusing the same secret for multiple tasks, we mention the works of [32, 16], who considered the question of doing so for signing and encrypting in the public key setting. Our work is much more general, as it considers *arbitrary* exponentially hard-to-invert auxiliary information, as opposed to those specified by a particular (secure) application.

We also mention that the LPN assumption, whose variant is used in this work, is getting increasingly used in various cryptographic applications; e.g., [6, 36, 40, 41]. In particular, we recently learned that a scheme, similar to our CPA-secure scheme, was independently discovered by Applebaum [2], in the context of building encryption schemes with key-dependent security. We stress that our notion of security is incomparable to the notion studied in [2]. In our case, the adversary is allowed to see arbitrary exponentially hard-to-invert information about the key, whereas security under key-dependent inputs allows the adversary to ask for encryptions of messages that depend on the key.

Finally, the notion of exponentially hard-to-invert auxiliary input is closely related to the notion of (computational) *unpredictability entropy* recently defined by [37] (see also [4]).

2. THE LPN AND LSN ASSUMPTIONS

We model PPT algorithms as families of poly-size circuits, use $\approx$ to denote standard computational indistinguishability, and let $\mathrm{negl}(n)$ denote a negligible function of n.

DEFINITION 1. *A polynomial-time computable function $f : \{0,1\}^* \to \{0,1\}^*$ is α-`exponentially hard-to-invert` if for every* PPT *adversary $\mathcal{A}$ and for every large enough n,*

$$\Pr_{x\in\{0,1\}^n}[\mathcal{A}(f(x)) = x] \leq 2^{-\alpha n}.$$

A function f is `exponentially hard-to-invert` *if there exists a constant $\alpha > 0$ such that f is α-exponentially hard-to-invert.*

THE LPN ASSUMPTION. Let us first recall the standard LPN assumption (see Footnote 2): for every constant $\gamma > 0$ and for every polynomial $t = \text{poly}(n)$, we have

$$\{A, Ax + e\}_{n\in\mathbb{N}} \approx \{A, U_t\}_{n\in\mathbb{N}}$$

where $A \in_R \{0,1\}^{t\times n}$, $x \in_R \{0,1\}^n$ and $U_t \in_R \{0,1\}^t$ are all uniformly distributed, and where $e = (e_1, \ldots, e_t) \in \{0,1\}^t$ is distributed as follows:

$$\begin{array}{ll} e_i = 0 & \text{with probability } \gamma \\ e_i \in_R \{0,1\} & \text{with probability } 1-\gamma \end{array}$$

The LPN assumption has some every appealing properties.

1. **Efficiency.** The LPN function $g(x; A, e) = Ax+e$ can be computed extremely efficiently. Thus, the LPN assumption is very suited for applications where efficiency is a main consideration.
2. **Closure under composition.** The LPN assumption is closed under composition. Namely, for any polynomial ℓ,
$$\left\{\left(A^{(i)}, A^{(i)}x + e^{(i)}\right)_{i=1}^{\ell}\right\}_{n\in\mathbb{N}} \approx \left\{\left(A^{(i)}, U^{(i)}\right)_{i=1}^{\ell}\right\}_{n\in\mathbb{N}}$$
3. **Robustness against bit-leakage**. If some k bits of x are leaked then the LPN assumption still holds, but the security parameter decreases from n to $n-k$. Moreover, the LPN assumption is robust against any *linear* leakage function (as before, if a total of k bits are leaked the security parameter decreases from n to $n-k$). This can be seen by applying a change of basis.

We would like to argue that the LPN assumption is robust against *any* (polynomial-time) leakage function $f(x)$, as long as at most $(1-\alpha)n$ bits are leaked, for some constant $\alpha > 0$. In fact, we could go even further and make the following assumption, which we call *auxiliary-input LPN* assumption: for any $\alpha > 0$, and any α-exponentially hard-to-invert function f, the standard LPN assumption holds even in the presence of auxiliary input $f(x)$. Namely,

$$\{f(x), A, Ax + e\}_{n\in\mathbb{N}} \approx \{f(x), A, U_t\}_{n\in\mathbb{N}}.$$

However, given that the auxiliary-input LPN assumption is not efficiently falsifiable and quantifies over all functions f, we would like to establish this assumption under a more standard assumption, such as standard LPN. Unfortunately, we were not able to use the standard LPN assumption.

Instead, we introduce a generalization of the LPN assumption, which we call the LSN (*Learning Subspace with Noise*) assumption. As our main technical result, we show that the LSN assumption implies the auxiliary-input LPN assumption, at least when the noise rate is "high" (in particular, polynomially close to 1; see Theorem 1).

Loosely speaking, our new LSN assumption asserts that it is hard to learn a random subspace $V \subseteq \{0,1\}^n$ of dimension αn, if there is "enough" noise.

THE LSN ASSUMPTION. For any constant $\alpha > 0$ there exists a polynomial $p(n) = p_\alpha(n)$ such that for any polynomial $t(n)$ the following two distributions are computationally indistinguishable:

- $(r_1, ..., r_t)$, where a random subspace $V \subseteq \{0,1\}^n$ of dimension $k = \alpha n$ is chosen, and where each $r_i \in \{0,1\}^n$ is chosen independently according to the following distribution:
$$\begin{array}{ll} r_i \in_R V & \text{with probability } \frac{1}{p(n)} \\ r_i \in_R \{0,1\}^n & \text{with probability } 1 - \frac{1}{p(n)} \end{array}$$
- $(r_1, ..., r_t)$, where each $r_i \in_R \{0,1\}^n$ is chosen independently at random.

Notice that if the dimension of V was $n-1$, then this would be exactly the LPN assumption with very high noise rate $(1-\gamma) = (1 - \frac{1}{\text{poly}(n)})$. In other words, our assumption corresponds to the most conservative (i.e., believable) variant of the LPN assumption in terms of the noise, but for a much smaller-dimensional subspace V than the standard LPN assumption.

It was observed by Raz [49] that our LSN assumption is false (for V of dimension αn) if we take only a constant noise rate $(1-\gamma) < 1$, as opposed to the noise rate $(1 - 1/\text{poly}(n))$ (which approaches 1 as n grows). Currently, for any constants $\epsilon, \alpha > 0$, Raz's specific attack does not go through even for $p_\alpha(n) = n^\epsilon$.[3] More details about Raz's attack appear in the full version [19].

To make our assumption as weak as possible we allow p_α to depend on α. However, since the polynomial $p = p_\alpha$ must be known by our schemes, whose efficiency is indeed proportional to $p(n)^2$, there is a good motivation to try making the degree of p as small as possible (perhaps independent of α), as long as there is evidence that it does not violate the LSN assumption. We leave this exploration to future work. However, for the sake of concreteness, the reader may think of $p_\alpha(n) = n$ for every constant $\alpha > 0$.

3. TECHNICAL CONTRIBUTION

We next state our main technical result, which essentially says that, under the LSN assumption, the (standard) LPN assumption is robust against any exponentially hard-to-invert auxiliary input, at least in the high noise regime.

THEOREM 1. *The LSN assumption implies the "high-noise" auxiliary-input LPN assumption. More precisely, for any constant $\alpha > 0$ there exists a polynomial p such that for any function $f : \{0,1\}^* \to \{0,1\}^*$ that is α-exponentially hard-to-invert, and for any polynomial t,*

$$\{f(x), A, Ax + e\}_{n\in\mathbb{N}} \approx \{f(x), A, U_t\}_{n\in\mathbb{N}}$$

where U_t is the uniform distribution over $\{0,1\}^t$, $x \in_R \{0,1\}^n$, $A \in_R \{0,1\}^{t\times n}$, and $e = (e_1, \ldots, e_t) \in \{0,1\}^t$ is distributed as follows:

$$\begin{array}{ll} e_i = 0 & \textit{with probability } \frac{1}{p(n)} \\ e_i \in_R \{0,1\} & \textit{with probability } 1 - \frac{1}{p(n)} \end{array}$$

In the proof of Theorem 1 we rely on the following well known lemma, which is a generalization of the Goldreich-Levin hard-core predicate theorem [30].

LEMMA 3.1 (GL89, COROLLARY 1). *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a poly-time computable function that is exponentially hard-to-invert; i.e., there exists a constant $\alpha > 0$ such that for every* PPT *adversary $\mathcal{A}$,*

$$\Pr[\mathcal{A}(f(x)) \in f^{-1}(f(x))] \leq 2^{-\alpha n}.$$

Then, for every constant $\beta < \alpha$

$$\{(f(x), B, Bx)\}_{n\in\mathbb{N}} \approx \{(f(x), B, U_{\beta n})\}_{n\in\mathbb{N}},$$

where $x \in_R \{0,1\}^n$ and $B \in_R \{0,1\}^{\beta n\times n}$.

[3] However, Raz's attack would work even in this case, but using quasi-polynimial number of samples and quasi-polynomial time.

A significant part of the proof of Theorem 1 consists of proving the following claim.

Claim 3.2. *Assume that the LSN assumption holds. Then for any constant β there exists a polynomial q such that for any polynomial t,*

$$\{x, A, Ax + e\}_{n\in\mathbb{N}} \approx \{x, A', A'x + e'\}_{n\in\mathbb{N}} \qquad (1)$$

where $x \in_R \{0,1\}^n$ and $A \in_R \{0,1\}^{t\times n}$ are independently and uniformly distributed, $e = (e_1, \dots, e_t) \in \{0,1\}^t$ is distributed by

$$\begin{array}{ll} e_i = 0 & \text{with probability } \frac{1}{q(n)} \\ e_i \in_R \{0,1\} & \text{with probability } 1 - \frac{1}{q(n)} \end{array}$$

The matrix A' is distributed by choosing a random subspace $V \subseteq \{0,1\}^n$ of dimension βn, and then choosing each row a'_i independently as follows

$$\begin{array}{ll} a'_i \in_R V & \text{with probability } \frac{1}{q(n)} \\ a'_i \in_R \{0,1\}^n & \text{with probability } 1 - \frac{1}{q(n)} \end{array}$$

The vector e' is distributed as follows: For each $i \in [t]$, if a'_i was chosen at random from V then set $e'_i = 0$; and if a'_i was chosen at random from $\{0,1\}^n$ then choose $e'_i \in_R \{0,1\}$ at random.

Proof of Claim 3.2. Fix any constant β. Let p be the polynomial as in the LSN assumption, $q(n) \triangleq 2p(n)$, $\delta(n) \triangleq \frac{1}{q(n)}$, and let t be any polynomial. We prove Equation (1) for this q. Notice, Equation (1) is equivalent to $\{x, A, e\}_{n\in\mathbb{N}} \approx \{x, A', e'\}_{n\in\mathbb{N}}$. And since A, A', e, e' are independent of x, it suffices to show that

$$\{\{a_i, e_i\}_{i=1}^t\}_{n\in\mathbb{N}} \approx \{\{a'_i, e'_i\}_{i=1}^t\}_{n\in\mathbb{N}} \qquad (2)$$

Without the noise e and e', this would be exactly the LSN assumption, which would be true even with $q = p$ (as opposed to $q = 2p$). However, the noise e'_i is correlated with the way a'_i was chosen, so we need to argue that this dependence does not prevent us from using the LSN assumption, at least when $q = 2p$. For that matter, notice that e_i and e'_i are identically distributed over $\{0,1\}$. Thus, to show Equation (2), it is enough to show that for every bit $b \in \{0,1\}$, the following conditional distributions are indistinguishable:

$$\{\{a_i \text{ s.t. } e_i = b\}_{i=1}^t\}_{n\in\mathbb{N}} \approx \{\{a'_i \text{ s.t. } e'_i = b\}_{i=1}^t\}_{n\in\mathbb{N}} \qquad (3)$$

This claim is obvious for $b = 1$, since when $e_i = e'_i = 1$, both a_i and a'_i are truly random in $\{0,1\}^n$. So let us turn to $b = 0$; i.e. $e_i = e'_i = 0$. In this case, a_i is still truly random. On the other hand, conditioned on $e'_i = 0$, a'_i is distributed according to the following distribution:

(1) With probability δ, sample $a'_i \in_R V$.

(2) With probability $\frac{1-\delta}{2}$, sample $a'_i \in_R \{0,1\}^n$.

(3) With probability $\frac{1-\delta}{2}$, go back to step (1). (This corresponds to $e'_i = 1$).

However, it is easy to see that the above distribution is identical to the following simpler distribution:

$$\begin{array}{ll} a'_i \in_R V & \text{with probability } \eta \\ a'_i \in_R \{0,1\}^n & \text{with probability } 1 - \eta \end{array}$$

where $\eta = \sum_{i\geq 0} \delta \left(\frac{1-\delta}{2}\right)^i \leq 2\delta = \frac{1}{p(n)}$. Thus, for $b = 0$, Equation (3) is implied by the LSN assumption. ∎

Proof of Theorem 1. Fix any constant $\alpha > 0$, and fix any function f that is hard-to-invert with probability $2^{-\alpha n}$. Lemma 3.1 implies that for any constant $\beta < \alpha$

$$\{(f(x), B, Bx)\}_{n\in\mathbb{N}} \approx \{(f(x), B, U_{\beta n})\}_{n\in\mathbb{N}} \qquad (4)$$

where $x \in_R \{0,1\}^n$ and $B \in_R \{0,1\}^{\beta n\times n}$. Fix an arbitrary constant $0 < \beta < \alpha$. Let q be the polynomial given by Claim 3.2, and let $p \geq q$ be a large enough polynomial so that the LSN assumption holds with respect to β and p.

Let $(r_1, \dots, r_t)$ be a sequence of vectors, each independently and uniformly distributed in $\{0,1\}^{n+1}$. Let $V \subseteq \{0,1\}^n$ be a random subspace of dimension βn, and let

$$V' \triangleq \{(v, vx) : v \in V\}.$$

Let $(r'_1, \dots, r'_t)$ be a sequence of vectors distributed according to the distribution below:

$$\begin{array}{ll} r'_i \in_R V' & \text{with probability } \frac{1}{p(n)} \\ r'_i \in_R \{0,1\}^{n+1} & \text{with probability } 1 - \frac{1}{p(n)} \end{array}$$

Equation (4) implies that given $f(x)$, the subspace V' is computationally indistinguishable from a uniformly chosen subspace of dimension βn. This, together with the LSN assumption, implies that

$$\{(f(x), r_1, \dots, r_t)\}_{n\in\mathbb{N}} \approx \{(f(x), r'_1, \dots, r'_t)\}_{n\in\mathbb{N}}.[4]$$

Comparing the above to the statement of Theorem 1, it is therefore sufficient to show that

$$\{f(x), \{(a_i, a_i x + e_i)\}_{i=1}^t\}_{n\in\mathbb{N}} \approx \{f(x), \{r'_i\}_{i=1}^t\}_{n\in\mathbb{N}}, \quad (5)$$

where $(r'_1, \dots, r'_t)$ are distributed as above, $(a_1, \dots, a_t)$ are uniform and independent in $\{0,1\}^n$, and $(e_1, \dots, e_t)$ are distributed as in the statement of the theorem. However, this follows immediately from Claim 3.2, which asserts that Equation (5) holds even if x was known. ∎

For the sake of concreteness, note that the LSN assumption with $p_\alpha = n$ (i.e., with noise $1 - \frac{1}{n}$) implies Theorem 1 with $p = 2n$. In the subsequent sections, we refer to the "auxiliary-input LPN assumption" to mean precisely the variant of this assumption stated in Theorem 1.

4. SYMMETRIC ENCRYPTION SCHEMES W.R.T. AUXILIARY INPUT

4.1 CPA Security

Our definition of CPA security for symmetric-key encryption $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is entirely analogous to the standard definition, except we give the adversary some auxiliary α-exponentially hard-to-invert information $f(k)$ about the secret key k. We also notice that the efficiency of our schemes depend on α, as explained below.

Definition 2. *A symmetric-key encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with message space $\mathcal{M} = \{\mathcal{M}_n\}_{n\in\mathbb{N}}$ is* `CPA secure with exponentially hard-to-invert auxiliary input` *if for any* PPT *adversary $\mathcal{A}$, any constant $\alpha > 0$, any α-exponentially hard-to-invert function f, if the key $x \leftarrow \mathcal{K}(1^n)$*

[4] Technically, to use the LSN assumption, we need to take $r'_i \in_R V'$ with probability at most $\frac{1}{p(n+1)}$ which is smaller than $\frac{1}{p(n)}$. We take p small enough so that this won't be a problem.

and a bit $b \leftarrow \{0,1\}$ are chosen at random, then

$$\Pr[\mathcal{A}^{LR_b(x,\alpha,\cdot,\cdot)}(1^n, f(x)) = b] \leq \frac{1}{2} + \mathrm{negl}(n)$$

where the left-or-right oracle $LR_b(x,\alpha,m_0,m_1)$ takes two messages $m_0, m_1 \in \mathcal{M}_n$, and returns $\mathcal{E}_{x,\alpha}(m_b)$ if $|m_0| = |m_1|$, and $\perp$ otherwise.

Our CPA encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is extremely simple. The key generation algorithm $\mathcal{K}(1^n)$ outputs a randomly chosen secret-key $x \in_R \{0,1\}^n$ (where n is the security parameter). Given $\alpha > 0$, let $p(n)$ be the corresponding polynomial so that the auxiliary-input LPN assumption from Theorem 1 holds (for the sake of concreteness, the reader may think of $p(n) = 2n$).

For any message $m \in \{0,1\}^\ell$ (of any length $\ell \leq \mathrm{poly}(n)$), let ECC : $\{0,1\}^\ell \to \{0,1\}^{q(n)}$ be an error-correcting code that efficiently corrects (with exponentially high probability in n) up to $\frac{1}{2} - \frac{1}{2p(n)}$ random errors (meaning that every bit of the codeword is independently flipped with this probability). Such codes are known to exist for $q(n) = O((\ell+n)p(n)^2) = \mathrm{poly}(n)$ (e.g. [28]).

The encryption algorithm $\mathcal{E}(m)$ picks a random matrix $A \in_R \{0,1\}^{q(n)\times n}$, a random error vector $e \in \{0,1\}^{q(n)}$, where it sets each $e_i \in \{0,1\}$ to 0 with probability $1/p(n)$ and picks it at random otherwise, and outputs

$$\mathcal{E}_{x,\alpha}(m) = (A, Ax + e + \mathrm{ECC}(m))$$

The decryption algorithm $\mathcal{D}$ takes as input a secret key x and a ciphertext (A, c), and outputs the decoding of the vector $c + Ax$. For future convenience, we let $\mathcal{D}$ output $\perp$ if the relative distance between $c + Ax$ and the encoding ECC(m) of the decoded message m is more than $\frac{1}{2} - \frac{1}{3p}$. By choosing q large enough, it is easy to see that this happens with only exponentially small probability, so this does not affect proper ciphertexts.

Theorem 2. *Under the auxiliary-input LPN (and, hence, ordinary LSN) assumption, the symmetric encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ above is CPA secure w.r.t. exponentially hard-to-invert auxiliary input.*

The proof of this theorem is immediate, since under the auxiliary-input LPN assumption, the value

$$LR_b(x, \alpha, m_0, m_1) = (A, Ax + e + \mathrm{ECC}(m_b))$$

is indistinguishable from $(A, U_q + \mathrm{ECC}(m_b)) \equiv (A, U_q)$, which is independent of b. And the composability of the LPN assumption means we can tolerate an arbitrary polynomial number of CPA queries to the left-or-right oracle.

Additional Properties. The encryption scheme presented above has two additional properties that are useful to us in the sequel. The first property is that with high probability, there is a *single* key that decrypts an honestly generated ciphertext. We refer to this property as the *unique-key property*, and define it formally below.

Definition 3. *Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme. We say that $\mathcal{SE}$ has the* `unique-key property` *if for every $n \in \mathbb{N}$, $x \in \mathcal{K}(1^n)$, and $m \in \mathcal{M}_n$,*

$$\Pr[\exists y \neq x \text{ s.t. } \mathcal{D}_{y,\alpha}(\mathcal{E}_{x,\alpha}(m)) \neq \perp] \leq \mathrm{negl}(n).$$

Claim 4.1. *The encryption scheme $\mathcal{SE}$ presented above has the unique-key property.*

Proof. Fix any $n \in \mathbb{N}$, any secret key $x \in \{0,1\}^n$, and any message $m \in \mathcal{M}_n$.

$$\begin{aligned}
&\Pr[\exists y \neq x \text{ s.t. } \mathcal{D}_{y,\alpha}(\mathcal{E}_{x,\alpha}(m)) \neq \perp] = \\
&\Pr_{A,e}[\exists y \neq x \text{ s.t. } \mathcal{D}_{y,\alpha}(A, Ax+e) \neq \perp] \leq \\
&\sum_{y \neq x} \Pr_{A,e}[\mathcal{D}_{y,\alpha}(A, Ax+e) \neq \perp] \leq \\
&\sum_{y \neq x} \Pr_{A,e}\left[\left|\{i : e_i = A_i(x \oplus y)\}\right| \geq q\left(\frac{1}{2} + \frac{1}{3p}\right)\right] \leq \\
&\sum_{y \neq x} e^{-2q/(3p)^2} \leq 2^n \cdot e^{-n} = \mathrm{negl}(n)
\end{aligned}$$

where the first equation follows from the definition of $\mathcal{E}$; the second follows from the union bound; the third follows from the definition of $\mathcal{D}$; the forth follows from Hoeffding bounds; and the fifth follows from the fact that $q = \Omega(np^2)$ (with a large enough constant). ∎

Another useful property of our scheme is what we call the *key hiding property with exponentially hard-to-invert auxiliary input*. This property essentially says that for any message m, given any exponentially hard-to-invert auxiliary input $f(x)$ it is hard to distinguish between an encryption of m with the secret key x and an encryption of m with a random secret key y. A formal definition follows, while Claim 4.2 follows immediately from the auxiliary-input LPN assumption (or regular LSN assumption by Theorem 1).

Definition 4. *Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme with message space $\mathcal{M} = \{\mathcal{M}_n\}_{n\in\mathbb{N}}$. We say that $\mathcal{SE}$ has the* `key hiding property with exponentially hard-to-invert auxiliary input` *if for every constant $\alpha > 0$, every α-exponentially hard-to-invert auxiliary input f,*

$$\{\mathcal{E}_{x,\alpha}(m), f(x)\}_{n\in\mathbb{N}, m\in\mathcal{M}_n} \approx \{\mathcal{E}_{y,\alpha}(m), f(x)\}_{n\in\mathbb{N}, m\in\mathcal{M}_n}$$

where $x, y \in_R \mathcal{K}(1^n)$ are chosen independently at random.

Claim 4.2. *Under the auxiliary-input LPN (or standard LSN) assumption, the encryption scheme $\mathcal{SE}$ presented above has the key hiding property with exponentially hard-to-invert auxiliary input.*

4.2 CCA security

In this section we present our CCA secure scheme w.r.t. exponentially hard-to-invert auxiliary input. Due to lack of space, this section does not contain precise definitions nor proofs.

The definition of CCA security w.r.t. exponentially hard-to-invert auxiliary input is very similar to Definition 2, except we give the attacker the decryption oracle $\mathcal{D}_{k,\alpha}$, forbidding it to only decrypt ciphertexts previously returned by the left-or-right oracle. We next propose a scheme that meets this definition.

To this end, we give a general method of converting any symmetric encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ that is CPA secure w.r.t. exponentially hard-to-invert auxiliary input, such as the scheme from the previous section, into a scheme which is CCA secure w.r.t. exponentially hard-to-invert auxiliary input.

We use the commonly used paradigm (originating in [45]) of adding a component to the encryption which makes the decryption algorithm "useless". In fact, our symmetric-key scheme is an authentication encryption scheme (w.r.t. auxiliary input), which is a strictly stronger notion than CCA security [5]. In other words, our scheme is (1) CPA secure and also (2) *ciphertext unforgeable*: given oracle access to the encryption oracle $\mathcal{E}_{k,\alpha}$, it is infeasible to produce a valid ciphertext not returned by this oracle. As we mentioned, (1)+(2) imply CCA security.

For our construction, in addition to our CPA-secure scheme $\mathcal{SE}$ (w.r.t. auxiliary input), we need a CCA-secure *public-key* encryption scheme E_{pk} (without auxiliary input), a universal one-way hash function (UOWHF) $h : \{0,1\}^n \to \{0,1\}^{\epsilon n}$, where $\epsilon > 0$ is a sufficiently small constant, and a simulation-sound non-interactive zero-knowledge (NIZK) proof system Π for NP [51]. All these components exist from trapdoor permutations [45, 46, 51].

The secret key of our new scheme $\mathcal{SE}'$ consists of the original secret-key x for $\mathcal{SE}$, a hash $z = h(x)$ of x, a public-key pk for the public-key scheme E_{pk}, and a common reference string σ for the NIZK proof system. We note that we only think of x as the secret-key, and we think of z, pk, σ as public parameters. Alternatively, one can think of z, pk, σ as part of the secret-key that does not need to be kept secret. In particular, when we refer to exponentially hard-to-invert auxiliary input, we mean that it is exponentially hard-to-invert w.r.t. x (but allow the leakage function to depend on z, pk and σ). Notice, however, that since the output z of $h(x)$ is only ϵn bits, where ϵ can be made smaller than α, the leakage function f is still exponentially-hard-to-invert *even given* z. Moreover, since the CRS and the public key pk are independent of x, f should be exponentially-hard-to-invert even given the trapdoor to the CRS and the decryption key for E_{pk}.

We now describe the (authenticated) encryption and decryption procedures of our new scheme. To encrypt m, we compute $c = \mathcal{E}_{x,\alpha}(m)$, $d = \mathrm{E}_{pk}(c, x)$, and attach a NIZK proof π for the statement "the input (pk, z, c, d) satisfies the claim that there exist secret x, message m and randomness r, such that $z = h(x)$, x decrypts c to m, and d is the encryption of the pair (c, x) using randomness r". The ciphertext is $C = (c, d, \pi)$. To decrypt $C' = (c', d', \pi')$ using (x, z, pk, σ), one first checks the validity of the proof π'. If valid, output $\mathcal{D}_{x,\alpha}(c')$, else output $\perp$.

Theorem 3. *If $\mathcal{SE}$ is CPA-secure w.r.t. exponentially hard-to-invert auxiliary input, E_{pk} is a CCA-secure public-key encryption scheme, h is a UOWHF, and Π is a simulation-sound NIZK, then the above scheme $\mathcal{SE}'$ is a secure authentication encryption scheme (and hence CPA/CCA-secure) w.r.t. exponentially hard-to-invert auxiliary input. In particular, one can build such a scheme assuming the existence of trapdoor permutations and the LSN assumption.*

The theorem is formally proven in the full version [19]. Here, we only comment on the intuition. First, the zero-knowledge of π and the CPA-security of E_{pk} are used to argue that $\mathcal{SE}'$ is still CPA-secure (w.r.t. auxiliary input). As for ciphertext unforgeability, the non-malleability of E_{pk} and the simulation-soundness of π are used to argue that, despite seeing many "fake" public-key encryptions returned by the "simulated" encryption oracle (where we will use the trapdoor to the CRS), a forged ciphertext C^* contains a valid encryption d^* of some (c^*, x^*), where (among other things) x^* is such that $h(x^*) = h(x) = z$. By the universal one-wayness of h, this means that $x^* = x$. But then, using the decryption key for E, we can extract the correct x from such a valid decryption query C^*. And this contradicts the CPA-security of $\mathcal{SE}$.

4.3 Dealing with Adaptive Leakage

In Section 4 we presented CPA/CCA secure symmetric encryption schemes w.r.t. what we call *static* exponentially hard-to-invert auxiliary input. Namely, we assumed that the auxiliary input $f(x)$ was fixed before the attack started; i.e., that $f(x)$ is independent of the information given by the encryption and decryption oracles.

A more realistic scenario is the *adaptive* one, where the auxiliary information can be adaptively gathered. For example, the adversary may receive a bunch of ciphertexts, and then may try to obtain additional information about the secret-key that depends in some way on these ciphertexts.

The first question we address is: how do we define security in this setting? Since there may be dependencies between the auxiliary input and the information given by the encryption/decryption oracles, it is natural to provide a guarantee of the form: As long as the total information gathered by the adversary is such that it is exponentially hard to guess the secret-key then the scheme remains secure. However, this definition is meaningful only if the encryption function itself is exponentially hard-to-invert. The encryption schemes we present (in Section 4) are *not* exponentially hard-to-invert, since they are based on the LSN assumption (described in Section 2), and thus one can use the algorithm of Blum, Kalai, and Wasserman [7] to invert them in time $2^{O(n/\log n)}$. Nevertheless, we would like to guarantee security against adaptive leakage. Therefore, we require that at most βn bits are adaptively leaked (for any constant $\beta < 1$).

Namely, in the adaptive setting, we give the adversary oracle access to an arbitrary boolean function (that outputs only one bit), and we allow the adversary to query this function at most βn times (this function should be easy to compute given the secret key). We say that a scheme is CPA/CCA `secure w.r.t. adaptive leakage` if for *every* (easy to compute) boolean function $f(sk, \cdot)$ and every constant $\beta < 1$, no PPT adversary $\mathcal{A}$, who is given oracle access to $f(sk, \cdot)$ can win the CPA/CCA game, after querying the oracle $f(sk, \cdot)$ at most βn times.

More generally, we add the (static) auxiliary input into the adaptive setting, and say that a scheme is CPA/CCA `secure w.r.t. adaptive leakage` if for every constants $\alpha, \beta > 0$ such that $\alpha + \beta < 1$, the scheme remains secure given any (static) α-exponentially hard-to-invert auxiliary input $f(sk)$ (i.e., it is hard to invert f with probability $2^{-\alpha n}$), and given any (adaptive) leakage of size βn. Note that this adaptive model generalizes the static one.

In the next two theorems, whose proofs are deferred to the final version [19], we show that the encryption schemes presented earlier are also secure w.r.t. *adaptive* leakage.

Theorem 4. *Under the auxiliary-input LPN (or standard LSN) assumption, the scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ presented in Section 4.1 is CPA secure w.r.t.* `adaptive leakage`.

Theorem 5. *Under the auxiliary-input LPN (or standard LSN) assumption and the existence of a trapdoor permutations, the scheme $\mathcal{SE}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ presented in Section 4.2 is CCA secure w.r.t.* `adaptive leakage`.

5. AVERAGE-CASE OBFUSCATION WITH AUXILIARY INPUT

We next show how to use the CPA encryption scheme, defined in Section 4.1, to construct an average-case obfuscator w.r.t. exponentially hard-to-invert auxiliary input, for the family of *extended point functions* $\mathcal{I} = \{I_{x,m}\}$, where $I_{x,m}$ is the function that on input z outputs $\perp$, unless $z = x$ in which case it outputs a (possibly secret) message m.

In what follows, we give formal definition of *average-case obfuscation w.r.t. auxiliary input*. For simplicity, we only consider static auxiliary input. Moreover, our obfuscator depends on the "hardness" of the auxiliary input. Namely, to get security w.r.t. α-exponentially hard auxiliary input our obfuscator depends on this constant α.[5]

DEFINITION 5. *An* `average-case obfuscator w.r.t.` α-`exponentially hard-to-invert auxiliary input` *for a family of circuits* $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ *is an algorithm* $\mathcal{O}_\alpha$ *that satisfies the following:*

1. **Functionality**: *For every* $n \in \mathbb{N}$ *and every* $C \in \mathcal{C}_n$, $\Pr[\forall x,\ \mathcal{O}_\alpha(C)(x) = C(x)] \geq 1 - \mathrm{negl}(n)$, *where the probability is over the randomness of* $\mathcal{O}$.
2. **Polynomial slowdown**: *There exists a polynomial* p, *such that for every* $n \in \mathbb{N}$ *and every* $C \in \mathcal{C}_n$, $|\mathcal{O}_\alpha(C)| \leq p(|C|)$.
3. **Average-case black-box property w.r.t. auxiliary input**: *For every* PPT *adversary* $\mathcal{A}$ *there exists a* PPT *simulator* $\mathcal{S}$ *such that for every* α*-exponentially hard-to-invert function* f, *the following difference is* $\mathrm{negl}(n)$:
$$|\Pr[\mathcal{A}(\mathcal{O}_\alpha(C), f(C)) = 1] - \Pr[S^C(1^n, f(C)) = 1|,$$
where the probabilities are over $C \in_R \mathcal{C}_n$ *and over the random coin tosses of* $\mathcal{A}$ *and* $\mathcal{S}$.

We stress that our definition requires security only *on average*. Namely, instead of requiring that a PPT adversary cannot gain any information from a garbled circuit (beyond black-box access) *for every* function in the family, we require that this holds for a *random* function in the family. So, in that respect our definition is weaker than the standard definition. However, our definition allows the adversary to have auxiliary input (that depends on the function being obfuscated). We believe that for many cryptographic applications, average-case obfuscation suffices, whereas the lack of auxiliary input consideration may actually be problematic.

We notice that we do not know of any obfuscator (even for the class of point functions) that is secure with auxiliary input, except for the one due to Canetti [10], which was given in the context of perfect one-way function. We remark on the connection between point function obfuscation and perfect one-way functions at the end of this section.

OUR OBFUSCATOR. Our obfuscator $\mathcal{O}_\alpha$ uses any symmetric encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, that is *CPA secure w.r.t.* α*-exponentially hard-to-invert auxiliary input*, and has the *unique key property* and the *key hiding property*, as defined in Section 4.1 (Definitions 3 and 4, respectively). The actual construction is very simple. It takes as input a pair (m, x), and outputs a circuit C that has a value $c \leftarrow \mathcal{E}_{x,\alpha}(m)$ hardwired into it. The circuit C, on input z, outputs $\mathcal{D}_{z,\alpha}(c)$.

THEOREM 6. *For every constant* $\alpha > 0$, *if* $\mathcal{SE}$ *is* CPA secure w.r.t. α-exponentially hard-to-invert auxiliary input, *and has the* unique key property *and the* key hiding property, *then the above construction* $\mathcal{O}_\alpha$ *is an average-case obfuscator w.r.t.* α*-exponentially hard-to-invert auxiliary input, for the family of extended point functions* $\{I_{x,m}\}$.

COROLLARY 1. *Under the auxiliary-input LPN (or standard LSN) assumption, for every constant* $\alpha > 0$ *there is an average-case obfuscator* $\mathcal{O}_\alpha$ *w.r.t.* α*-exponentially hard-to-invert auxiliary input, for the family of extended point-functions* $\{I_{x,m}\}$.

We defer the formal proof of Theorem 6 to the final version [19]. However, we give a brief intuition. The correctness property follows from the correctness of the encryption and the *unique key property* of the underlying encryption scheme. The polynomial slowdown property follows from the fact that the operation of decrypting given a secret-key is a poly-time operation. The average-case black-box property w.r.t. auxiliary input follows from the security of the underlying CPA encryption scheme w.r.t. auxiliary input, and from the *key hiding property*.

REMARK. One can think of this obfuscator as a *perfect one-way function* [10, 14]. Loosely speaking, a perfect one-way function is a function that hides all partial information about its input. It is formalized as a probabilistic function, and it is required to be verifiable. Namely, a perfect one-way hash function is associated with a pair of functions (H, V), and it is required that for every $x \in \{0,1\}^n$,
$$\Pr_R[V(x, H(x, R)) = 1] = 1.$$

Under the auxiliary-input LPN (or standard LSN) assumption, for every constant $\alpha > 0$ we construct a perfect one-way function with the guarantee that for every α-exponentially hard-to-invert function f, and for every polynomial t,
$$\left\{f(x), \{H(x, R_i)\}_{i=1}^t\right\} \approx \left\{f(x), \{H(x_i, R_i)\}_{i=1}^t\right\} \qquad (6)$$
where the probabilities are over randomly and independently chosen $x, x_1, \ldots, x_t, R_1, \ldots, R_t$. We note that the Canetti [10] constructs a perfect one-way function w.r.t. auxiliary input, however his assumption quantifies over all auxiliary inputs. Other known constructions [14] do not deal with auxiliary input. Moreover, they can only handle x that is uniformly distributed, or in the CRS model they can handle x that has min-entropy, though this min-entropy must be *independent* of the CRS.

However, we do point out that the constructions in previous work were collision resistant; i.e., it was guaranteed to be hard to generate a string c and two distinct strings x, x' such that $V(x, c) = V(x', c) = 1$. Our function only has the guarantee that with overwhelming probability (over R), given $c = H(x, R)$, there does not exist $x' \neq x$ such that $V(c, x) = V(c, x') = 1$. (Loosely speaking, we guarantee target collision resistance, whereas previous work guaranteed collision resistance). However, this form of collision resistance seems to be sufficient for many applications.

[5] We note that if we strengthen the LSN assumption, and require that the polynomial $p = p_\alpha$ does not depend on α then we get a single obfuscator that is secure w.r.t. *any* exponentially hard-to-invert auxiliary input.

6. REUSABLE AND ROBUST EXTRACTORS

Due to lack or space, the details of these applications are deferred to the full version [19]. In brief, the reusable extractor will use our CPA encryption, while the reusable and robust extractor will use the CCA encryption (in the CRS model, but allowing the auxiliary function to depend on the CRS). We also remark that, prior to this work, existing reusable extractors were either based on assumptions which are not efficiently falifiable [10], or required exponential assumptions and min-entropy rate above 1/2 [48].

7. REFERENCES

[1] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous Hardcore Bits and Cryptography Against Memory Attacks. In *TCC 2009: 474-495.*

[2] B. Applebaum. Fast Cryptographic Primitives Based on the Hardness of Decoding Random Linear Code. *Unpublished Manuscript.*

[3] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (Im)possibility of Obfuscating Programs. In *CRYPTO 2001: 1-18.*

[4] B. Barak, R. Shaltiel, and A. Wigderson. Computational analogues of entropy . In *Proc. of 7th Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM), 2003.*

[5] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm In *Advances in Cryptology – ASIACRYPT*, volume 1976 of Lecture Notes in Computer Science, pp. 531–545, 2000.

[6] A. Blum, M. L. Furst, M. J. Kearns, and R. J. Lipton. Cryptographic Primitives Based on Hard Learning Problems. In *CRYPTO 1993: 278-291.*

[7] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. In *STOC 2000: 435-440.*

[8] X. Boyen. Reusable cryptographic fuzzy extractors. In *ACM Conference on Computer and Communications Security 2004: 82-91.*

[9] X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky, A. Smith. Secure Remote Authentication Using Biometric Data. In *EUROCRYPT*, pp. 147–163, 2005.

[10] R. Canetti. Towards Realizing Random Oracles: Hash Functions That Hide All Partial Information. In *CRYPTO 1997: 455-469.*

[11] R. Canetti, R. R. Dakdouk. Obfuscating Point Functions with Multibit Output. In *EUROCRYPT 2008: 489-508.*

[12] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai. Exposure-Resilient Functions and All-or-Nothing Transforms. In *EUROCRYPT 2000: 453-469.*

[13] R. Canetti, D. Eiger, S. Goldwasser, and D. Lim. How to Protect Yourself without Perfect Shredding. In *ICALP (2) 2008: 511-523.*

[14] R. Canetti, D. Micciancio, and O. Reingold. Perfectly One-Way Probabilistic Hash Functions (Preliminary Version). In *STOC 1998: 131-140.*

[15] D. Cash, Y. Z. Ding, Y. Dodis, W. Lee, R. J. Lipton, and S. Walfish. Intrusion-Resilient Key Exchange in the Bounded Retrieval Model. In *TCC 2007: 479-498.*

[16] J. Coron, M. Joye, D. Naccache, and P. Paillier. Universal Padding Schemes for RSA. In *CRYPTO 2002: 226-241.*

[17] G. Di Crescenzo, R. Lipton and S. Walfish. Perfectly Secure Password Protocols in the Bounded Retrieval Model. In *Theory of Cryptography Conference*, pp. 225–244, 2006.

[18] Y. Dodis. PhD Thesis, "Exposure-Resilient Cryptography". Massachusetts Institute of Technology, August 2000.

[19] Y. Dodis, Y. Kalai, and S. Lovett. On Cryptiography with Auxiliary Input. Preliminary version in *STOC 2009.* Full Version.

[20] Y. Dodis, J. Katz, L. Reyzin, and A. Smith. Robust Fuzzy Extractors and Authenticated Key Agreement from Close Secrets. In *CRYPTO 2006: 232-250.*

[21] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In *SIAM Journal of Computing*, 38(1):97–139, 2008.

[22] Y. Dodis, L. Reyzin, A. Smith. Fuzzy Extractors. In *Security with Noisy Data* (edited by Pim Tuyls, Boris Skoric, and Tom Kevenaar), Springer, 2007.

[23] Y. Dodis, A. Smith. Correcting Errors Without Leaking Partial Information In STOC 2005: 654-663.

[24] S. Dziembowski. On Forward-Secure Storage. In *CRYPTO 2006: 251-270.*

[25] S. Dziembowski. Intrusion-Resilience Via the Bounded-Storage Model. In *TCC 2006: 207-224.*

[26] S. Dziembowski and K. Pietrzak. Intrusion-Resilient Secret Sharing. In *FOCS 2007: 227-237.*

[27] S. Dziembowski and K. Pietrzak. Leakage-Resilient Cryptography. In *FOCS 2008: 293–302.*

[28] G. Forney. Concatenated Codes. MIT Press, Cambridge, MA, 1966.

[29] S. Goldwasser and Y. T. Kalai. On the Impossibility of Obfuscation with Auxiliary Input. In *FOCS 2005: 553-562.*

[30] O. Goldreich and L. A. Levin. A Hard-Core Predicate for all One-Way Functions. In *STOC 1989: 25-32.*

[31] S. Goldwasser and G. N. Rothblum. On Best-Possible Obfuscation. In *TCC 2007: 194-213.*

[32] S. Haber and B. Pinkas. Securely combining public-key cryptosystems. In *ACM Conference on Computer and Communications Security 2001:215-224.*

[33] S. Hada. Zero-Knowledge and Code Obfuscation. In *ASIACRYPT 2000: 443-457.*

[34] D. Hofheinz, J. Malone-Lee and M. Stam. Obfuscation for Cryptographic Purposes. In *TCC 2007: 214-232.*

[35] S. Hohenberger, G. N. Rothblum, A. Shelat, and V. Vaikuntanathan. Securely Obfuscating Re-encryption. In *TCC 2007: 233-252.*

[36] N. J. Hopper, Manuel Blum. Secure Human Identification Protocols. In *ASIACRYPT 2001:52-66.*

[37] C. Hsiao, C. Lu, and L. Reyzin. Conditional Computational Entropy, or Toward Separating Pseudoentropy from Compressibility. In *EUROCRYPT 2007: 169-186.*

[38] Y. Ishai, M. Prabhakaran, A. Sahai, and D. Wagner. Private Circuits II: Keeping Secrets in Tamperable Circuits. In *EUROCRYPT 2006: 308-327.*

[39] Y. Ishai, A. Sahai, and D. Wagner. Private Circuits: Securing Hardware against Probing Attacks. In *CRYPTO 2003: 463-481.*

[40] A. Juels and S. A. Weis. Authenticating Pervasive Devices with Human Protocols. In *CRYPTO 2005:293-308.*

[41] J. Katz and J. S. Shin. Parallel and Concurrent Security of the HB and HB+ Protocols. In *EUROCRYPT 2006: 73-87.*

[42] B. Lynn, M. Prabhakaran, and A. Sahai. Positive Results and Techniques for Obfuscation. In *EUROCRYPT 2004: 20-39.*

[43] S. Micali and L. Reyzin. Physically Observable Cryptography. In *TCC 2004: 278-296.*

[44] M. Naor. On Cryptographic Assumptions and Challenges. In *CRYPTO 2003: 96-109.*

[45] M. Naor and M. Yung. Universal One-Way Hash Functions and their Cryptographic Applications. In *STOC 1989: 33-43.*

[46] M. Naor and M. Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *STOC 1990: 427-437.*

[47] N. Nisan and D. Zuckerman. More deterministic simulation in logspace. In *STOC*, pp. 235–244, 1993.

[48] K. Pietrzak. A Leakage-Resilient Mode of Operation. In *EUROCRYPT 2009*.

[49] R. Raz. Private communication.

[50] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC 2005: 84-93.*

[51] A. Sahai. Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security. In *FOCS*, pp. 543–553, 1999.

[52] A.Shamir. Identity-Based Cryptosystems and Signature Schemes. In *CRYPTO 1984: 47-53.*

[53] H. Wee. On obfuscating point functions. In *STOC*, pp. 523–532.

[54] D. Zuckerman. Private communication.