Exposure-Resilient Functions and All-Or-Nothing Transforms

Ran Canetti* Yevgeniy Dodis† Shai Halevi* Eyal Kushilevitz‡ Amit Sahai†

Abstract

In this work, we study the problem of *partial key exposure*. Standard cryptographic definitions and constructions do not guarantee any security even if a tiny fraction of the secret key is compromised. We show how to build cryptographic primitives, in the standard model (without random oracles), that remain secure even when an adversary is able to learn *almost all* of the secret key. We accomplish this by giving constructions for the All-Or-Nothing Transform (**AONT**), introduced by Rivest. An **AONT** is an efficiently computable transform T on strings such that:

- For any string x, given *all* of $T(x)$, one can efficiently recover x.
- There exists some threshold ℓ such that any polynomial-time adversary that (adaptively) learns all but ℓ bits of $T(x)$ obtains no information about x (in a computational sense).

By applying an **AONT** to the secret key of any cryptographic system, we can obtain security against partial key exposure. The only previous construction of an **AONT** with provable security was based on random oracles.

The key to our approach is a new notion, which may be of independent interest, which we call an *Exposure-Resilient Function* (**ERF**) — a deterministic function whose output appears random even if *almost all* the bits of the input are known. We show how to construct **ERF**'s and **AONT**'s with nearly optimal parameters from any one-way function. We also obtain several related results about these notions.

*IBM T.J. Watson Research Center, P.O. Box 704, Yorktown Heights, New York 10598, USA. Email: {canetti,shaih}@watson.ibm.com.

†Lab. of Computer Science, Massachusetts Institute of Technology, 545 Tech Square, Cambridge, MA 02139, USA. Email: {yevgen,amits}@theory.lcs.mit.edu.

‡Department of Computer Science, Technion, Haifa 32000, Israel. Email: eyalk@cs.technion.ac.il.

1 Introduction

A great deal of cryptography can be seen as finding ways to leverage the possession of a small but totally secret piece of knowledge (a key) into the ability to perform many useful and complex actions: from encryption and decryption to identification and message authentication. But what happens if our most *basic* assumption breaks down — that is, if the secrecy of our key becomes partially compromised?

It has been noted that key exposure is one of the greatest threats to security in practice [1]. Indeed, at the recent Rump session of CRYPTO '98, Nicko van Someren [25] illustrated a breathtakingly simple attack by which keys stored in the memory of a computer could be identified and extracted, by looking for regions of memory showing high entropy. Within weeks of the appearance of the followup paper [24], a new generation of computer viruses emerged that tried to use these ideas to steal secret keys [9]. Shamir and van Someren gave some heuristic suggestions on preventing these kinds of attacks, such as having software "spread a key among different memory locations" in order to avoid being found. While such measures help to ensure that attackers will not recover the entire secret key, they do not solve the problem of *partial exposure.*

Unfortunately, standard cryptographic definitions and constructions cannot guarantee security *even if a tiny fraction of the secret key is exposed*. In this work, we show how to build cryptographic primitives, in the standard model (without random oracles) and using general computational assumptions, that remain provably secure even when the adversary is able to learn *almost all* of the secret key. Our techniques also have several applications in other settings.

Previous approaches and our goals. The most widely considered solutions to the problem of key exposure are distribution of keys across multiple servers via secret sharing [23, 4], and protection using specialized hardware. Instantiations of the key distribution paradigm include threshold cryptosystems [8] and proactive cryptosystems [14]. Distribution across many systems, however, is quite costly. Such an option may be available to large organizations, but is not realistic for the average user. Another widely considered proposal is the use of specially protected hardware such as smartcards, which can also be costly, inconvenient, or inapplicable to many contexts. Thus, the cost or inconvenience of such solutions may make them prohibitive for many applications; some users simply may not have the luxury to afford the investment such solutions would require.

Instead, we seek to enable a single user to protect itself against partial key exposure on a single machine. A natural idea would be to use a secret sharing scheme to split the key into shares, and then attempt to provide protection by storing these shares instead of storing the secret key directly. However, secret sharing schemes only guarantee security if the adversary misses at least one share *in its entirety*. Unfortunately, each share must be fairly large (about as long as the security parameter). Thus, in essence *we return to our original problem*: even if an adversary only learns a small fraction of all the bits, it could be that it learns a few bits from *each* of the shares, and hence the safety of the secret can no longer be guaranteed. We would like to do better[1].

The All-Or-Nothing Transform. Recently Rivest [22], motivated by different security concerns arising in the context of block ciphers, introduced an intriguing primitive called the *All-Or-Nothing Transform (*__AONT__*)*. An **AONT**[2] is an efficiently computable transformation T on strings such that:

- For any string x, given *all* of $T(x)$, one can efficiently recover x.

[1]Indeed, our techniques can be seen as yielding, for certain parameters, highly efficient "gap" analogues of computational secret sharing schemes [17], where the share size can be small as 1 *bit*! See Remark 5.5.

[2]Here we informally present a refinement of the definition due to Boyko [5].

- There exists some threshold ℓ such that any polynomial-time adversary that (adaptively) learns all but ℓ bits of $T(x)$ obtains *no* information about x (in a computational sense).

The **AONT** solves the problem of partial key exposure: Rather than storing a secret key directly, we store the **AONT** applied to the secret key. If we can build an **AONT** where the threshold value ℓ is very small compared to the size of the output of the **AONT**, we obtain security against almost total exposure. Notice that this methodology applies to secret keys with arbitrary structure, and thus protects all kinds of cryptographic systems. One can also consider **AONT**'s that have a two-part output: a public output that doesn't need to be protected, and a secret output that has the exposure-resilience property stated above. Such a notion would also provide the kind of protection we seek to achieve. The **AONT** has many other applications, as well, such as enhancing the security of block-ciphers and making fixed-blocksize encryption schemes more efficient [16]. For an excellent exposition on these and other applications of the **AONT**, see [5].

Our Results: Until now, the only known construction of an **AONT**[3] with provable security was given by Boyko [5] in the random oracle model, who showed that Bellare and Rogaway's Optimal Asymmetric Encryption Padding (OAEP) [2] yields an **AONT**. In this work, we give the first constructions for **AONT**'s with essentially optimal resilience in the standard model, based only on computational assumptions.

The key to our approach and our main conceptual contribution is the notion of an *Exposure-Resilient Function* (**ERF**) — a deterministic function whose output appears random even if *almost all* the bits of the input are revealed. We believe this notion is both very useful and interesting in its own right. Consider for example an **ERF** with an output that is longer than its input — this can be seen a particularly strong kind of pseudorandom generator, where the generator's output remains pseudorandom even if most of the seed is known. We show that **ERF**'s provide a solution to the partial key exposure problem for many settings in private-key cryptography, where the secret key need only be a pseudorandom string.

More specifically, our results are:

- We show how to construct, from any one-way function, for any $\epsilon > 0$, an **ERF** mapping an input of n bits to an output of *any size* polynomial in n, such that as long as *any* n^ϵ bits of the input remain unknown, the output will be pseudorandom. We give examples of how to use **ERF**'s directly to address key exposure problems in private key cryptography; most notably we show how to solve what we call the *gradual key exposure* problem, where an adversary is able to learn more and more bits of a shared secret key over time.

- We give a simple construction of an **AONT** based on any **ERF**. For any $\epsilon > 0$, we show how to achieve a resilience threshold of $\ell = N^\epsilon$, where N is the size of the output of the **AONT**. If viewed as an **AONT** with separate public and secret outputs, then the size of the output of the **AONT** can be made optimal, as well.

- We also show that the existence of an **AONT** with $\ell < k - 1$, where k is the size of the input, implies the existence of one-way functions. We show that this result is tight up to a constant factor by constructing an unconditionally secure **AONT** with $\ell = \Theta(k)$ using no assumptions.

- We also give another construction of an **AONT** based on any function f such that both $[x \mapsto f(x)]$ and $[x \mapsto f(x) \oplus x]$ are **ERF**'s. This construction is similar to the OAEP, and so our analysis makes a step towards abstracting the properties of the random oracle needed to make the OAEP work as an

[3]Though for a much weaker definition of security than the one we study here, Stinson [27] has given a simple elegant construction for **AONT** with provable security in the unconditional setting. As observed by [5], however, this construction does not achieve the kind of security considered here.

AONT. It also has the advantage of meeting the standard definition of an **AONT** (without separate public and secret outputs) while retaining a relatively short output length.

- Finally, we show that a weaker "average-case" definition of **AONT** is equivalent to the standard "worst-case" definition of **AONT**, by giving an efficient transformation that achieves this goal.

Previous Work: Chor et al. [7] considered a notion called a t-resilient function, which are related to our notion of an Exposure-Resilient Function (**ERF**). A t-resilient function is a function whose output is *truly* random even if an adversary can fix any t of the inputs to the function. This turns out to be equivalent to the strongest formulation of unconditional security for an **ERF**. We give constructions for statistical unconditionally secure **ERF**'s that beat the impossibility results given in [7], by achieving an output distribution that is not truly random, but rather exponentially close in statistical deviation from truly random. Work on privacy amplification in unconditionally secure key agreement protocols is also related to our work (see e.g. [3, 6]).

Bellare and Miner [1] consider the notion of forward-security for signature schemes, which is a different attempt to address the key exposure problem. The kind of security they achieve prevents an adversary that gains a current secret key from being able to forge signatures on messages "dated" in the past. A similar notion of security can be defined for encryption, where a compromised current secret key would not enable an adversary to decrypt messages sent in the past. In contrast, our work deals with providing security for both the future as well as the past, but assuming that not *all* of the secret key is compromised. In Section 4.4, we also address the problem of gradual key exposure, where no *a priori* bound on the amount of information the adversary obtains is assumed, rather we assume only a bound on the *rate* at which that the adversary gains information.

Organization: In Section 2 we briefly define some preliminaries. Section 3 defines our main notions of Exposure-Resilient Functions and All-Or-Nothing Transforms. Section 4 talks in detail about constructions and application of **ERF**'s, while Section 5 is concentrated with constructing and examining the properties of **AONT**'s. Due to space limitations, some of the proofs and discussion are left to Appendices.

2 Preliminaries

For a randomized algorithm F and an input x, we denote by $F(x)$ the output distribution of F on x, and by $F(x;r)$ we denote the output string when using the randomness r. If one of the inputs to F is considered a "key", then we write it as a subscript (e.g., $F_s(x)$). In this paper we will not optimize certain constant factors which are not of conceptual importance. Unless otherwise specified, we will consider security against nonuniform adversaries. Note that all the proofs of security can be made to work with uniform adversaries as well, with appropriate standard modifications to the definitions and proofs.

Let $\{{n \atop \ell}\}$ denote the set of ℓ element subsets of $[n] = \{1 \dots n\}$, and for $L \in \{{n \atop \ell}\}$, $y \in \{0,1\}^n$, let $[y]_L$ denote y restricted to its $(n-\ell)$ bits *not* in L. We denote by $\oplus$ the bit-wise exclusive OR operator.

We recall that that *statistical difference* (also called *statistical distance*) between two random variables X and Y on a finite set D is defined to be

$$\max_{S \subset D} \Big| \Pr[X \in S] - \Pr[Y \in S] \Big| = \frac{1}{2} \cdot \sum_{\alpha} \Big| \Pr[X = \alpha] - \Pr[Y = \alpha] \Big|.$$

2.1 Indistinguishability

Given two distributions A and B, we denote by $A \cong_c B$ ($A \cong_\epsilon B$, $A \equiv B$) the fact that they are computationally (statistically within ϵ, perfectly) indistinguishable. For the case of statistical closeness, we will always have that ϵ is negligible in an appropriate security parameter. When the statement can hold for any of the above choices (or the choice is clear from the context), we will simply write $A \approx B$.

We need the following lemma whose proof can be found in Appendix A.

Lemma 2.1 *Let A and B be any two random variables. Let R be chosen uniformly at random and C be chosen according to a distribution p, both independently from A and B. Then the following are equivalent:*

(1) $\langle A, B \rangle \approx \langle A, R \rangle$.

(2) $\langle A, B, C \rangle \approx \langle A, B \oplus C, C \rangle$, *for any polynomial time sampleable (PTS) distribution p.*

(3) $\langle A, B, C \rangle \approx \langle A, B \oplus C, C \rangle$, *for uniform p.*

The proof of the next simple lemma is straightforward and is omitted.

Lemma 2.2 *Let g be polynomial time computable function and X, X', Y, Y' be random variables. Then*

a) $\langle X, Y \rangle \approx \langle X', Y' \rangle \Longleftrightarrow \langle X, Y \oplus g(X) \rangle \approx \langle X', Y' \oplus g(X') \rangle$.

b) assume X is independent from both Y and Y', and h is some function. Then

$$\langle X,\ h(Y, g(X)) \rangle \approx \langle X,\ h(Y', g(X)) \rangle \Longleftrightarrow \langle g(X),\ h(Y, g(X)) \rangle \approx \langle g(X),\ h(Y', g(X)) \rangle$$

We call $g(X)$ sufficient statistics.

3 Definitions

In this section, we define the central concepts in our paper: Exposure-Resilient Functions (**ERF**'s) and All-Or-Nothing Transforms (**AONT**'s). An **ERF** is a function such that if its input is chosen at random, and an adversary learns *all* but ℓ bits of the input, for some threshold value ℓ, then the output of the function will still appear pseudorandom to the adversary. Formally,

Definition 3.1 *A (deterministic) polynomial time computable function $f(x) : \{0,1\}^n \to \{0,1\}^k$, is ℓ-***ERF** (exposure-resilient function) *if for any $L \in \{ {n \atop \ell} \}$ and for a randomly chosen $r \in \{0,1\}^n$, $R \in \{0,1\}^k$, the following distributions are indistinguishable:*

$$\langle [r]_L, f(r) \rangle \approx \langle [r]_L, R \rangle \tag{1}$$

Here $\approx$ can refer to perfect, statistical or computational indistinguishability.

Remark 3.2 *Note that this definition is a "non-adaptive" version of the definition. One may also consider an adaptive version of the definition, where the adversary may adaptively choose one-bit-at-a-time which $n - \ell$ positions of the input to examine. Owing only to the messiness of such a definition, we do not give a formal definition here, but we stress that all our constructions* satisfy this adaptive definition*, as well.*

The definition states that an **ERF** transforms n random bits into k (pseudo-)random bits, such that even learning all but ℓ bits of the input leaves the output indistinguishable from a random value. There are several parameters of interest here: ℓ, n, and k. We see that the smaller ℓ is, the harder is to satisfy the condition above, since fewer bits are left unknown to the adversary. However, ℓ is not the only parameter of interest, it is both n and ℓ that tell us how "exposure-resilient" is the **ERF** for a given k. For example, saying that $\ell = k^\epsilon$ does not mean much on its own. It could be that $n = \ell$, and the function in this case has no exposure-resilience properties. Generally, there are two measures of interest: the fraction of ℓ with respect to n, which we would like to be as small as possible (this shows the "resilience"); and the size of k with respect to ℓ, which we want to be as large as possible (this shows how many pseudorandom bits we obtain compared to the number of random bits the adversary cannot see).

We now define the notion of an **AONT**:

Definition 3.3 *A randomized polynomial time computable function $T(x) : \{0,1\}^k \to \{0,1\}^N$ is ℓ-* **AONT** *(*all-or-nothing transform*) if*

1. *T is efficiently invertible, i.e. there is a polynomial time machine I such that for any $x \in \{0,1\}^k$ and any $y \in T(x)$, we have $I(y) = x$.*
2. *For any $L \in \{{N \atop \ell}\}$, any $x_0, x_1 \in \{0,1\}^k$ we have*
$$\langle x_0, x_1,\ [T(x_0)]_L\rangle \approx \langle x_0, x_1,\ [T(x_1)]_L\rangle \tag{2}$$
In other words, the random variables in $\{[T(x)]_L \mid x \in \{0,1\}^k\}$ are all indistinguishable from each other. Here $\approx$ can refer to perfect, statistical or computational indistinguishability.

Remark 3.4 *Note again, as in Remark 3.2, that the definition given here is a "non-adaptive" definition. We stress that all our constructions satisfy the corresponding adaptive definition, as well.*

The definition given above is the natural analogue of the formal definition of **AONT** given by Boyko [5] (refining an earlier definition of Rivest [22]) in a setting with a random oracle[4]. We also consider a generalization of this notion, which we call an **AONT** *with secret and public outputs*. In this case, we consider an **AONT** where the output y is divided in two sections: a secret part y_1 and a public part y_2. The public part of the output is such that it requires *no* protection – that is, it can revealed to the adversary in full. It is only secret part y_1 that requires some protection. The security guarantee now states that as long as ℓ bits of the secret output y_1 remain hidden (while all the bits of y_2 can be revealed), the adversary should have no information about the message. Note that clearly, this generalized notion solves the problem of partial key exposure as well (and also remains equally applicable to all the other known uses of the **AONT**). This generalized form allows us to characterize the security of our constructions more precisely. For a more detailed discussion of this notion, see Appendix B.

Formally, the definition of ℓ-**AONT** remains as above with the following simple modification: Now we have $N = s + p$, and $T(x)$ outputs a pair $y = (y_1, y_2)$ where $y_1 \in \{0,1\}^s$ and $y_2 \in \{0,1\}^p$. Finally, the security holds for all $L \in \{{s \atop \ell}\}$ rather than $L \in \{{N \atop \ell}\}$ (observe that notationally, $[y]_L = ([y_1]_L, y_2)$). Everything else remains the same. The standard definition corresponds to the case where the public output is of size 0 (i.e. $p = 0$, $s = N$). We call such **AONT**'s *secret-only*.

The above definition is "indistinguishability" based. As usual, one can make the equivalent "semantic security" based definition, where the adversary, given $z = [T(x)]_L$ (where x is picked according to some distribution M), cannot compute β satisfying some relation $\mathcal{R}(x, \beta)$ "significantly better" than without z

[4]Boyko's definition looks somewhat more complicated on the surface, since he allows the adversary to choose, say, x_0 and x_1 based on the random oracle. In our case, there are no random oracles, so the definition simplifies.

at all. The proof of equivalence is standard and is omitted. Thus, the all-or-nothing transforms allow one to "encode" any x in such a form that the encoding is easily invertible, and yet, an adversary learning all but ℓ bits of the (secret part of the) encoding "cannot extract any useful" information about x.

Boyko [5] showed that assuming the existence of a random oracle, the following so called "optimal asymmetric encryption padding" (OAEP) construction is an ℓ-**AONT** (where ℓ can be chosen to be logarithmic in the security parameter). Let $G : \{0,1\}^n \to \{0,1\}^k$ and $H : \{0,1\}^k \to \{0,1\}^n$ be random oracles (where n is any number greater than ℓ). The randomness of T is $r \leftarrow \{0,1\}^n$. Define $T(x) = \langle u, t\rangle$, where

$$u = G(r) \oplus x \tag{3}$$
$$t = H(u) \oplus r \tag{4}$$

We note that $I(u,t) = G(H(u) \oplus t) \oplus u$. No constructions of **AONT** based on standard assumptions were previously known.

Remark 3.5 *The notions of* **ERF** *and* **AONT** *are closely related with the following crucial difference. In an* **ERF***, the "secret" is a (pseudo-)random value $f(r)$.* **ERF** *allows one to represent this random secret in an "exposure-resilient" way by storing r instead. In* **AONT***, the secret is an* arbitrary *x, which can be represented in an "exposure-resilient" way by storing $T(x)$ instead. Thus,* **ERF** *allows one to represent a random secret in an exposure-resilient way, while* **AONT** *allows this for any secret.*

4 Exposure-Resilient Functions (ERF)

In this section we give constructions and some applications of exposure-resilient functions (**ERF**'s). First, we describe perfect **ERF**'s and their limitations. Then, on our way to building computational **ERF**'s with very strong parameters, we first build statistical **ERF**'s, and achieve essentially the best possible parameters. Finally we show how to combine this construction with standard pseudorandom generators (**PRG**) to construct computational **ERF**'s based on any one-way function (**OWF**) that achieve any $\ell = \Omega(n^\epsilon)$ and any $k = poly(n)$ (in fact, such **ERF**'s are equivalent to the existence of one-way functions). We conclude by giving several applications of **ERF**'s.

4.1 Perfect ERF

Here we require that $\langle [r]_L, f(r)\rangle \equiv \langle [r]_L, R\rangle$. Since the distributions are identical, this is equivalent to saying that no matter how one sets any $(n-\ell)$ bits of r (i.e. sets $[r]_L$), as long as the remaining r bits are set at random, the output $f(r)$ is still perfectly uniform over $\{0,1\}^k$. This turns out to be exactly the notion of so called $(n-\ell)$-*resilient* functions considered in [7]. As an example, if $k = 1$, exclusive OR of n input bits is a trivial perfect 1-**ERF** (or a $(n-1)$-resilient function).

We observe that perfect ℓ-**ERF** can potentially exist only for $\ell \geq k$. Optimistically, we might expect to indeed achieve $\ell = O(k)$. However, already for $k = 2$ Chor et al [7] show that we must have $\ell \geq n/3$, i.e. at least third of the input should remain secret in order to get just 2 random bits! On the positive side, for $\ell > n/2$, using *binary linear error correcting codes* one can construct perfect ℓ-**ERF**.For a sketch of the proof of the following theorem and discussion of its implications (along with some background on error correcting codes), see Appendix C.

Theorem 4.1 ([7]) *Let M be a $k \times n$ matrix. Define $f(r) = M \cdot r$, where $r \in \{0,1\}^n$. Then f is perfect ℓ-***ERF** *if and only if M is the generator matrix for a code of distance $n - \ell + 1$.*

4.2 Statistical ERF

We saw that perfect **ERF** cannot achieve $\ell < n/3$. Breaking this barrier will be crucial in achieving the level of security we ultimately desire from (computational) **ERF**'s. In this section, we show that by relaxing the requirement only slightly to allow negligible (in fact *exponentially small*) statistical deviation, we are able to obtain **ERF**'s for essentially any value of ℓ (with respect to n) such that we obtain an output size $k = \Theta(\ell)$. Note that this is the best we can hope for up to constant factors, since it is not possible to have $k > \ell$ for any **ERF** with statistical deviation $\epsilon < \frac{1}{2}$ (proof is obvious, and omitted).

The key ingredient in our construction will be a combinatorial object called a *strong extractor*. An *extractor* is a family of hash functions $\mathcal{H}$ such that when a function h is chosen at random from $\mathcal{H}$, and is applied to a random variable X that has "enough randomness" in it, the resulting random variable $Y = h(X)$ is statistically close to the uniform distribution. In other words, by investing enough true randomness, namely the amount needed to select a random member of $\mathcal{H}$, one can "extract" from X a distribution statistically close to the uniform distribution. A *strong* extractor has an extra property that Y is close to the uniform distribution even when the random function h is revealed. (Perhaps the best known example of a strong extractor is given in the Leftover Hash Lemma of [15], where standard 2-universal hash families are shown to be strong extractors.) Much work has been done in developing this area (e.g. [12, 26, 28, 21]). In particular, it turns out that one can extract almost all the randomness in X by investing very few truly random bits (i.e. having small $\mathcal{H}$). For more information on these topics, see the excellent survey article of Nisan [20].

The intuition behind our construction is as follows: Notice that after the adversary observes $(n - \ell)$ bits of the input (no matter how it chose those bits), the input can still be any of the 2^ℓ completions of the input with equal probability. In other words, conditioned on any observation made by the adversary, the probability of any particular string being the input is at most $2^{-\ell}$. Thus, if we apply a sufficiently good extractor to the input, we have a chance to extract $\Omega(\ell)$ bits statistically close to uniform — exactly what we need. The problem is that we need some small amount of true randomness to select the hash function in the extractor family. However, if this randomness is small enough (say, at most $\ell/2$ bits), we can take it from the input itself. Hence, we view the first $\ell/2$ bits of r (which we will call u) as the randomness used to select the hash function h, and the rest of r we call v. The output of our function will be $h(v)$. Then even observing u and $(n - \ell)$ other bits of r leaves at least $2^{\ell/2}$ equally likely possible values of r (since $|u| = \ell/2$). Now, provided our extractor is good enough, we indeed obtain $\Omega(\ell)$ bits statistically close to uniform.

A few important remarks are in place before we give precise parameters. First, the adversary may choose to learn the entire u (i.e. it knows h). This is not a problem since we are using a *strong* extractor, *i.e.* the output is random even if one knows the true randomness used. Secondly, unlike the perfect **ERF** setting, where it was equivalent to let the adversary set $(n - \ell)$ input bits in any manner it wants, here the entire input (including u) *must* be chosen uniformly at random (and then possibly observed by the adversary).

Our most important requirement is that the hash function in the strong extractor be describable by a very short random string. This requirement is met by the strong extractor of Srinivasan and Zuckerman [26] using the hash families of Naor and Naor [19]. Their results can be interpreted as giving the following lemma:

Lemma 4.2 ([26]) *For any ℓ and $t < \ell/2$, there exists a family $\mathcal{H}$ of hash functions mapping $\{0,1\}^n$ to a range $\{0,1\}^k$, where $k = \ell - 2t$, such that the following holds: A random member of $\mathcal{H}$ can be described by and efficiently computed using $4(\ell - t) + O(\log n)$ truly random bits (we will identify the hash function h with these random bits). Furthermore, for any distribution X on $\{0,1\}^n$ such that $\Pr[X = x] \le 2^{-\ell}$ for all $x \in \{0,1\}^n$, we have that the statistical difference between the following two distributions is at* *most* $\epsilon = 2 \cdot 2^{-t}$*:*

(A) Choose h *uniformly from* $\mathcal{H}$ *and* x *according to* X*. Output* $\langle h, h(x)\rangle$*.*

(B) Choose h *uniformly from* $\mathcal{H}$ *and* y *uniformly from* $\{0,1\}^k$*. Output* $\langle h, y\rangle$*.*

We are now ready to describe our statistical construction.

Theorem 4.3 *There exist statistical* ℓ-**ERF** $f : \{0,1\}^n \to \{0,1\}^k$ *with* $k = O(\ell)$ *and statistical deviation* $2^{-\Omega(\ell)}$*, for any* k *and* n *satisfying* $\omega(\log n) \le k \le n$*.*

Proof: Note that we will not optimize constant factors in this proof. Let $\ell' = \ell/5$ and $t = \ell/20$. We let the output size of our **ERF** be $k = \ell' - 2t = \ell/10$ and the statistical deviation be $\epsilon = 2 \cdot 2^{-t} = 2^{-\Omega(\ell)}$. Suppose the (random) input to our function is r. Now, we will consider the first $d = 4(\ell' - t) + O(\log n) < 4\ell/5$ bits of r to be h, which describes some hash function in $\mathcal{H}$ mapping $\{0,1\}^n$ to $\{0,1\}^k$ as given in Lemma 4.2. Let r' be r with the first d bits replaced by 0's. Note that r' is independent of h, and the length of r' is n. Define $f(r) = h(r')$.

We now analyze this function. Observe that for any $L \in \{{n \atop \ell}\}$, conditioned on the values of both $[r]_L$ and h, there are still at least $\ell/5$ bit positions (among the last $n-d$ bit positions) of r that are unspecified. Hence, for all $L \in \{{n \atop \ell}\}$, for all $z \in \{0,1\}^{n-\ell}$, and for all $y \in \{0,1\}^n$, we have that

$$\Pr_r\left[r' = y \;\middle|\; L,\ [r]_L = z\right] \le 2^{-\ell/5} = 2^{-\ell'}.$$

Thus, by Lemma 4.2, we have that $\langle [r]_L, h, f(r)\rangle = \langle [r]_L, h, h(r')\rangle \cong_\epsilon \langle [r]_L, h, R\rangle$, where R is the uniform distribution on $\{0,1\}^k$. This is a stronger condition than required by the definition of **ERF**, so the theorem is established. ■

We make a few remarks about the security of this construction:

Remark 4.4 *The constant factors in this construction can be further improved to achieve* $k = (1-\delta)\ell$*, for any* $\delta > 0$*, by using the strong extractors of [21], under the slightly stronger assumption that* $\ell = \omega(\log^2 n)$*. Recall that* k *must be smaller than* ℓ*, so this is nearly optimal. Note that the statistical deviation obtained is also exponentially small in* $\Theta(\ell)$*.*

Remark 4.5 *Note that, in particular, we can choose* ℓ *to be* n^ϵ *for any* $\epsilon > 0$*, providing excellent security against partial key exposure. Seen another way, we can choose* n *to be any size larger than* ℓ*, to provide as much security against partial key exposure as we desire. The only drawback is that the output size is only* $\Theta(\ell)$*.*

4.3 Computational ERF

In the statistical construction given in the previous section, we were able to achieve essentially all the security against partial key exposure we wanted. The only thing limiting the applicability of the statistical construction is that the output size is limited to $k < \ell$. We would like to be able to achieve an arbitrary output size. By finally relaxing our requirement to *computational* security, we can easily accomplish this goal by using a pseudorandom generator as the final outermost layer of our construction. We also show that any **ERF** with $k > \ell$ implies the existence of pseudoranom generators, closing the loop.

Lemma 4.6 *Let* n, ℓ, m *be any polynomially related quantities. Let* f *be any statistical* ℓ-**ERF** *mapping* $\{0,1\}^n$ *to* $\{0,1\}^k$ *with negligible statistical deviation* ϵ*, for some* k *polynomially related to* m*. Let* G *be a pseudorandom generator stretching* $\{0,1\}^k$ *to* $\{0,1\}^m$*. Then the function* $g : \{0,1\}^n \to \{0,1\}^m$ *which sends* $x \mapsto G(f(x))$ *is a computational* ℓ-**ERF**.

Proof: Let $L \in \{{n \atop \ell}\}$. Suppose there was distinguisher D distinguishing between $A = \langle [r]_L, G(f(r))\rangle$ and $B = \langle [r]_L, R\rangle$ with advantage δ, where R is the uniform distribution on $\{0,1\}^m$. By the properties of f as a statistical ℓ-**ERF**, and the fact that statistical difference can only decrease by applying a function (G in our case), we have that $A = \langle [r]_L, G(f(r))\rangle$ and $C = \langle [r]_L, G(K)\rangle$ are within statistical distance ϵ of one another, where K is the uniform distribution on $\{0,1\}^k$. Thus, D distinguishes C from B with advantage $\delta - \epsilon$, as well. Note that in both B and C, the second component is independent of the first. Thus, we can use D to distinguish $G(K)$ from R (with advantage $\delta - \epsilon$), by simply picking a random $r \in \{0,1\}^n$, and providing D with $[r]_L$ as the first component. This contradicts the security of the generator G, completing the proof. ■

Theorem 4.7 *Assume one-way functions exist. Then for any ℓ, any $n = \ell^{\Theta(1)}$ and $k = n^{O(1)}$, there exists a computational ℓ-**ERF** mapping $\{0,1\}^n$ to $\{0,1\}^k$.*

Proof: We use Theorem 4.3 to build a statistical ℓ-**ERF**f mapping $\{0,1\}^n$ to $\{0,1\}^{\ell/10}$, with statistical deviation $2^{-\Omega(\ell)}$. Since ℓ is polynomially related to k, by [13], one-way functions imply the existence of a pseudorandom generator G mapping $\{0,1\}^{\ell/10}$ to $\{0,1\}^k$. Applying Lemma 4.6, we see that $g(r) = G(f(r))$ is a computational ℓ-**ERF**, as desired. ■

Finally, we show a "converse", i.e. that computational **ERF**'s with $k > \ell$ imply the existence of pseudorandom generators (and hence, one-way functions).

Lemma 4.8 *If there exists an ℓ-**ERF** f mapping $\{0,1\}^n$ to $\{0,1\}^k$, for $k > \ell$ (for infinitely many different values of ℓ, n, k), then one-way functions exist.*

Proof: We simply observe that the hypothesis implies the existence of the ensemble of distributions (we hide the obvious parametrization): $A = \langle [r]_L, f(r)\rangle$ and $B = \langle [r]_L, R\rangle$, where R is uniform on $\{0,1\}^k$. By assumption, A and B are computationally indistinguishable ensembles. Note that A can have at most n bits of entropy (since the only source of randomness is r), while B has $n - \ell + k \geq n + 1$ bits of entropy. Thus, the statistical difference between A and B is at least $1/2$. By the result of Goldreich [11], the existence of a pair of efficiently samplable distributions that are computationally indistinguishable but statistically far apart implies the existence of pseudorandom generators, and hence one-way functions. ■

As an immediate corollary, we get

Theorem 4.9 *For any (infinite sequence of) n, ℓ, k satisfying $\Omega(k^\epsilon) \leq \ell < k$, $n = \ell^{\Theta(1)}$, the following are equivalent:*

1. *The existence of ℓ-**ERF** $f : \{0,1\}^n \to \{0,1\}^k$.*
2. *The existence of one-way functions.*

A particularly useful kind of ℓ-**ERF** will be a *length-preserving* $f : \{0,1\}^k \to \{0,1\}^k$, which is impossible to achieve in the statistical or perfect sense. Thus, we get

Corollary 4.10 *If one-way-functions exist, length-preserving ℓ-**ERF** $f : \{0,1\}^k \to \{0,1\}^k$ exists, for any $\ell = \Omega(k^\epsilon)$.*

4.4 Applications of ERF

As we said, ℓ-**ERF** $f : \{0,1\}^n \to \{0,1\}^k$ allows one to represent a random secret in an "exposure-resilient" way. In Section 5 we show how to construct **AONT**'s using **ERF**'s. Here we give some other examples.

As an immediate example, especially when $k > n$, it allows us to obtain a much stronger form of pseudorandom generator, which not only stretches n bits to k bits, but remains pseudorandom when any $(n - \ell)$ bits of the seed are revealed. As a natural extension of the above example, we can apply it to private-key cryptography. A classical one-time private key encryption scheme over $\{0,1\}^k$ chooses a random shared secret key $r \in \{0,1\}^n$ and encrypts $x \in \{0,1\}^k$ by the pseudorandom "one-time pad" $G(r)$ (where G is a **PRG**), i.e. $E(x;\ r) = x \oplus G(r)$. We can make it resilient to the partial key exposure by replacing **PRG** G with **ERF** f.

For the next few examples, we assume for convenience that **ERF** $f : \{0,1\}^k \to \{0,1\}^k$ is length-preserving. Using such f, we show how to obtain *exposure-resilient form of a pseudorandom function family* (**PRF**). A **PRF** family $\mathcal{F} = \{F_s \mid s \in \{0,1\}^k\}$ has the property that f_s is indistinguishable from a random function when the seed s is chosen at random from $\{0,1\}^k$, but the pseudorandomness is only guaranteed if s is completely hidden. Defining $\tilde{F}_s = F_{f(s)}$, we get a new pseudorandom function family $\tilde{\mathcal{F}} = \{\tilde{F}_s \mid s \in \{0,1\}^k\}$, which remains pseudorandom even when all but ℓ bits of the seed s are known. We call such family an *exposure-resilient* **PRF**. We appply this again to private-key cryptography. The classical private-key encryption scheme selects a random shared key $s \in \{0,1\}^k$ and encrypts x by a pair $\langle x \oplus F_s(R), R\rangle$, where $\mathcal{F} = \{F_s : \{0,1\}^k \to \{0,1\}^k \mid s \in \{0,1\}^k\}$ is a **PRF**, and R is chosen at random. Again, replacing $\mathcal{F}$ by an exposure-resilient **PRF**, we obtain resilience against partial key exposure. Here, our secret key is $s \in \{0,1\}^k$, but $f(s)$ is used as the index to a regular **PRF**.

In fact, we can achieve security even against what we call the *gradual key exposure* problem in the setting with random private keys. Namely, consider a situation where the adversary is able to learn more and more bits of the secret key over time. Here, we do not place any upper bound on the amount of information that the adversary can learn about the secret key, but instead assume only that the rate at which the adversary can gain information is bounded. For example, suppose that every week the adversary somehow learns at most b bits of our secret r. We know that as long as the adversary misses ℓ bits of r, the system is secure[5]. However, if our key is relatively short, pretty soon there is a danger that the adversary knows more than $k - \ell$ bits of r, so the system is no longer secure. We argue that we can avoid this, provided the rate b the adversary learns our secret is not too large. Namely, both parties periodically (say with period slightly less than $(k - \ell)/b$ weeks), update our key by setting $r_{new} = f(r_{old})$. Since at the time of each update, the adversary missed at least ℓ bits of our current key r, the value $f(r)$ is still pseudorandom, and thus secure. Hence, we can agree on the secret key only *once*, even if the adversary continuously learns more and more of our secret!

5 All-Or-Nothing Transform (AONT)

As we pointed out, no **AONT** constructions without random oracles are known. We give several such constructions. One of our constructions implies that for the interesting settings of parameters, the existence of ℓ-**AONT**'s, ℓ-**ERF**'s and one-way functions are equivalent. The other construction can be viewed as the special case of the OAEP contruction of Bellare and Rogaway [2]. Thus, our result can be viewed as the first step towards abstracting the properties of the random oracle that suffice for this construction to work. Finally, we give a "worst-case/average-case" reduction for **AONT**'s that shows that one only needs to check the definition of **AONT** for *random* x_0, x_1.

[5] Here we assume that our **ERF** is secure even against adaptive key exposure, where the adversary may choose which next bits to learn based on his current information. However, our constructions achieve this.

5.1 Simple Construction using ERF

We view the process of creating ℓ-**AONT** as that of *one-time private-key encryption*, similarly to the application in Section 4.4. Namely, we look at the simplest possible one-time private key encryption scheme – the one-time pad, which is unconditionally secure. Here the secret key is a random string R of length k, and the encryption of $x \in \{0,1\}^k$ is just $x \oplus R$. We simply replace R by $f(r)$ where f is ℓ-**ERF** and r is our new secret. We get

Theorem 5.1 *Let $f : \{0,1\}^n \to \{0,1\}^k$ be computational (statistical, perfect) ℓ-***ERF**. *Define $T : \{0,1\}^k \to \{0,1\}^n \times \{0,1\}^k$ (that uses n random bits r) as follows: $T(x;\ r) = \langle r, f(r) \oplus x\rangle$. Then T is computational (statistical, perfect) ℓ-***AONT** *with secret part r and public part $f(r) \oplus x$.*

Proof: Take any $L \in \{{n \atop \ell}\}$, and $x_0, x_1 \in \{0,1\}^k$. We have to show that

$$\langle x_0, x_1, [r]_L, f(r) \oplus x_0\rangle \approx \langle x_0, x_1, [r]_L, f(r) \oplus x_1\rangle$$

Let $C = x_0 \oplus x_1$. Adding x_0 to the last component of both distributions and noticing that C is sufficient statistics, we get (using both parts of Lemma 2.2) that it suffices to show

$$\langle C, [r]_L, f(r)\rangle \approx \langle C, [r]_L, f(r) \oplus C\rangle$$

But this follows immediately from the definition of **ERF** and Lemma 2.1, since C is independent of r. ■

As an immediate corollary of Theorems 4.7 and 5.1, we have:

Theorem 5.2 *Let ℓ, s, k be any settings of parameters such that $s = \ell^{\Theta(1)}$ and $k = \ell^{O(1)}$. Then there exists an ℓ-***AONT** *for messages of length k, with secret output size s and public output size k.*

Remark 5.3 *To see the power of the above construction, observe that one can decide on any value for ℓ (which will essentially be a security parameter), and any value for $s - \ell$, which is the number of bits the adversary can see without gaining any information, with only the constraint that $s - \ell$ be polynomially related to ℓ. Then, one can build an ℓ-***AONT** *with secret outputs of size s and public outputs of size k, for messages of any length k polynomially related to ℓ. In particular, we can select parameters such that $\ell = s^\epsilon$ for any $\epsilon > 0$ and $s = O(k)$ if we so choose.*

Remark 5.4 *Observe that any ℓ-***AONT** *with public and secret outputs of length p and s, respectively, also gives a secret-only ℓ'-***AONT** *with output size $N = s+p$ and $\ell' = \ell+p$ (since if the adversary misses $\ell + p$ bits of the output, that means it must miss at least ℓ bits of the secret output). Note that viewing our construction as a secret-only ℓ-***AONT** *on messages of length k, if one requires a security paremeter of ℓ_0, we will need to pick $\ell = \ell_0 + k$. However, we make two observations:*

1. *As before, for any choice of security parameter ℓ_0, and for any choice of the resilience in terms of the number of bits e we allow the adversary to learn (polynomially related to ℓ_0), and for any message length k, we can build an secret-only ℓ-***AONT** *with output length $N = e + \ell_0 + k$.*

2. *Also, as before, for any security parameter ℓ_0 and message length k and $\epsilon > 0$, we can build a secret-only ℓ-***AONT** *with output size N, such that $\ell = N^\epsilon$. Here, however, $N = (\ell_0 + k)^{1/\epsilon}$, whereas in the case of* **AONT***'s with public and secret outputs, s could essentially be chosen independently of k.*

We conclude with a remark about the applicability of **AONT**'s as certain kinds of computational secret sharing schemes.

Remark 5.5 *Consider an ℓ-**AONT** with public output of size p and secret output of size s. We can interpret this as being a kind of "gap" computational secret sharing scheme. For some secret x, we apply the **AONT** to obtain a secret output y_1 and public output y_2. Here, we think of y_2 as being a public share that is unprotected. We interpret the bits of y_2 as being tiny shares that are only 1 bit long, with one share given to each of s parties. We are guaranteed that if all the players cooperate, by the invertability of the **AONT**, they can recover the secret x. On the other hand, if $s - \ell$ or fewer of the players collude, they gain no computational information about the secret whatsoever. We call this a "gap" secret sharing scheme because there is a gap between the number of players needed to reconstruct the secret and the number of players that cannot gain any information. Note that such a gap is unavoidable when the shares are smaller than the security parameter. Using our constructions, we can obtain such schemes for any value of ℓ larger than the security parameter, and any value of s larger than ℓ.*

5.2 AONT implies OWFs

Theorem 5.6 *Assume we have a computational ℓ-**AONT** $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^p = \{0,1\}^N$ where $\ell < k - 1$. Then one-way functions exist.*

Proof: To show that **OWF**'s exist it is sufficient to show that *weak* **OWF**'s exist [10]. Fix $L = [\ell] \subseteq [s]$. Define $g(x_0, x_1, b, r) = \langle x_0, x_1, [y]_L \rangle$, where $y = T(x_b;\ r)$. We claim that g is a weak **OWF**. Assume not. Then there is an inverter A such that when x_0, x_1, b, r are chosen at random, $y = T(x_b;\ r)$, $z = [y]_L$, $\langle \tilde{b}, \tilde{r} \rangle = A(x_0, x_1, z)$, $\tilde{y} = T(x_{\tilde{b}};\ \tilde{r})$, $\tilde{z} = [\tilde{y}]_L$, we have $\Pr(z = \tilde{z}) > \frac{3}{4}$.

To show that there exist x_0, x_1 breaking the indistinguishability property of T, we construct a distinguisher F for T that has non-negligible advantage for *random* $x_0, x_1 \in \{0,1\}^k$. This would show that the required x_0, x_1 exist. Hence, the job of F is the following. x_0, x_1, b, r are chosen at random, and we set $y = T(x_b;\ r)$, $z = [y]_L$. Then F is given the challenge z together with x_0 and x_1. Now, F has to predict b correctly with probability non-negligibly more than $1/2$. We let F run $A(x_0, x_1, z)$ to get $\tilde{b}, \tilde{r}$. Now, F sets $\tilde{y} = T(x_{\tilde{b}};\ \tilde{r})$, $\tilde{z} = [\tilde{y}]_L$. If indeed $\tilde{z} = z$ (i.e. A succedeed), F outputs $\tilde{b}$ as its guess, else it flips a coin.

Let B be the event that A succeeds inverting. From the way we set up the experiment, we know that $\Pr(B) \geq \frac{3}{4}$. Call U the event that when x_0, x_1, b, r are chosen at random, $[T(x_b;\ r)]_L \in [T(x_{1-b})]_L$, i.e. there exists some r' such that $[T(x_{1-b};\ r')]_L = z$ or $g(x_0, x_1, 1-b, r') = g(x_0, x_1, b, r)$. If U does not happen and A succeeded inverting, we know that $\tilde{b} = b$, as it is $1 - b$ is an impossible answer. Thus, using $\Pr(X \wedge \overline{Y}) \geq \Pr(X) - \Pr(Y)$, we get:

$$\begin{aligned}
\Pr(\tilde{b} = b) &\geq \frac{1}{2}\Pr(\overline{B}) + \Pr(B \wedge \overline{U}) \geq \frac{1}{2}\Pr(\overline{B}) + \Pr(B) - \Pr(U) \\
&= \frac{1}{2} + \frac{1}{2}\Pr(B) - \Pr(U) \geq \frac{1}{2} + \left(\frac{3}{8} - \Pr(U)\right)
\end{aligned}$$

To get a contradiction, we show that $\Pr(U) \leq 2^{\ell - k}$, which is at most $\frac{1}{4} < \frac{3}{8}$ since $\ell < k - 1$. To show this, we observe that U measures the probability of the event that when we choose x, x', r at random and set $z = [T(x;\ r)]_L$, what is the probability that there is r' such that $z = [T(x';\ r')]_L$. However, for any fixed setting of z, there are only 2^ℓ possible completions $y \in \{0,1\}^N$. And for each such completion y, invertability of T implies that there could be at most one $x' \in T^{-1}(y)$. Hence, for any setting of z, at most 2^ℓ out of 2^k possible x' have a chance to have the corresponding r'. Thus, $\Pr(U) \leq 2^{\ell-k}$ indeed. ■

Up to a constant factor, the result is optimal, since one can achieve statistical (even secret-only) ℓ-**AONT** with $\ell = O(k)$. Indeed, we use statistical ℓ-**ERF** $f : \{0,1\}^n \to \{0,1\}^k$ with $\ell = O(k)$ (and any $n \geq \ell$ we like) as achieved in Theorem 4.3, and then apply Theorem 5.1 to it. This yields unconditional ℓ-**AONT** with $N = n + k$ and $\ell = O(k)$. Merging secret and public parts together gives secret-only ℓ'-**AONT** with $\ell' = \ell + k = O(k)$ still.

5.3 Towards secret-only AONT

We also give another construction of an **AONT** based on any length-preserving function f such that both $[r \mapsto f(r)]$ and $[r \mapsto f(r) \oplus r]$ are **ERF**'s. This construction can be viewed as a special case of the OAEP construction as defined by Equations (3)-(4) (but without random oracles). Thus, our analysis makes a step towards abstracting the properties of the random oracle needed to make the OAEP work as an **AONT**. It also has the advantage of meeting the standard definition of an **AONT** (without separate public and secret outputs), while retaining a relatively short output length.

Recall that the OAEP construction sets $T(x;\ r) = \langle u, t\rangle$, where $u = G(r) \oplus x$, $t = H(u) \oplus r$, and $G : \{0,1\}^n \to \{0,1\}^k$ and $H : \{0,1\}^k \to \{0,1\}^n$ are some functions (e.g., random oracles). We analyze the following construction, which is a special case of the OAEP construction with $n = k$, and H being the identity function.

$$u = f(r) \oplus x \tag{5}$$
$$t = s \oplus r \tag{6}$$

where $f : \{0,1\}^k \to \{0,1\}^k$. Thus, $T(x;\ r) = \langle f(r) \oplus x, (f(r) \oplus r) \oplus x\rangle$, and the inverse is $I(u,t) = u \oplus f(u \oplus t)$.

Theorem 5.7 *Assume f is such that both $f(r)$ and $(f(r) \oplus r)$ are length-preserving computational ℓ-* **ERF***s. Then T above is computational secret-only 2ℓ-***AONT**.

The proof and the related discussion can be found in Appendix D. We note, though, that random oracle f clearly satisfies the conditions of the Theorem. Thus, we obtained a simple proof that even removing random oracle H leaves the OAEP construction secure for $n = k$. We believe that the assumption of the theorem is quite reasonable, even though we leave open the question of constructing such f based on standard assumptions.

5.4 Worst-case / Average-case Equivalence of AONT

In the definition of **AONT** we require that Equation (2) holds for any x_0, x_1. This implies (and is equivalent) to saying that it holds if one is to choose x_0, x_1 according to any distribution $p(x_0, x_1)$. A natural such distribution is the uniform distribution, which selects random x_0, x_1 uniformly and independently from $\{0,1\}^k$. We call an **AONT** secure against (possibly only) the uniform distribution an *average-case* **AONT**. A natural question to ask is whether average-case **AONT** implies (regular) **AONT** with comparable parameters, which can be viewed as the worst-case/average case equivalence. We show that up to a constant factor, the notions are indeed identical in the statistical or computational settings. Below we assume without loss of generality that our domain is a finite field (e.g. $GF(2^k)$), so that addition and multiplication are defined.

Lemma 5.8 *Let $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^p$ be an average-case (statistical or computational) ℓ-***AONT***. Then the following $T' : \{0,1\}^k \to \{0,1\}^{4s} \times \{0,1\}^{4p}$ is a (statistical or computational) 4ℓ-* **AONT***, where a_1, a_2, b are chosen uniformly at random subject to $a_1 + a_2 \neq 0$ (as part of the randomness of T'):*

$$T'(x) = \langle T(a_1), T(a_2), T(b), T((a_1 + a_2) \cdot x + b)\rangle$$

In the above output, we separately concatenate secret and public parts of T's output.

Proof: See Appendix E ■

6 Conclusions

We have studied the problem of partial key exposure and related questions. We have proposed solutions to these problems based on new constructions of the All-Or-Nothing Transform in the standard model based on any one-way function, without random oracles.

The key ingredient in our approach is an interesting new primitive which we call an Exposure-Resilient Function. This primitive has natural applications in combatting key exposure, and we believe it is also interesting in its own right. We showed how to build essentially optimal **ERF**'s based on any one-way function. We also explored many other interesting properties of **ERF**'s and **AONT**'s.

References

[1] M. Bellare, S. Miner. A Forward-Secure Digital Signature Scheme. In *Proc. of Crypto*, pp. 431–448, 1999.

[2] M. Bellare, P. Rogaway. Optimal Asymetric Encryption. In *Proc. of EuroCrypt*, pp. 92–111, 1995.

[3] C. Bennett, G. Brassard, C. Crepeau, U. Maurer. Generalized Privacy Amplification. In *IEEE Transactions on Information Theory*, 41(6), 1995.

[4] G. Blackley. Safeguarding Cryptographic Keys. In *Proc. of AFIPS 1979 National Computer Conference*, 1979.

[5] V. Boyko. On the Security Properties of the OAEP as an All-or-Nothing Transform. In *Proc. of Crypto*, pp. 503–518, 1999.

[6] C. Cachin, U. Maurer. Linking information reconciliation and privacy amplification. In *Journal of Cryptology*, 10(2):97-110, 1997.

[7] B. Chor, J. Friedman, O. Goldreich, J. Hastad, S. Rudich, R. Smolensky. The Bit Extraction Problem or t-resilient Functions. In *Proc. of FOCS*, pp. 396–407, 1985.

[8] Y. Desmedt, Y. Frankel. Threshold Cryptosystems. In *Proc. of Crypto*, pp. 307–315, 1989.

[9] A. Dornan. New Viruses Search For Strong Encryption Keys. In *PlanetIT Systems Management News*, `http://www.planetit.com/techcenters/docs/systems_management/news/PIT19990317S0015`, March, 1999.

[10] O. Goldreich. Foundations of Cryptography (Fragments of a Book). Available at `http://www.wisdom.weizmann.ac.il/home/oded/public_html/frag.html`

[11] O. Goldreich. A Note on Computational Indistinguishability. In *IPL*, 34:277–281, 1990.

[12] O. Goldreich, A. Wigderson. Tiny Families of Functions with Random Properties: A Quality-Size Trade-off for Hashing. In *Proc. of STOC*, pp. 574–583, 1994.

[13] J. Hastad, R. Impagliazzo, L. Levin, M. Luby. A Pseudorandom generator from any one-way function. In *Proc. of STOC*, 1989.

[14] A. Hertberg, M. Jackobson, S. Jarecki, H. Krawczyk, M. Yung. Proactive public key and signature schemes. In *Proc. of Conference on Computer and Communication Security*, ACM, 1997.

[15] R. Impagliazzo, L. Levin, M. Luby. Pseudo-random Generation from one-way functions. In *Proc. of STOC*, pp. 12–24, 1989.

[16] M. Jakobsson, J. Stern, M. Yung. Scramble All, Encrypt Small. In *Proc. of Fast Software Encryption*, pp. 95–111, 1999.

[17] H. Krawczyk. Secret Sharing Made Short. In *Proc. of Crypto*, pp. 136–146, 1993.

[18] F. MacWilliams, J. Sloane. Theory of Error-Correcting Codes, Amsterdam, 1981.

[19] J. Naor, M. Naor. Small-Bias Probability Spaces: Efficient Constructions and Applications. In *SIAM J. Computing*, 22(4):838-856, 1993.

[20] N. Nisan. Extracting Randomness: How and Why A survey. In *IEEE Conference on Computational Complexity*, pp. 44–58, 1996

[21] R. Raz, O. Reingold, S. Vadhan. Error Reduction for Extractors. In Proc. of FOCS, pp. 191–201, 1999.

[22] R. Rivest. All-or-Nothing Encryption and the Package Transform. In *Fast Software Encryption, LNCS*, 1267:210–218, 1997.

[23] A. Shamir. How to share a secret. In *Communications of the ACM*, 22:612-613, 1979.

[24] A. Shamir, N. Someren. Playing "hide and seek" with stored keys. In *Proc. of Financial Cryptography*, 1999.

[25] N. Someren. How not to authenticate code. Crypto'98 Rump Session, Santa Barbara, 1998.

[26] A. Srinivasan, D. Zuckerman. Computing with Very Weak Random Sources. In *Proc. of FOCS*, pp. 264–275, 1994.

[27] D. Stinson. Some observations on all-or-nothing transforms. Available from `http://cacr.math.uwaterloo.ca/ dstinson/papers/AON.ps`, 1998.

[28] L. Trevisan. Construction of Extractors Using Pseudo-Random Generators. In Proc. of STOC, pp. 141–148, 1999.

A Proof of Lemma 2.1

Lemma A.1 *Let α, β be two (possibly dependent) random variables taking value in $\{0,1\}$. Let D be the following experiment: observe α and β. If $\alpha = \beta$, then flip a coin, else output α $(= 1 - \beta)$. Let γ be the output of D. Then*

$$\Pr(\gamma = 1) = \frac{1}{2} + \frac{1}{2} \cdot [\Pr(\alpha = 1) - \Pr(\beta = 1)]$$

Proof: We use the formula that for any events A and B, $\Pr(A \wedge B) + \Pr(A \wedge \overline{B}) = \Pr(A)$.

$$\begin{aligned}
\Pr(\gamma = 1) &= \Pr(\alpha = 1 \wedge \beta = 0) + \frac{1}{2} \cdot [\Pr(\alpha = 1 \wedge \beta = 1) + \Pr(\alpha = 0 \wedge \beta = 0)] \\
&= \frac{1}{2} \cdot \{[\Pr(\alpha = 1 \wedge \beta = 0) + \Pr(\alpha = 1 \wedge \beta = 1)] + [\Pr(\alpha = 1 \wedge \beta = 0) + \Pr(\alpha = 0 \wedge \beta = 0)]\} \\
&= \frac{1}{2} \cdot [\Pr(\alpha = 1) + \Pr(\beta = 0)] = \frac{1}{2} + \frac{1}{2} \cdot [\Pr(\alpha = 1) - \Pr(\beta = 1)]
\end{aligned}$$

■

We now prove Lemma 2.1, which stated the following: Let A and B be any two random variables. Let R be chosen uniformly at random and C be chosen according to a distribution p, both independently from A and B. Then the following are equivalent:

(1) $\langle A, B\rangle \approx \langle A, R\rangle$.

(2) $\langle A, B, C\rangle \approx \langle A, B \oplus C, C\rangle$, for any polynomial time sampleable (PTS) distribution p.

(3) $\langle A, B, C\rangle \approx \langle A, B \oplus C, C\rangle$, for uniform p.

We concentrate on the computational case, which is the hardest of the above.

$(1) \Rightarrow (2)$. Assume (2) is false for some PTS p, so there is an adversary F distinguishing $\langle A, B, C\rangle$ from $\langle A, B \oplus C, C\rangle$ with advantage ϵ. We construct a distinguisher D that distinguishes $\langle A, B\rangle$ from $\langle A, R\rangle$. D gets as input $\langle A, X\rangle$. It generates C according to p, sets $\alpha = F(A, X, C)$, $\beta = F(A, X \oplus C, C)$. Then D proceeds as in Lemma A.1. Thus,

$$\begin{aligned}
\Pr(\gamma = 1) &= \frac{1}{2} + \frac{1}{2} \cdot [\Pr(\alpha = 1) - \Pr(\beta = 1)] \\
&= \frac{1}{2} + \frac{1}{2} \cdot [\Pr(F(A, X, C) = 1) - \Pr(F(A, X \oplus C, C) = 1)]
\end{aligned}$$

When $X = B$, the difference above is at least ϵ, by assumption of F. Thus, $\Pr(\gamma = 1) > \frac{1}{2} + \frac{\epsilon}{2}$.

When $X = R$, both R and $R \oplus C$ are uniform and independent of C. Thus, $\Pr(F(A, X, C) = 1) = \Pr(F(A, X \oplus C, C) = 1)$, and so $\Pr(\gamma = 1) = \frac{1}{2}$. Hence, D is a good distinguisher indeed.

$(2) \Rightarrow (3)$ is trivial.

$(3) \Rightarrow (1)$. Let $R = B \oplus C$. If C is uniform and independent from A and B, then so is R. If there is an adversary that can distinguish $\langle A, B\rangle$ from $\langle A, R\rangle$, then there is an adversary distinguishing $\langle A, B, C\rangle$ from $\langle A, B \oplus C, C\rangle = \langle A, R, C\rangle$, that simply ignores the extra information C and runs the original adversary on the first two components.

B Discussion of AONT with Secret and Public Outputs

We now discuss the generalized definition of **AONT**. Recall that the standard definition requires that security should hold for *any* ℓ-element subset L of $[N]$. The interpretation is that we wish to protect the secret x, we encode secret x into a new secret $y = T(x)$ such that x is protected against the adversary learning all but ℓ bits of y. Thus, it is implicitly assumed that all parts of the transform are "equally important" and should have the same protection against the attacker. In reality, different parts of the transform serve different purposes for the decoding process. Some of them could be used just for the decoding process (so that the mapping is invertible), but are not important to keep secret against the attacker, while others are really the ones that do all the cryptographic work, and thus, should be kept secret.

For example, we could have a transform of output length $2k$, where, as long as the adversary does not learn $\sqrt{k}$ bits from the second half of the transform, we are completely secure, but are totally insecure if it learns the entire second half. This seems like a very reasonable solution to the key leakage problem; we will simply protect as hard as we can the second half of the transform, while the first part we might as well publish. However, in the standard setting we must set $\ell = k + \sqrt{k}$ to ensure that the adversary misses at least $\sqrt{k}$ bits of the second half. This seems to be an artificial setting for ℓ, indicating that more than half of the transform should be kept hidden. Common sense tells us that the real answer is $\ell = \sqrt{k}$, because first and second half serve different purposes, and we are secure as long as $\sqrt{k}$ bits of the second half remain hidden.

This leads us to the following more general notion of **AONT**. Here we can encode x into a "secret" part y_1 and a "public" part y_2, such that the public part might as well be published, while the secret part has our standard resilience property. Namely, the adversary learning all but ℓ bits of y_1 (and the entire public y_2) cannot learn anything useful about x. Thus, public part is only used to decode x back (in conjunction with the secret part), but we really do not care about protecting it. It is the secret part y_1 that is important to protect.

We argue that this generalized notion allow us more flexibility than before. First of all, it allows reasonable **AONT** constructions, as in the example above, to have small ℓ, as they should. Secondly, while without the public part, the size of the secret part had to be at least the size of the message, now it can be much smaller (at the expense of the public part). Thus, the public part may be stored on some insecure device with fast access time (like public cache), while secret part may be stored further away in some publically read-protected memory (and still give a guarantee that small accidental leakage will not compromise the security). In addition, we will see that more general **AONT**'s (with the public part) seem to be more efficient and much easier to construct than the corresponding **AONT**'s with secret part only. We also point out that this generalized notion of **AONT** naturally suffices for all applications of **AONT** we are aware of.

C Error Correcting Codes and Perfect ERF

A *binary linear* $[n, k, d]$ *error-correcting code* can be seen as a linear transformation from $\{0,1\}^k$ to $\{0,1\}^n$ (where these are viewed as vector spaces over $GF(2)$). Thus, such a code can be described by an $k \times n$ *generating matrix* M over $GF(2)$. For any vector $v \in \{0,1\}^k$, the codeword corresponding to v is vM. A code is said to have *minimum distance* d if for every two distinct vectors $u, v \in \{0,1\}^k$, uM and vM differ on at least d coordinates. Note that by linearity, this is equivalent to requiring that every non-zero codeword has at least d non-zero components. For further information on error correcting codes, as well as for proofs of the results on error correcting codes that we use, see [18].

Theorem C.1 ([7]) *Let M be a $k \times n$ matrix. Define $f(r) = M \cdot r$, where $r \in \{0,1\}^n$. Then f is perfect ℓ-***ERF** *if and only if M is the generator matrix for a code of distance $n - \ell + 1$.*

The proof of this theorem follows by observing that every codeword is a linear combination of the rows of M (since codewords are of the form uM for $u \in \{0,1\}^k$). The distance properties of the code imply that the rows of M are linearly independent, and furthermore that every non-trivial linear combination of the rows creates a codeword of Hamming weight at least d. Hence, even after removing any $(d-1) = (n-\ell)$ columns of M, the resulting rows of M are still linearly independent, which gives the desired result.

We apply this result to a special kind of code. A code is said to be *asymptotically good* if $n = O(k)$ and $d = \Omega(n)$ (i.e., the three parameters n, k, and d differ by multiplicative constants). Many explicit constructions for asymptotically good codes (e.g. the Justesen code) also exist. Using such a code, we can get both ℓ/n and k/n to be (very small) constants.

Note that for any code, $k \leq n - d + 1$ (this is called the *singleton bound*). Thus, we have $k \leq n - (n - \ell + 1) + 1 = \ell$, as expected. Also, it is known that $d \leq n/2$ for $k \geq 2 \log n$. This implies that we are limited to have $\ell \geq n/2$. However, at the expense of making $n = poly(k)$, using a Reed-Solomon code concatenated with a Hadamard code, we can achieve $\ell = n - d + 1$ to be arbitrarily close to $n/2$, but can never cross it.

D Discussion and Proofs for Section 5.3

Recall that the OAEP construction sets $T(x;\ r) = \langle u, t\rangle$, where $u = G(r) \oplus x$, $t = H(u) \oplus r$, and $G : \{0,1\}^n \to \{0,1\}^k$ and $H : \{0,1\}^k \to \{0,1\}^n$ are some functions (e.g., random oracles). Let's try to develop some informal intuition of why this construction works; in particular, to separate the properties of G and H that are essential (and hopefully sufficient) for this construction to be an **AONT**. We look at the two extreme cases.

First, assume we know u and miss ℓ bits of t. Then we miss ℓ bits of r, since $r = H(u) \oplus t$. Note that $x = G(r) \oplus u$, so in order to "miss x completely", G must have the property that missing ℓ bits of G's random input r makes the output pseudorandom (random oracle clearly satisfies this). But this is *exactly* the notion of ℓ-**ERF**! Thus, G must be an **ERF**, and this suffices to handle the case when we miss ℓ bits of t.

Now assume that we know t and are missing ℓ bits of u. Assume for a second that H is a random oracle. Then, since $r = H(u) \oplus t$, we are essentially missing r completely. But from the previous argument about G, we know that even missing ℓ bits of r leaves x completely unknown. Thus, random oracle H achieves even more than we need. In some sense, as long as H does not "unhide" information we miss about u, we will miss at least ℓ bits of r. In other words, assume H satisfies the property that missing ℓ of its input bits implies "missing" at least ℓ of its output bits. Then missing ℓ bits of u implies missing ℓ bits of r, which implies missing entire $G(r)$, which implies missing x completely. So we ask the question of which H satisfy this property? Clearly, the easiest one is the identity function (assuming $n = k$). This has led us to analyze the following construction, which is a special case of the OAEP construction with $n = k$, and H being the identity function.

$$u = f(r) \oplus x \tag{7}$$
$$t = s \oplus r \tag{8}$$

where $f : \{0,1\}^k \to \{0,1\}^k$. Thus, $T(x;\, r) = \langle f(r) \oplus x, (f(r) \oplus r) \oplus x\rangle$, and the inverse is $I(u,t) = u \oplus f(u \oplus t)$.

Theorem D.1 *Assume f is such that both $f(r)$ and $(f(r) \oplus r)$ are length-preserving computational ℓ-***ERF***s. Then T above is computational secret-only 2ℓ-***AONT**.

Proof: Let $N = 2k$ be the size of the output, $L_1 = \{1 \dots \ell\}$, $L_2 = \{\ell{+}1 \dots 2\ell\}$. As in Equations (7)-(8), let $u = f(r) \oplus x$, $t = (f(r) \oplus r) \oplus x)$. Note that $u \oplus t = r$.

Take any $L \in \{{N \atop 2\ell}\}$ and any $x_0, x_1 \in \{0,1\}^k$. It must be the case that either $|L \cap L_1| \geq \ell$ or $|L \cap L_2| \geq \ell$. Thus, it suffices to show security when we either know t completely and miss ℓ bits of u, or when we know u completely and miss ℓ bits of t. Hence, it suffices to assume that $|L| = \ell$ and consider the two cases separately.

1) $L \subseteq L_1$. Then we must show that

$$\langle x_0, x_1, [f(r) \oplus x_0]_L, (f(r) \oplus r) \oplus x_0\rangle \approx \langle x_0, x_1, [f(r) \oplus x_1]_L, (f(r) \oplus r) \oplus x_1\rangle$$

Since $[u]_L \oplus [t]_L = [r]_L$, by Lemma 2.2 the above is the same as

$$\langle x_0, x_1, [r]_L, (f(r) \oplus r) \oplus x_0\rangle \approx \langle x_0, x_1, [r]_L, (f(r) \oplus r) \oplus x_1\rangle$$

Adding x_0 to the last component and letting $C = x_0 \oplus x_1$, using both parts of Lemma 2.2, above is the same as

$$\langle C, [r]_L, (f(r) \oplus r)\rangle \approx \langle C, [r]_L, (f(r) \oplus r) \oplus C\rangle$$

The result now follows from the fact that $(f(r) \oplus r)$ is ℓ-**ERF** and Lemma 2.1.

2) $L \subseteq L_2$. The proof is identical to above with the roles of $f(r)$ and $(f(r) \oplus r)$ interchanged. In particular, security follows from the fact that $f(r)$ is ℓ-**ERF**. ■

We remark that non-trivial length-preserving **ERF**'s can exist only in the computational sense, since $\ell \geq k$ for any statistical **ERF**.

E Proof of Lemma 5.8

First, since T is invertible and $a_1 + a_2 \neq 0$, we have that T' is invertible (invert all four components and recover x). Assume now that T' is not an ℓ-**AONT**, that is for some $L' \in \{{4s \atop 4\ell}\}$, $x'_0, x'_1 \in \{0,1\}^k$ (obviously, $x'_0 \neq x'_1$) we have

$$\langle x'_0, x'_1, [T'(x'_0)]_{L'}\rangle \not\approx \langle x'_0, x'_1, [T'(x'_0)]_{L'}\rangle$$

Call z the last input to T, i.e. either $(a_0 + a_1) \cdot x'_0 + b$ or $(a_0 + a_1) \cdot x'_1 + b$. Let x_0, x_1 be selected at random form $\{0,1\}^k$, and we are given $\langle x_0, x_1, [T(x_i)]_L\rangle$, where i is either 0 or 1. We need to choose L allowing us to "blindly transform" $[T(x_i)]_L$ into $[T'(x'_i)]_{L'}$. Then, since the latter distribution is not indistinguishable for $i = 0$ and $i = 1$, the so is the former distribution. Thus, the quadruple $\langle a_1, a_2, b, z\rangle$ resulting from our "blind" transformation should satisfy

$$(a_1 + a_2) \cdot x'_i + b = z \tag{9}$$

Moreover, $\langle a_1, a_2, b, z\rangle$ should be (statistically close to) *random* satisfying the corresponding equation above (subject to $a_1 + a_2 \neq 0$).

We construct L by looking at which part of the output of T' has the most bits in L'. Namely, let $L_j = \{m \in [\ell] \mid m + j\ell \in L'\}$. Since $|L'| \geq 4\ell$, some $|L_j| \geq \ell$. We let L be any ℓ-element subset of this L_j. We now consider 4 cases depending on the identity of this j. In all the cases, one of a_1, a_2, b, z (depending on j) will have to be implicitly set to x_i (for unknown i). The remaining 3 parameters must be then set in the *same way independent of* i (but stay otherwise random), such that irrespective of $i = 0$ or $i = 1$, Equation (9) is satisfied. We now illustrate how this can be done for each $j = 1, 2, 3, 4$.

- $|L_1| \geq \ell$. Hence, we know that

  $$\langle x'_0, x'_1,\ [T(a_1)]_L,\ T(a_2),\ T(b),\ T((a_1 + a_2) \cdot x'_0 + b)\rangle \not\approx$$
  $$\langle x'_0, x'_1,\ [T(a_1)]_L,\ T(a_2),\ T(b),\ T((a_1 + a_2) \cdot x'_1 + b)\rangle$$

  Clearly, we should (implicitly) make $a_1 = x_i$ (which is random since x_i is random). In order to set a_2, b, z in an identical manner independent of i, we solve the linear system in a_2 and d (d is to be interpreted as $z - b$)

  $$\begin{aligned} (x_0 + a_2) \cdot x'_0 &= d \\ (x_1 + a_2) \cdot x'_1 &= d \end{aligned}$$

  This system is always solvable since $x'_0 \neq x'_1$. Moreover, a_2 and d are random and independent of each other for a random choice of x_0 and x_1. We then pick random b, z such that $z - b = d$. We note that $x_0 + a_2$ or $x_1 + a_2$ are 0 with only negligibly small probability (since resulting a_2 is random), so we can ignore this case happening for the statistical or computational setting. Then we immediately observe that $\langle x'_i, a_2, b, z\rangle$ satisfy $(x_i + a_2) \cdot x'_i + b = z$. Moreover, this is a *random* quadruple of inputs to T used for computing $T'(x'_i)$ (technically, statistically close to it). Hence, by the argument above, we obtain a contradiction to the fact that T is ℓ-**AONT**.

- $|L_2| \geq \ell$. This is symmetric to the above with a_1 and a_2 interchanged.

- $|L_3| \geq \ell$. Hence, we know that

$$\langle x'_0, x'_1,\ T(a_1),\ T(a_2),\ [T(b)]_L,\ T((a_1+a_2)\cdot x'_0 + b)\rangle \not\approx$$
$$\langle x'_0, x'_1,\ T(a_1),\ T(a_2),\ [T(b)]_L,\ T((a_1+a_2)\cdot x'_1 + b)\rangle$$

Clearly, we should (implicitly) make $b = x_i$ (which is random since x_i is random). In order to set a_1, a_2, z in an identical manner independent of i, we solve the linear system in a and z (a is to be interpreted as $a_1 + a_2$)

$$\begin{aligned} a\cdot x'_0 + x_0 &= z \\ a\cdot x'_1 + x_1 &= z \end{aligned}$$

This system is always solvable since $x'_0 \neq x'_1$. Moreover, a and z are random and independent of each other for a random choice of x_0 and x_1. Also, unless $x_0 = x_1$ (which happens with exponentially small probability), $a \neq 0$. Pick random a_1, a_2 such that $a_1 + a_2 = a$. Then $\langle a_1, a_2, x_i, z\rangle$ satisfy $(a_1 + a_2)\cdot x'_i + x_i = z$. Moreover, this is a *random* quadruple of inputs to T used for computing $T'(x'_i)$ (technically, statistically close to it). Hence, we obtain a contradiction to the fact that T is ℓ-**AONT**.

- $|L_4| \geq \ell$. Hence, we know that

$$\langle x'_0, x'_1,\ T(a_1),\ T(a_2),\ T(b),\ [T((a_1+a_2)x'_0 + b)]_L\rangle \not\approx$$
$$\langle x'_0, x'_1,\ T(a_1),\ T(a_2),\ T(b),\ [T((a_1+a_2)x'_1 + b)]_L\rangle$$

Clearly, we should (implicitly) make $z = x_i$ (which is random since x_i is random). In order to set a_1, a_2, b in an identical manner independent of i, we solve the linear system in a and b (a is to be interpreted as $a_1 + a_2$)

$$\begin{aligned} a\cdot x'_0 + b &= x_0 \\ a\cdot x'_1 + b &= x_1 \end{aligned}$$

This system is always solvable since $x'_0 \neq x'_1$. Moreover, a and b are random and independent of each other for a random choice of x_0 and x_1. Also, unless $x_0 = x_1$ (which happens with exponentially small probability), $a \neq 0$. Pick random a_1, a_2 such that $a_1 + a_2 = a$. Then $\langle a_1, a_2, b, x_i\rangle$ satisfy $(a_1 + a_2)\cdot x'_i + b = x_i$. Moreover, this is a *random* quadruple of inputs to T used in computing $T'(x'_i)$ (technically, statistically close to it). Hence, we obtain a contradiction to the fact that T is ℓ-**AONT**.