Exposure-Resilient Functions and All-Or-Nothing Transforms

Ran Canetti[1], Yevgeniy Dodis[2], Shai Halevi[1], Eyal Kushilevitz[3], and Amit Sahai[2]

[1] IBM T.J. Watson Research Center, P.O. Box 704, Yorktown Heights, New York 10598, USA. {canetti,shaih}@watson.ibm.com.
[2] Lab. for Computer Science, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02149, USA. {yevgen,amits}@theory.lcs.mit.edu.
[3] IBM T.J. Watson Research Center and Department of Computer Science, Technion, Haifa 32000, Israel. eyalk@cs.technion.ac.il.

Abstract. 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 that remain secure even when an adversary is able to learn *almost all* of the secret key.

The key to our approach is a new primitive of independent interest, which we call an *Exposure-Resilient Function* (**ERF**) – a deterministic function whose output appears random (in a perfect, statistical or computational sense) even if *almost all* the bits of the input are known. **ERF**'s by themselves efficiently solve the partial key exposure problem in the setting where the secret is simply a random value, like in private-key cryptography. They can also be viewed as very secure pseudorandom generators, and have many other applications.

To solve the general partial key exposure problem, we use the (generalized) notion of an *All-Or-Nothing Transform* (**AONT**), an *invertible* (randomized) transformation T which, nevertheless, reveals "no information" about x even if *almost all* the bits of $T(x)$ are known. By applying an **AONT** to the secret key of any cryptographic system, we obtain security against partial key exposure. To date, the only known security analyses of **AONT** candidates were made in the random oracle model.

We show how to construct **ERF**'s and **AONT**'s with nearly optimal parameters. Our computational constructions are based on any one-way function. We also provide several applications and additional properties concerning these notions.

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]. For example, at the Rump session of CRYPTO '98, van Someren [23] 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 with high entropy. Within weeks of the appearance of the followup paper [21], a new generation of computer viruses emerged that tried to use these ideas to steal secret keys [8]. Shamir and van Someren [21] gave some heuristic suggestions on preventing these kinds of attacks, but their methods still do not solve the problem of *partial exposure*.

Unfortunately, standard cryptographic definitions and constructions do not guarantee security *even if a tiny fraction of the secret key is exposed*. Indeed, many constructions become provably insecure (the simplest example would be "one-time pad" encryption), while the security of others becomes unclear. In this work, we show how to build cryptographic primitives, in the standard model (i.e., 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 [20, 3] and protection using specialized hardware. 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. Similarly, the use of specially protected hardware (such as smartcards) can also be costly, inconvenient, or inapplicable in many contexts.

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. (Indeed, our techniques provide, for certain parameters, highly efficient computational secret sharing schemes [15], where the size of secret shares can be as small as *one bit*! See Remark 9 in Section 5.1.)

The All-Or-Nothing Transform. Recently Rivest [19], 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**[1] is an efficiently computable transformation T on strings such that:

[1] Here we informally present a refinement of the definition due to Boyko [4].

– For any string x, given (*all* the bits 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.

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 more general **AONT**'s that have a two-part output: a public output that doesn't need to be protected (but is used for inversion), 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. As mentioned above, **AONT** has many other applications, such as enhancing the security of block-ciphers, hash functions and making fixed-blocksize encryption schemes more efficient (e.g., [14, 22]). For an excellent exposition on these and other applications of the **AONT**, see [4].

Our Results. Until now, the only known analysis of an **AONT** candidate was carried out by [4], who showed that Bellare and Rogaway's Optimal Asymmetric Encryption Padding (OAEP) [2] yields an **AONT** *in the Random Oracle model.* However, analysis in the Random Oracle model provides only a limited security guarantee for real-life schemes where the random oracle is replaced with an actual hash function [5].[2] 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 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.* **ERF**'s provide an alternative solution to **AONT** for the partial key exposure problem, since (at least, in principle) we can assume that our secret key is a truly random string R (say, the randomness used to generate the actual secret key). In such a case, we choose and store a random value r and use $f(r)$ (where f is an **ERF**) in place of R. In many settings (such as in private-key cryptography) this alternative is much more efficient than **AONT**. Another application of **ERF**'s is for protecting against gradual key exposure, where no bound on the amount of information the adversary obtains is assumed; instead, we assume only a bound on the *rate* at which that the adversary gains information.
Our main results regarding **ERF**'s and **AONT**'s are summarized as follows.

[2] Though for a much weaker definition, Stinson [25] has given an elegant construction for **AONT** with security analysis in the standard setting. As observed by [4], however, this construction does not achieve the kind of security considered here.

– 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 build an *unconditionally* secure **ERF** whose output of size k is statistically close to uniform provided one misses only $\ell = k + o(k)$ bits of the input. This is optimal up to the lower order term, as no unconditionally secure **ERF**'s exist when $k < \ell$.
– Furthermore, we show that any computationally secure **ERF** with $k < \ell$ implies the existence of one-way functions.
– 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 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 give another construction of an **AONT** based on any length-preserving 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 **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 seemingly 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. [6] 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 [6], by achieving an output distribution that is not truly random, but rather exponentially close in statistical deviation from truly random.

The concern of forward-security (or, protection from the *complete* exposure of past keys) was considered by Diffie et. al. [7] in the context of key exchange, and by Bellare and Miner [1] in the context of signature schemes. These works prevent an adversary that gains current secret keys from being able to decrypt past messages or forge signatures on messages "dated" 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.

Organization. Section 2 briefly defines some preliminaries. Section 3 defines Exposure-Resilient Functions and All-Or-Nothing Transforms. Section 4 talks in detail about constructions and application of **ERF**'s, while Section 5 is concerned with constructing and examining the properties of **AONT**'s.

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. We write $m = poly(k)$ to indicate that m is polynomially bounded in k. 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.

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

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

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

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 (see, for instance, [9]). For the case of statistical closeness, we will always have ϵ negligible in the appropriate security parameter. When the statement can hold for any of the above choices (or the choice is clear from the context), we simply write $A \approx B$.

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 (pseudo) random to the adversary. Formally,

Definition 1. *A polynomial time computable function $f : \{0,1\}^n \to \{0,1\}^k$ 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:*

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

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

Remark 1. Note that this 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. In general, 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 2. *A* randomized *polynomial time computable function* $T : \{0,1\}^k \rightarrow \{0,1\}^s \times \{0,1\}^p$ *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 $y = (y_1, y_2) \in T(x)$, we have $I(y) = x$.*
2. *For any $L \in \{{s \atop \ell}\}$, any $x_0, x_1 \in \{0,1\}^k$ we have*

$$\langle x_0, x_1,\ [T(x_0)]_{\bar{L}} \rangle \approx \langle x_0, x_1,\ [T(x_1)]_{\bar{L}} \rangle \tag{2}$$

In other words, the random variables in $\{[T(x)]_{\bar{L}} \mid x \in \{0,1\}^k\}$ are all indistinguishable from each other. Here $\approx$ can refer to perfect, statistical or computational indistinguishability.

If $T(x) = (y_1, y_2)$, we call y_1 the secret output *and y_2 the* public output *of T. If $p = 0$ (there is no public output), we call T a* secret-only ℓ-**AONT**.

Remark 2. Note again, as in Remark 1, that the definition given here is a "non-adaptive" definition. We stress that all our constructions satisfy the corresponding adaptive definition, as well.

Remark 3. The above definition is "indistinguishability" based. As usual, one can make the equivalent "semantic security" based definition, where the adversary, given $z = [T(x)]_{\bar{L}}$ (where x is picked according to some distribution M), cannot compute β satisfying some relation $\mathcal{R}(x, \beta)$ "significantly better" than without z 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.

Remark 4. The definition given above generalizes and simplifies (because there are no random oracles) the formal definition for secret-only **AONT** given by Boyko [4] (refining an earlier definition of Rivest [19]) in a setting with a random oracle. In particular, while previous definitions were restricted to *secret-only* **AONT**, our definition allows one to split the output y into two sections: a secret part y_1 and a public part y_2. The public part of the output requires *no protection* — that is, it is used only for inversion and can be revealed to the adversary in full. The security guarantee 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 input. We note that our generalized notion of **AONT** solves the problem of partial key exposure and also remains equally applicable to all the other known uses of the secret-only **AONT**. However, we will see that it gives us more flexibility and also allows us to characterize the security of our constructions more precisely.

Boyko [4] showed that, *in the random oracle model*, the following so called "optimal asymmetric encryption padding" (OAEP) construction of [2] is a (secret-only) 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 $T(x;\ r) = \langle u, t\rangle$, where $u = G(r) \oplus x$, $t = H(u) \oplus r$. We note that the inverse $I(u,t) = G(H(u) \oplus t) \oplus u$. No constructions of **AONT** based on standard assumptions were previously known.

Remark 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. We remark that **ERF**'s can be much more efficient that **AONT**'s for the case of (pseudo) random secrets; for example, in the computational setting we can store the value r that is *shorter* than the length of the actual secret $f(r)$, which is impossible to achieve with **AONT**'s due to their invertibility.

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 build statistical **ERF**'s, achieving essentially the best possible parameters and surpassing the impossibility results for perfect **ERF**'s. Finally, we show how to combine this construction with standard pseudorandom generators to construct computational **ERF**'s (from n to k bits) based on any one-way function that achieve any $\ell = \Omega(n^\epsilon)$ and any $k = poly(n)$ (in fact, we show that such **ERF**'s are equivalent to the existence of one-way functions). Our main results are summarized in the following theorem:

Theorem 1. *Assume $\ell \geq n^\epsilon$ (for some $\epsilon > 0$). Then*

1. *There exist statistical ℓ-**ERF**'s $f : \{0,1\}^n \to \{0,1\}^k$ with $k = \ell - o(\ell)$.*
2. *If $\ell < k \leq poly(n)$, computational ℓ-**ERF**'s $f : \{0,1\}^n \to \{0,1\}^k$ exist iff one-way functions exist.*

4.1 Perfect ERF

Here we require that $\langle [r]_{\bar{L}}, f(r)\rangle \equiv \langle [r]_{\bar{L}}, R\rangle$. 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 $[r]_{\bar{L}}$), 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 [6]. 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 [6] 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, using *binary linear error correcting codes* (see [16]), one can construct the following perfect ℓ-**ERF**.

Theorem 2 ([6]). *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 $d \geq n-\ell+1$.*

Applying it to any *asymptotically good* (i.e. $n = O(k)$ and $d = \Omega(n)$) linear code (e.g. the Justesen code), we can get $\ell = (1-\epsilon)n$, $k = \delta n$, where ϵ and δ are (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.

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 $k = \Omega(\ell)$ (in fact, even $\ell - o(\ell)$). Note that this is the best we can hope for (up to constant factors or even the lower order term), 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 [13], where standard 2-universal hash families are shown to be strong extractors.) Much work has been done in developing this area (e.g. [24, 26, 18]). 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 $\mathcal{H}$).

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 observing $(n-\ell)$ bits of r leaves at least $2^{\ell/2}$ equally likely possible values of v (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 family be describable by a very short random string. This requirement is met by the strong extractor of Srinivasan and Zuckerman [24] using the hash families of Naor and Naor [17]. Their results can be summarized as follows:

Lemma 1 ([24]). *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 3. *There exist statistical ℓ-**ERF** $f : \{0,1\}^n \to \{0,1\}^k$ with $k = \Omega(\ell)$ and statistical deviation $2^{-\Omega(\ell)}$, for any ℓ and n satisfying $\omega(\log n) \le \ell \le n$.*

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 (here we use $\ell = \omega(\log n)$), which describes some hash function in $\mathcal{H}$ mapping $\{0,1\}^n$ to $\{0,1\}^k$ as given in Lemma 1. 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 $[r]_{\bar{L}}$ 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

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

Thus, by Lemma 1, we have that $\langle [r]_{\bar{L}}, h, f(r) \rangle = \langle [r]_{\bar{L}}, h, h(r') \rangle \cong_\epsilon \langle [r]_{\bar{L}}, h, R \rangle$, where R is the uniform distribution on $\{0,1\}^k$. This implies $\langle [r]_{\bar{L}}, f(r) \rangle \cong_\epsilon \langle [r]_{\bar{L}}, R \rangle$, completing the proof.

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

Remark 6. Note that, in particular, we can choose ℓ to be anything super-logarithmic is n (e.g., n^ϵ for any $\epsilon > 0$), providing excellent security against partial key exposure. Seen another way, we can choose n to be essentially any size larger than ℓ.

Remark 7. The output size of our construction can be substantially improved by using recent strong extractors of [18]. In particular, we can achieve $k = \ell - o(\ell)$, provided $\ell = \omega(\log^3 n)$, or $k = (1 - \delta)\ell$ (for any $\delta > 0$), provided $\ell = \omega(\log^2 n)$. In both cases the statistical deviation can be made exponentially small in ℓ. As k must be less than ℓ, this is essentially optimal.

4.3 Computational ERF

The only limiting factor of the statistical construction is that the output size is limitted to $k < \ell$. By finally relaxing our requirement to *computational* security, we are able to achieve an arbitrary output size, by using a pseudorandom generator (**PRG**) as the final outermost layer of our construction. We also show that any **ERF** with $k > \ell$ implies the existence of **PRG**'s (and thus, one-way functions), closing the loop. The proof of the following is straightforward, and therefore omitted:

Lemma 2. *Let $m, n = poly(k)$, $f : \{0,1\}^n \to \{0,1\}^k$ be a statistical ℓ-**ERF** (with negligible ϵ) and $G : \{0,1\}^k \to \{0,1\}^m$ be a **PRG**. Then $g : \{0,1\}^n \to \{0,1\}^m$ mapping $x \mapsto G(f(x))$ is a computational ℓ-**ERF**.*

Theorem 4. *Assume one-way functions exist. Then for any ℓ, any $n = poly(\ell)$ and $k = poly(n)$, there exists a computational ℓ-**ERF** mapping $\{0,1\}^n$ to $\{0,1\}^k$.*

Proof. Since $k = poly(\ell)$, one-way functions imply [12] the existence of a **PRG** $G : \{0,1\}^{\ell/10} \to \{0,1\}^k$. Theorem 3 implies the existence of a statistical ℓ-**ERF** f from $\{0,1\}^n$ to $\{0,1\}^{\ell/10}$ with negligible statistical deviation $2^{-\Omega(\ell)}$. By Lemma 2, $g(r) = G(f(r))$ is the desired computational ℓ-**ERF**.

Lemma 3. *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. The hypothesis implies the existence of the ensemble of distributions $A = \langle [r]_{\bar{L}}, f(r) \rangle$ and $B = \langle [r]_{\bar{L}}, R \rangle$, 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 [10], 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.

Theorem 1 now follows from Remark 7, Theorem 4 and Lemma 3.

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 applications.

As an immediate application, 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 application, 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 applications, 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**) [11]. Let $\mathcal{F} = \{F_s \mid s \in \{0,1\}^k\}$ be a regular **PRF** family. 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 apply 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 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 an 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 shared random keys. Namely, consider a situation where the adversary is able to learn more and more bits of the secret key over time. We do not place any upper bound on the *amount* of information the adversary learns, 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[3]. To avoid ever changing the secret key, both parties periodically (say, with period slightly less than $(k-\ell)/b$ weeks) update their 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, parties agree on the secret key *only once*, even if the adversary continuously learns more and more of the (current) secret!

5 All-Or-Nothing Transform (AONT)

As we pointed out, no **AONT** constructions with analysis outside the random oracle model were 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 construction 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 it suffices to design **AONT**'s that are secure only for *random* x_0, x_1.

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. Thus, we obtain the following theorem, whose proof is omitted due to space constraints:

Theorem 5. *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 $f(r) \oplus x$.*

Notice that the size of the secret part $s = n$ and size of the public part $p = k$. As an immediate corollary of Theorems 1 and 5, we have:

[3] We assume that our **ERF** is secure against *adaptive* key exposure, but our construction achieves this.

Theorem 6. *Assume $\ell \le s \le poly(\ell)$. There exist functions $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^k$ (with secret output of length s and public output of length k) such that*

1. *T is statistical ℓ-**AONT** with $k = \ell - o(\ell)$, or*
2. *T is computational ℓ-**AONT** with $\ell < k \le poly(s)$.*

For example, we could set $\ell = s^\epsilon$ to have excellent exposure-resilience. The computational construction also allows us to have essentially any input size k we want (as long as it is polynomial in s), and have the total output size $N = s+k$ be dominated by k, which is close to optimal. A reasonable setting seems to be $s = o(k)$ (i.e., just slightly smaller than k) and $\ell = s^\epsilon$.

Remark 8. 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, it must miss at least ℓ bits of the secret output). Applying this to our construction (where $p = k$), we see that $\ell' = \ell + k$ and we can achieve essentially any $N > \ell'$. In particular, we can still have excellent exposure-resilience $\ell' = N^\epsilon$, but now the output size $N = (\ell')^{1/\epsilon} > k^{1/\epsilon}$ is large compared to the input length k. See Section 5.3 for a possible solution to this problem. We also notice that we can have $\ell' = 2k+o(k) = O(k)$ (and essentially any N) even in the statistical setting.

Remark 9. 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* [15]. 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 essentially any value of s larger than ℓ (plus essentially any length k of the secret).

5.2 AONT implies OWFs

Theorem 7. *Assume we have a computational ℓ-**AONT** $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^p$ 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 [9]. Fix $L = [\ell] \subseteq [s]$. Define $g(x_0, x_1, b, r) = \langle x_0, x_1, [y]_{\bar{L}} \rangle$, 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, $y = T(x_b;\ r)$, $z = [y]_{\bar{L}}$, $\langle \tilde{b}, \tilde{r} \rangle = A(x_0, x_1, z)$, $\tilde{y} = T(x_{\tilde{b}};\ \tilde{r})$, $\tilde{z} = [\tilde{y}]_{\bar{L}}$, 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* $x_0, x_1 \in \{0,1\}^k$. Hence, the job of F is the following. x_0, x_1, b, r are chosen at random, and we set $y = T(x_b;\ r)$, $z = [y]_{\bar{L}}$. 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 $\tilde{y} = T(x_{\tilde{b}};\ \tilde{r})$, $\tilde{z} = [\tilde{y}]_{\bar{L}}$. 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, $[T(x_b;\ r)]_{\bar{L}} \in [T(x_{1-b})]_{\bar{L}}$, i.e. there exists some r' such that $[T(x_{1-b};\ r')]_{\bar{L}} = z$ 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, observe that U measures the probability of the event that when we choose x, x', r at random and set $z = [T(x;\ r)]_{\bar{L}}$, there is r' such that $z = [T(x';\ r')]_{\bar{L}}$. However, for any fixed setting of z, there are only 2^ℓ possible completions $y \in \{0,1\}^{s+p}$. And for each such completion y, invertibility 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.

We note that the result is essentially optimal (up to the lower order term), since by Theorem 6 there are statistical **AONT**'s with $\ell = k + o(k)$. In fact, merging the secret and public parts of such an ℓ-**AONT** (the latter having length k) gives a statistical *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. The construction has the advantage of achieving *secret-only* **AONT**'s, while retaining a relatively short output length.

Recall that the OAEP construction of [2] 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. Let $f : \{0,1\}^k \to \{0,1\}^k$, define $T(x;\ r) = \langle f(r) \oplus x, (f(r) \oplus r) \oplus x\rangle$, and note that the inverse is $I(u,t) = u \oplus f(u \oplus t)$. Due to space limitations, we omit the proof of the following:

Theorem 8. *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**.*

We note, that random oracle f clearly satisfies the conditions of the Theorem. Thus, our analysis makes a step towards abstracting the properties of the random oracle needed to make the OAEP work as an **AONT**. We believe that the assumption of the theorem is quite reasonable, even though 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 $q(x_0, x_1)$. 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**.[4] 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. We omit the proof of the following due to space constraints:

Lemma 4. *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 outputs of T. In particular, if T is secret-only, then so is T'.

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 (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 and **AONT**'s (in the computational setting, based on any one-way function). We also explored many other interesting properties of **ERF**'s and **AONT**'s.

Acknowledgements. We would like to thank Madhu Sudan for several helpful discussions. Much of this work was performed while all authors were at the IBM T.J. Watson Research Center. Amit Sahai's research was also supported in part by a DOD NDSEG Fellowship.

[4] Note, for instance, the proof of Theorem 7 works for average-case **AONT**'s as well.

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. G. Blackley. Safeguarding Cryptographic Keys. In *Proc. of AFIPS 1979 National Computer Conference*, 1979.
4. V. Boyko. On the Security Properties of the OAEP as an All-or-Nothing Transform. In *Proc. of Crypto*, pp. 503–518, 1999.
5. R. Canetti, O. Goldreich and S. Halevi. The Random-Oracle Model, Revisited. In *Proc. of STOC*, pp. 209–218, 1998.
6. 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.
7. W. Diffie, P. van Oorschot and M. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2:107–125, 1992.
8. 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.
9. O. Goldreich. Foundations of Cryptography (Fragments of a Book). Available at `http://www.wisdom.weizmann.ac.il/home/oded/public_html/frag.html`
10. O. Goldreich. A Note on Computational Indistinguishability. In *IPL*, 34:277–281, 1990.
11. O. Goldreich, S. Goldwasser and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):210–217, 1986.
12. J. Hastad, R. Impagliazzo, L. Levin, M. Luby. A Pseudorandom generator from any one-way function. In *Proc. of STOC*, 1989.
13. R. Impagliazzo, L. Levin, M. Luby. Pseudorandom Generation from one-way functions. In *Proc. of STOC*, pp. 12–24, 1989.
14. M. Jakobsson, J. Stern, M. Yung. Scramble All, Encrypt Small. In *Proc. of Fast Software Encryption*, pp. 95–111, 1999.
15. H. Krawczyk. Secret Sharing Made Short. In *Proc. of Crypto*, pp. 136–146, 1993.
16. F. MacWilliams, J. Sloane. Theory of Error-Correcting Codes, Amsterdam, 1981.
17. J. Naor, M. Naor. Small-Bias Probability Spaces: Efficient Constructions and Applications. In *SIAM J. Computing*, 22(4):838-856, 1993.
18. R. Raz, O. Reingold, S. Vadhan. Error Reduction for Extractors. In Proc. of FOCS, pp. 191–201, 1999.
19. R. Rivest. All-or-Nothing Encryption and the Package Transform. In *Fast Software Encryption, LNCS*, 1267:210–218, 1997.
20. A. Shamir. How to share a secret. In *Communic. of the ACM*, 22:612-613, 1979.
21. A. Shamir, N. van Someren. Playing “hide and seek” with stored keys. In *Proc. of Financial Cryptography*, 1999.
22. S. U. Shin, K. H. Rhee. Hash functions and the MAC using all-or-nothing property. In *Proc. of Public Key Cryptography, LNCS*, 1560:263–275, 1999.
23. N. van Someren. How not to authenticate code. Crypto'98 Rump Session, Santa Barbara, 1998.
24. A. Srinivasan, D. Zuckerman. Computing with Very Weak Random Sources. In *Proc. of FOCS*, pp. 264–275, 1994.
25. D. Stinson. Something About All or Nothing (Transforms). Available from `http://cacr.math.uwaterloo.ca/~dstinson/papers/AON.ps`, 1999.
26. L. Trevisan. Construction of Extractors Using PseudoRandom Generators. In Proc. of STOC, pp. 141–148, 1999.