Exposure-Resilient Cryptography

by

Yevgeniy Dodis

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2000

Author .....
Department of Electrical Engineering and Computer Science
August 25, 2000

Certified by .....
Madhu Sudan
Associate Professor
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

Exposure-Resilient Cryptography

by

Yevgeniy Dodis

Submitted to the Department of Electrical Engineering and Computer Science
on August 25, 2000, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

We develop the notion of *Exposure-Resilient Cryptography*. While standard cryptographic definitions and constructions do not guarantee any security even if a tiny fraction of the secret entity (e.g., cryptographic key) is compromised, the objective of Exposure-Resilient Cryptography is to build information structures such that *almost complete* (intentional or unintentional) exposure of such a structure still protects the secret information embedded in this structure.

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 exposure of secrets in the setting where the secret is simply a random value, like in the private-key cryptography. They can also be viewed as very secure pseudorandom generators and have many other applications.

To solve the general partial exposure of secrets, we use the (generalized) notion of an *All-Or-Nothing Transform* (AONT) introduced by Rivest [51] and refined by Boyko [16]: 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 entity (of arbitrary structure), we obtain security against almost total exposure of secrets. AONT's have also many other diverse applications in the design of block ciphers, secret sharing and secure communication. To date, however, the only known analyses of AONT candidates were made in the random oracle model (by Boyko [16]).

In this thesis we construct ERF's and AONT's with nearly optimal parameters in the standard model (without random oracles), in the perfect, statistical and computational settings (the latter based only on one-way functions). We also show close relationship between and examine many additional properties of what we hope will become important cryptographic primitives — Exposure-Resilient Functions and All-Or-Nothing Transforms.

Thesis Supervisor: Madhu Sudan
Title: Associate Professor

Acknowledgments

A lot of people have contributed to this thesis, and made my entire stay at MIT as enjoyable and fulfilling as it has been. Most of all, I would like to thank my parents for their great love and support in all stages of my life. I would also like to thank my co-authors Ran Canetti, Shai Halevi, Eyal Kushilevitz, Amit Sahai and Adam Smith for their collaboration on various stages of my research that has led to this thesis. In particular, most of this thesis is based on the papers [17, 24].

Next, I would like to thank my advisor Madhu Sudan for having his door always open and for helping me to direct my research and to stay focused. Then, I would like to express great gratitude to Silvio Micali, who literally started me as a cryptographer "over lunch" which turned into my first cryptographic paper. Silvio also helped me with a great piece advice in many-many situations, and collaborating with him was a true pleasure. Many thanks to Ron Rivest and Shafi Goldwasser, who are the founding members of the cryptography group at MIT, and from whom I learned a lot. In addition, Shafi has taught my first course in Cryptography, and Ron was the one who invented All-Or-Nothing Transforms [51]. I would also like to thank Victor Boyko, who was first to put the notion of the All-or-Nothing Transform to the formal level [16], and whose great work was the true starting point of this thesis. I had several very helpful discussions with Victor, who was also kind enough to let me use some of his figures in this thesis.

I also had a lot of truly illuminating and fun discussions with some of the fellow students at MIT. Mainly, I would like to thank Salil Vadhan, who always had time for my "simple questions", some of which turned into several hour discussions. In addition, Salil together with Luca Trevisan were the ones who introduced me to the concept of extractors, which became crucial building blocks in several constructions in this thesis. In particular, Salil suggested to me to slightly generalize his paper with Luca [62] and to formally define δ-sure extractors, which has led to the construction of adaptively secure Exposure-Resilient Functions. I also had a lot of great conversations with Venkatesan Guruswami, who also helped me on some of the coding theory stuff.

It was also a lot of fun working with Amit Sahai. Somehow, Amit's mere presence has stimulated a lot of ideas to come out of my head, some of which became important pieces of this thesis. Amit also possesses some great managerial skills and a good sense of humor, contrary to some adversarial beliefs. Special thanks to Eric Lehman, who shares the same love to "stupid puzzles" as I do. As a result, we wasted a lot of time solving such puzzles, but had great fun doing so. I would also like to thank some other great students I met at MIT: Anna Lysyanskaya for many fun times, Sofya Raskhodnikova for being a great friend, Mathias Ruhl for collaborating on finding a simple proof that semantic security is equivalent to indistinguishability, Adam Smith for his enthusiasm which carried over to me, and Leo Reyzin and Stanislaw Jarecki for many fun conversations. Finally, let me thank Dan Spielman who helped to greatly simplify the presentation of the lower bound on perfect All-Or-Nothing Transforms.

Another person who was extremely useful in starting my research career was Sanjeev Khanna. Sanjeev put a lot of time into me, gave me a lot of great advice, and has taught me many things, especially in algorithms and optimization. Two summers at Bell Labs under his supervision will always be a fond memory. I would also like to thank the IBM T.J. Watson Research Center, and in particular Tal Rabin and her entire wonderful cryptography group. The summer spent in this group was truly fantastic, and the research eventually leading to this thesis was started there (from a casual comment by Eyal Kushilevitz). Finally, thank you, MIT, for the great time I had here![1]

[1] I am sure there are a lot of people whom I forgot to mention. Please, forgive me, but thanks!!! you were great!!!

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Our Results

Secret Keys. In very general terms, cryptography can be defined as a branch of computer science aimed to protect the disclosure of secret information to unauthorized parties. The exact meaning of "disclosure", "secret information", "unauthorized parties" and many other related terms varies dramatically from application to application, and crucially depends on the exact cryptographic model we use to abstract the reality. However, most cryptographic models can be described in terms of the following pieces (only the first of which will be relevant to the subsequent discussion):

1. A bunch of secret entities (usually called *keys*) known only to "legitimate users". We notice that we do not restrict our attention to so called "cryptographic" keys, like secret keys for encryption, signatures, identification, etc. For example, a "secret key" can be a confidential document, a secret technology, a patent, a piece of proprietary software, a copyrighted audio/video recording, a database of employee salaries, records of financial transactions, etc. For the lack of a better term, all of the above secret entities will be called "secret keys". For simplicity, we will also assume that there is only a single key that needs to be kept secret.

2. The desired functionality of the system (by legitimate users).

3. The (often somehow limited) capabilities of the "illegitimate users" (typically

assumed to be coordinated by a single entity, called the *adversary*).

4. Finally, the security claim we can make about our system.

In the above generic description, the thing concerning us the most will be the implicit, but nevertheless *fundamental* assumption that the secret key has to be kept *completely hidden* from the adversary.[1] This assumption is so basic and so "obviously needed" for any reasonable notion of security, that one may wonder why to even bring it up. But what happens if this most basic assumption breaks down?

The Problem of Key Exposure. Namely, *what happens if the secrecy of our key becomes (partially) compromised?* After a brief initial surprise of being asked such an obvious question, the equally obvious answers (stated in the order of information content) would be:

- "I don't know".
- "Well... make sure it does not".
- "This is outside the model of conventional cryptography".
- "Good luck..." (meaning "you are doomed", as the adversary knows the same information as the legitimate user).

While these (reasonable) answers might suggest that this is a strange question to ask, it has been noted that key exposure is one of the greatest threats to security in practice (see [7]). For a concrete recent example, at the Rump session of CRYPTO '98 van Someren [58] 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 [55], a new generation of computer viruses emerged that tried to use these ideas to steal secret keys [25]. More abstractly, one can imagine very sophisticated attacks to break

[1] Aside from the information that the adversary can get from his "legal interaction" with the system. But this is taken into account when defining the security of the system.

the security of a given system, but getting the secret key, if possible, would be the most trivial way to completely demolish any security claim![2]

PREVIOUS SOLUTIONS. The most widely considered solutions to the problem of key exposure are distribution of keys across multiple servers via secret sharing [54, 38, 13] and protection using specialized hardware. Instantiations of the key distribution paradigm include threshold cryptosystems [22] and proactive cryptosystems [35]. 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.

Another approach to the problem of key exposure is that of *forward-security* (or, protection from the exposure of "past" keys) considered by Diffie et al. [23] in the context of key exchange, and by Anderson [3], Bellare and Miner [7] and Abdalla and Reyzin [2] in the context of signature schemes. In these works the secret key is being dynamically updated (without affecting the public information). The objective is to prevent an adversary that gains *current* secret keys from being able to decrypt past messages or forge signatures on messages "dated" in the *past*. Inevitably, however, the system can no longer be used in the future once the current keys have been exposed.

PARTIAL KEY EXPOSURE. As we pointed out, secrecy of keys is a fundamental assumption of conventional cryptography. While partial solutions exist, not much can be done since the adversary has the same information as the legitimate user after the secret has been exposed. Instead, we will look at a slight relaxation of this question, which looks considerably more hopeful. Namely, we assume that out secret is not *completely* exposed. Rather, the adversary learns *most*, but not *all* of the secret.

For example, imagine using a smartcard to protect our key. While it is quite reasonable to assume that the smartcard is tamper-resistant enough not to leak the entire key, it might be a bit too dangerous to be confident that not even a small

[2] As a famous Russian philosopher Koz'ma Prutkov said: "Zri v koren' " ("look into the root"). Two common interpretations of this amazingly deep phrase are "seek the obvious" and "get to the bottom of things". See [47] for more information about Koz'ma Prutkov.

part of the key can be extracted. Or imagine sending a sensitive information over some communication channel, which is believed to be secure (so that no encryption is performed, or the parties did not have a chance to exchange keys yet). However, the adversary manages to partially break into the channel and overhear some portion of the communication. Alternatively, the channel is known to be reliable for the authorized parties, and is known to be somewhat noisy to the adversary. While encryption would be a good solution, it could be an overkill since we can exploit the noise introduced to the adversary. Another situation would be when the adversary is trying to copy a large confidential document, but the intrusion detection system cut the transmission in the middle. Yet another example would be a large file copied to several floppy disks (it is too large to fit onto one disk), and one of these disks being lost, stolen or copied.

In the same vein, we may purposely (e.g., for security reasons) split the key into physical shares (rather than using space-inefficient conventional secret sharing schemes), and to store these shares in different parts of memory (or even on different machines). But then we cannot in general argue the security since leaking even one physical share may make the underlying application insecure. It would be nice to find a way to make this simple approach work. Alternatively, a natural idea would be to use a conventional 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). Hence, 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.[3]

Exposure-Resilient Cryptography. In fact, standard cryptographic definitions and constructions do not guarantee security *even if a tiny fraction of the secret key is*

[3]Indeed, our techniques provide, for certain parameters, highly efficient "gap" secret sharing schemes, where the size of secret shares can be as small as *one bit*! Inevitably, however, there is a gap between the number of people who can reconstruct the secret and the number of people who "get no information" about the secret.

exposed. Indeed, many constructions become provably insecure (the simplest example would be the "one-time pad" encryption), while the security of others becomes unclear (and a complete mess to verify!). In other words, conventional cryptographic systems are not (and should not be!) designed so as to tolerate partial key exposure. In this thesis we develop the notion of *Exposure-Resilient Cryptography*, one of whose main goal is to identify and to build general cryptographic primitives which are oblivious to the cryptographic system we are using, but can make any such system *provably* secure against *almost total* key exposure. More generally than this, the objective of Exposure-Resilient Cryptography will be to *build information structures such that almost complete (intentional or unintentional) exposure of such a structure still protects certain secret information embedded in this structure*. In particular, once we define more precisely the cryptographic primitives we develop, these primitives will prove very useful in many applications beyond the problem of partial key exposure. These applications include secret sharing, secure communication, secret-key exchange, more secure and efficient block ciphers, remotely keyed encryption, coin-flipping, fair information exchange and others (see Section 3). In other words, the techniques for solving the problem of partial key exposure will prove useful in may other areas, making Exposure-Resilient Cryptography a general useful tool.

Without further delay, we are now ready to introduce the main primitives for Exposure-Resilient Cryptography: *All-Or-Nothing Transforms* and *Exposure-Resilient Functions*.

ALL-OR-NOTHING TRANSFORMS. Recently Rivest [51], motivated by different security concerns arising in the context of block ciphers, introduced an intriguing primitive called the *All-Or-Nothing Transform (*AONT*)*. Rivest's work was refined and extended by Boyko [16], whose definition we informally present below. An AONT is an efficiently computable randomized transformation T on strings such that:

- 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.

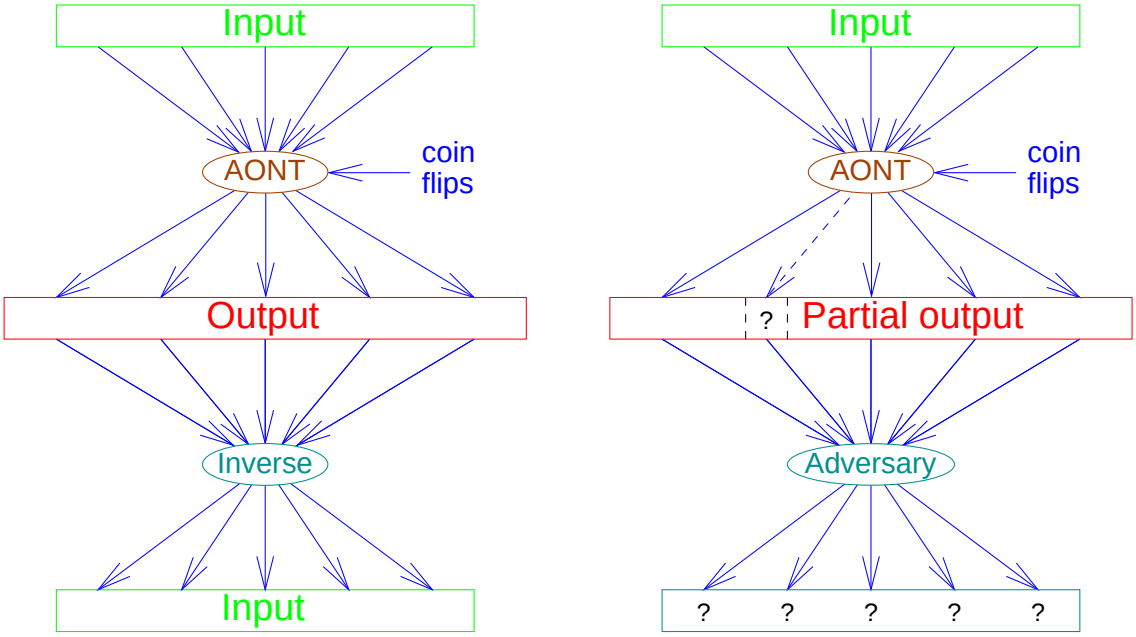

Figure 1-1: All-Or-Nothing Transform.

This is informally illustrated in Figure 1-1.

We observe that 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. We 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, suffices for all known applications of AONT, and allows us much more flexibility. Thus, we refer to the traditional notion of AONT as *secret-only*. As mentioned above, AONT has many other applications, such as enhancing the security of block-ciphers [51, 21, 16], hash functions [57], secure communication [10], making fixed-blocksize encryption schemes more efficient [37, 41, 4], gap secret sharing schemes [51, 17] and others [52, 16]. We will survey these and other applications later.

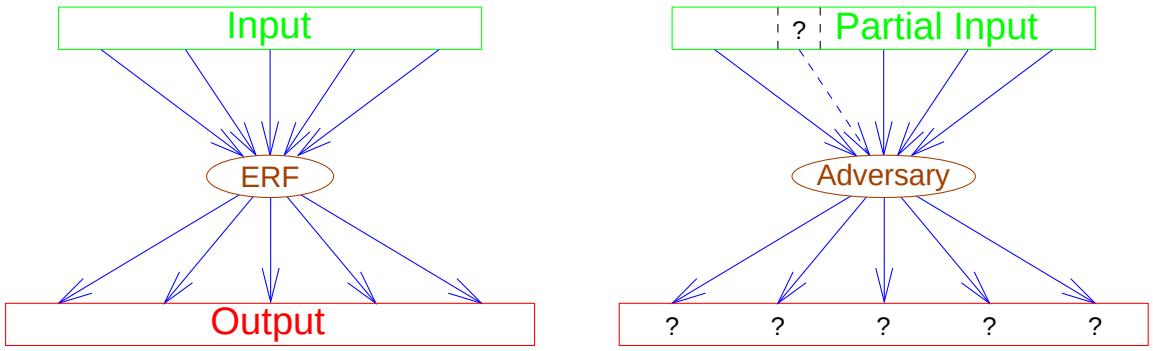

Figure 1-2: Exposure-Resilient Function.

EXPOSURE-RESILIENT FUNCTIONS. The key to our approach and our main conceptual contribution is the new 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. This is informally illustrated in Figure 1-2.

We demonstrate that the notion of ERF is 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*. 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 ERF(r) 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. We will later show other applications of exposure-resilient functions.

OUR RESULTS. We give natural and simple definitions for ERF's and AONT's in the perfect, statistical and computational settings (i.e., achieving ideal, unconditional with a negligible error and computational security, respectively). We then derive

essentially optimal results concerning ERF's and AONT's in each of these settings. These results can be briefly summarized as follows.

- Limitations of Perfect ERF's and AONT's. We show that perfect ERF's imply perfect AONT's with the "same parameters". Unfortunately, we prove that perfect AONT's (and thus ERF's) have very strong combinatorial limitations. In essence, even if we allow exponential size output of the AONT, the enemy must miss *at least half of the output* in order to not learn anything about the input! The result is interesting in its own right and shows the impossibility of certain "balanced" colorings of the hypercube. It also generalizes the lower bound of Friedman [28] and further settles the conjectures of Chor et al. [20].

- Construction of Perfect ERF's and AONT's. On a positive side, we can almost match our lower bound (for both ERF's and AONT's) with a general construction of [20, 10] that uses linear binary error-correcting codes.

- Optimal Statistical ERF. 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 (whose size can be arbitrarily large compared to ℓ). This is optimal up to the lower order term, since we show that no unconditionally secure ERF's exist when $k < \ell$. Thus, statistical ERF's can achieve much better exposure-resilience than perfect ERF's, and their only limitation is the limited output size (at most ℓ). This statistical construction is one of our main technical contribution, and it uses very powerful combinatorial objects called *strong randomness extractors*.

- Computational ERF's $\Rightarrow$ one-way functions. Furthermore, we show that any computationally secure ERF with $k < \ell$ implies the existence of one-way functions, which is nearly the best we can hope to show due to the unconditional construction above.

- Computational ERF's from one-way functions. We show how to construct, from any one-way function, for any $\varepsilon > 0$, 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. This can be viewed as an extremely strong pseudorandom generator, and shows that we can achieve essentially any conceivable setting of parameters in the computational setting.

- AONT'S FROM ERF'S. We give a simple "universal" construction of an AONT based on any ERF, which works in any setting (in particular, statistical and computational). Moreover, when used with the best ERF's in the corresponding setting, we get nearly optimal AONT's, as we explain below.

- OPTIMAL STATISTICAL AONT'S. In the statistical setting, we get an AONT with resilience $\ell = k + o(k)$ (where k is the size of the input, and the secret part can be arbitrarily large compared to ℓ), which is optimal up to the lower order term since we show that $\ell \geq k$ for any statistical AONT. In fact, we can even get a *secret-only* AONT with $\ell = O(k)$ still. Again, these results dramatically beat our impossibility result for perfect AONT's, and show a large gap in exposure-resilience between the perfect and the statistical settings.

- COMPUTATIONAL AONT'S $\Rightarrow$ ONE-WAY FUNCTIONS. Furthermore, the existence of computational AONT with $\ell < k$, where k is the size of the input, implies the existence of one-way functions. This is nearly the best we can hope to show due to the statistical construction above.

- COMPUTATIONAL AONT'S FROM ANY ONE-WAY FUNCTION. If k is the length of the input, we get a public output of length k, a secret output of essentially arbitrary size s, and achieve resilience $\ell = s^\epsilon$ (for any $\epsilon > 0$). For example, setting $s = k$ we can achieve nearly optimal total output size $2k$, secret and public parts of size k and very good resilience $\ell = k^\epsilon$.

- TOWARDS SECRET-ONLY AONT. We give another construction of a secret-only 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

construction of Bellare and Rogaway [8] (which was shown to be an AONT in the *random oracle model*[4] by Boyko [16]), 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 being *secret-only* (without separate public and secret outputs) while retaining a relatively short output length.

- Worst-case/Average-case AONT's. We also show a structural result that a seemingly weaker "average-case" definition of AONT is almost equivalent to the standard "worst-case" definition of AONT, by giving an efficient transformation that achieves this goal.

- Adaptively Secure ERF's and AONT's. Finally, we consider the notion of *adaptively* secure ERF's and AONT's. Contrary to the "non-adaptive" notions we discussed above, where the adversary decides in advance which bits of the stored secret he is going to observe (as long as he misses ℓ bits), here we allow the adversary to access the secret adaptively "one-bit-at-a-time", i.e. to base its decision of which bits to read depending on the information that he gathered so far. We call the ERF's and AONT's resilient against such adversaries *adaptively secure*. It turns out that it is significantly more challenging to build adaptively secure ERF's and AONT's. In particular, some of our "non-adaptive" constructions above do not work against adaptive adversaries. Based on the ideas of Trevisan and Vadhan [62], we overcome these difficulties and give efficient *probabilistic* constructions of adaptively secure ERF's and AONT's with essentially the same (and even slightly better) parameters than in the regular non-adaptive setting.

To reiterate our results, we show that perfect AONT's and ERF's, while conceptually attractive, cannot achieve the exposure-resilience we ultimately desire. On the other hand, statistical ERF's and AONT's can achieve excellent exposure-resilience. However, they are limited in terms of requiring that the adversary misses at least as

[4] In this idealized model all the participants have public access to a certified truly random function.

many bits as the amount of information or randomness we are trying to hide. Finally, we show that in the computational setting we can overcome even this limitation and achieve essentially any desirable setting of parameters — all based only on the existence of one-way functions. In fact, for "interesting" settings of parameters, computational ERF's, AONT's and one-way functions are "equivalent". Finally, we show that all the above results and implications can be extended to the *adaptive setting*, except our main constructions become probabilistic.

In addition to the above results, we examine many additional properties and applications of what we hope will become important cryptographic primitives — Exposure-Resilient Functions and All-Or-Nothing Transforms.

Previous Work. Until this work, the only known analysis of an AONT candidate was carried out by Boyko [16],[5] who showed that Bellare and Rogaway's Optimal Asymmetric Encryption Padding (OAEP) [8] yields an AONT *in the Random Oracle model.* Boyko's work was the first formal treatment of the AONT, stimulated a lot of subsequent research and achieved essentially the best possible AONT's 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 [18]. Subsequent to our work, Desai [21] gave another provable construction of an AONT (based on the original informal construction of Rivest [51]) and analyzed it in the so called "ideal cipher model".[6] This construction also achieves a somewhat weaker security notion than the one we consider here, even though this notion is strong enough for several important applications of the AONT. Thus, our work gives the *first provable constructions* for AONT's with essentially optimal resilience in the standard model, based either on no assumptions, or only on the minimal computational assumption that one-way functions exist.

Vazirani [63] defined a notion later called a *t-resilient function*, which turns out to

[5] Though for a much weaker definition of security than the one we study here, Stinson [60] has given an elegant construction for AONT with security analysis in the standard setting. As observed by [16], however, this construction does not achieve the kind of security considered here.

[6] I.e., all the participants have access to a keyed family of independent random permutations.

be equivalent to our notion of perfect ERF's.[7] A t-resilient function is a function whose output is *truly* random even if the adversary can *fix* any t of the inputs to the function. Chor et al. [20] and, independently, Bennett et al. [10] considered this notion in a much greater detail. In particular, a very nice construction for t-resilient functions was given by [20, 10] based on error-correcting codes. We use this construction when talking about perfect ERF's, and then extend it to constructing perfect AONT's, as was also implicitly done by [10]. Chor et al. [20] gave some initial impossibility results for t-resilient functions (which, as we said, are equivalent to perfect ERF's) and conjectured that much more general impossibility results hold. In particular (viewed in terms of ERF's), the adversary must essentially miss at least half of the input bits in order for the output to be random (which was the fundamental limitation of their coding theory construction that we mentioned). This conjecture stood for some time and was finally affirmatively resolved by Friedman [28] (another proof was later given by [11]). Our impossibility result for perfect AONT's non-trivially extends this conjecture (since we show that perfect ERF's imply perfect AONT's) and subsumes the results of [28, 11], whose techniques do not apply to our more general setting.

Kurosawa et al. [39] considered a slightly relaxed notion of *almost t-resilient functions*. An almost t-resilient function is a function whose output is "very close" to uniform even if the adversary can fix any t of the inputs to the function. As we will see, this notion stands somewhere "in between" the notions of perfect and statistical ERF's, and turns out to be essentially equivalent to our notion of *adaptively secure* statistical ERF.[8] Kurosawa et al. somewhat improved the parameters achieved by [20] in constructing (regular) t-resilient functions, but their construction still requires the adversary to fix *at most half* of the input bits. While the considerable complexity of this construction, coupled with the pessimistic parameters it achieves, might suggest that almost t-resilient functions share the same strong limitations as regular t-resilient functions (i.e., perfect ERF's), we will show that this is *not* the case. More specifi-

[7] If n is the size of the input and $\ell = n - t$, then t-resilient function is the same as ℓ-ERF.

[8] Almost t-resilient functions are slightly stricter, but our construction of adaptively secure statistical ERF's will actually achieve it.

cally, using our construction of adaptively secure statistical ERF's we will allow the adversary to fix $t \approx (n-k)$ input bits, where n is the size of the input and k is the size of the output, which is easily seen to be the best possible. Even though our construction is probabilistic (contrary to that of [39]), it succeeds with overwhelming probability and shows that almost t-resilient functions are much more powerful than regular t-resilient functions.

Finally, we already mentioned the works of [23, 3, 7, 2, 1] on forward-security. 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 OF THE THESIS. In Chapter 2 we define some preliminaries and some general results that we will use. In particular, we will examine the notions of semantic security and indistinguishability, define some important cryptographic basics, like one-way functions and pseudorandom generators, talk about error-correcting codes, introduce randomness extractors and t-wise independent functions, and state some basic facts from linear algebra and Fourier analysis. Some of the results are new and of independent interest. For example, we give a simple generic proof that semantic security is equivalent to indistinguishability, substantially simplifying (albeit for a slightly weaker but equally natural definition of semantic security) the original proof of Goldwasser and Micali [33].

In Chapter 3 we formally define our main gadgets: Exposure-Resilient Functions and All-Or-Nothing Transforms. We give simple definitions in the perfect, statistical and computational settings, and also distinguish between non-adaptive and adaptive ERF's and AONT's. We then compare ERF's and AONT's with each other and with some other fundamental notions like pseudorandom generators and error-correcting codes. We also talk in detail about many applications of ERF's and AONT's, some of which are new.

Chapter 4 talks in detail about constructions and limitations of Exposure-Resilient

Functions. We start with the perfect setting, where we use the construction of [20, 10] via error-correcting codes, and the lower bound of Friedman [28] to show that our construction is tight. Then we move to the statistical setting, and show how to use randomness extractors to obtain an efficient and nearly optimal construction of ERF's, which is one of our main contributions. Next we move to the computational setting and show how to combine pseudorandom generators with our statistical construction to get optimal computational ERF's. Finally, we move to the adaptive setting and observe that our statistical construction of regular ERF's is not adaptively secure. However, we give a probabilistic construction of adaptively secure ERF's achieving the same optimal parameters as our non-adaptive construction.

In Chapter 5 we construct and examine the properties of All-Or-Nothing Transforms. A large part of this chapter will be devoted to perfect AONT's. In particular, to comparing them with perfect ERF's and proving the lower bound on perfect AONT's (which extends the lower bound of Friedman [28] on perfect ERF's). We will then give a simple construction of AONT's using ERF's, which will yield essentially optimal AONT's. Next we will suggest a secret-only AONT construction which is a special case of the OAEP construction of [8], which may serve as the first step in abstracting the properties of the random oracles that make OAEP an AONT. After that we give a surprisingly non-trivial proof that AONT's with "interesting" parameters imply one-way functions, which, combined with the previous results, shows that "interesting" computational ERF's, AONT's and one-way functions are all equivalent. The chapter concludes with a structural result showing the "worst-case/average-case" equivalence of AONT's.

Finally, Chapter 6 has some concluding thoughts.

Chapter 2

Preliminaries

How to read this chapter. This chapter tuned out to be somewhat longer and more detailed than was originally planned. In fact, some of the general results we present in this chapter will be used only *once* in the later chapters. In addition, some of the results are presented with a higher level of generality than is actually needed in subsequent applications. So why did not we define such preliminaries "in-place" or without this extra generality? The answer to this question is that such results are of independent interest to be treated separately. For example, this applies to our treatment of semantic security/indistinguishability and deterministic extractors. In addition, treating such results "in-place" would be actually more confusing, making some of the later results either "come out from the sky" (i.e., without a clear reason of what actually happened), or to look unnecessarily technical and complicated.

As a result, however, the reader might get a little bit overwhelmed with the amount of diverse information presented here, and even distracted from the main topics studied in subsequent chapters. As a compromise, we suggest that the reader follows the following general guidelines.

- Section 2.1 (basic notation), Section 2.2 (distributions), Section 2.4 (basic cryptography) and Section 2.5 (private-key encryption) are quite basic and used extensively. Therefore, they should be at least skimmed right away.

- Section 2.6 (error-correcting codes) is used only in Section 4.1 when constructing

perfect ERF's (and slightly in Section 5.1.1). It could be better to skip it first.

- Section 2.7 (strong extractors) is used only in Section 4.2 when constructing statistical ERF's. It could be better to skip it first.

- Section 2.8 (deterministic extractors) is used only in Section 4.4.1 when constructing adaptively secure statistical ERF's. It could be better to skip it first.

- Section 2.9 (Fourier analysis) is used only in Section 5.1.4 when proving the lower bound on perfect AONT's. It could be better to skip it first.

- Finally, Section 2.3 (semantic security vs. indistinguishability) is mainly used in Section 3.3 to justify that the simple definition of an AONT that we use is actually much stronger than it seems at first. As such, this section is not really needed in order to follow our presentation. It is up to the reader whether to read it right away, in Section 3.3, or to skip it altogether. We recommend to read (or skim) this section right away (possibly skipping the proofs) since it is quite simple.

As a short summary, Sections 2.6–2.9 can be easily skipped upon the first reading.

2.1 Basic Notation and Terminology

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. Recall that a function $\mu(k)$ is called *negligible* if for any polynomial $p(k)$ there exists k_0 such that for all $k > k_0$ we have $\mu(k) < 1/p(k)$. We often write $negl(k)$ to indicate some negligible function of k, without giving it an explicit name. We denote by $x \in_R D$ a process of selecting x from the domain D uniformly at random. We denote by PPT a probabilistic polynomial time algorithm, and given such an A, we denote by $y \leftarrow A(x)$ sampling an output y when running A on input x. We let 1^k denote the string of 1's of length k. When a PPT algorithm is given 1^k as the input,

this suggests that A is allowed to work in time polynomial in k. We will often omit 1^k, however, when the security parameter k is clear. Unless otherwise specified, we will consider security against nonuniform adversaries.

Let $\binom{n}{\ell}$ denote the set of size-ℓ subsets of $[n] = \{1 \ldots n\}$. For $L \in \binom{n}{\ell}$, $y \in \{0,1\}^n$, 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, and by $\langle a_1, \ldots, a_k \rangle$ the k-tuple of $a_1, \ldots, a_k$. Given vectors $x, y \in \{0,1\}^n$, we will denote their inner product modulo 2 as $x \cdot y$. We also denote by $GF(q)$ a finite field on q elements, where q is a prime power.

2.2 Distributions and Indistinguishability

We denote by U_k the uniform distribution on $\{0,1\}^k$. The distribution *induced* by a function $f : \{0,1\}^n \to \{0,1\}^k$ is its output distribution over $\{0,1\}^k$ when the input was chosen uniformly at random in $\{0,1\}^n$. A family of distributions $p = \{p_k\}$ is called *efficiently samplable* if there exists a PPT algorithm that on input 1^k outputs a random sample from p_k. Often, when the security parameter k is clear or implicit, we simply say that the distribution p is efficiently samplable.

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

$$\max_{S \subseteq D} \Big| \Pr[X \in S] - \Pr[Y \in S] \Big| = \frac{1}{2} \cdot \sum_{\alpha} \Big| \Pr[X = \alpha] - \Pr[Y = \alpha] \Big| \tag{2.1}$$

Definition 1 *Let k be the security parameter and $A = \{A_k\}$, $B = \{B_k\}$ be two ensembles of probability distributions. We say that A and B are*

- perfectly indistinguishable*, denoted $A \equiv B$, if distributions A_k and B_k are identical for all k.*
- statistically indistinguishable*, denoted $A \cong_\epsilon B$, if the statistical distance $\|A_k - B_k\|$ is a negligible function of k.*
- computationally indistinguishable*, denoted $A \cong_c B$, if for any PPT algorithm*

D (called the distinguisher*) we have that*

$$|\Pr(D(A_k) = 1) - \Pr(D(B_k) = 1)| = negl(k)$$

where the probability is taken over the coin tosses of D and the random choices of A_k and B_k. The absolute value above is called the advantage *of D in distinguishing A from B, denoted $\mathsf{Adv}_D(A, B)$.*

Sometimes when the security parameter is implicit, we will sometimes slightly abuse the terminology and identify the ensembles A and B with the corresponding probability distributions A_k and B_k. And when the statement can hold for any of the above choices (or the choice is clear from the context), we simply write $A \approx B$.

We notice that the notions of statistical and perfect indistinguishability can be cast into the same framework as that of computational indistinguishability. Namely, if we relax the requirement that D is polynomial time bounded, we exactly get the definition of statistical indistinguishability, while if in addition we require that the advantage of D is always 0, we get perfect indistinguishability. This suggests the following methodology for proving statements of the form $A \approx B \Rightarrow A' \approx B'$, when it can hold for any choice of $\approx$. Namely, we assume that there exists a distinguisher D' having advantage ε' in distinguishing A' from B'. We construct then a distinguisher D, whose complexity is polynomial in that of D', and that distinguishes A from B with advantage ε. Then if $\varepsilon \geq c \cdot (\varepsilon')^t$ for some positive constants c and t, we have proven our implication. In particular, if D' is polynomially bounded, then so is D, if $\varepsilon' > 0$, then so is ε, and if $\varepsilon' > 1/p(k)$ for some polynomial p and for infinitely many k, then so is ε (for a different polynomial q).

We start from the following useful fact.

Lemma 1 *Let α, β be two (possibly dependent) random variables taking values 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 fact that for any X and Y, $\Pr(X \wedge Y) + \Pr(X \wedge \overline{Y}) = \Pr(X)$.

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

■

Lemma 2 *Let A and B be any two (ensembles of) probability distributions. Let R be chosen uniformly at random and let 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 efficiently samplable p.*

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

Proof:

(1) $\Rightarrow$ (2). Assume (2) is false for some efficiently samplable 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 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 the assumption on F. Thus, $\Pr(\gamma = 1) \geq \frac{1}{2} + \frac{\varepsilon}{2}$.

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. ■

Typically, we will only use the following simple corollary of the above.

Corollary 1 *Let A and B be any two (ensembles of) probability distributions. Let R be chosen uniformly at random, and let x_0 and x_1 be any two fixed strings (independent of the random variables above). Then*

$$\langle A, B \rangle \approx \langle A, R \rangle \quad \Rightarrow \quad \langle x_0, x_1, A, B \oplus x_0 \rangle \approx \langle x_0, x_1, A, B \oplus x_1 \rangle$$

Proof: Call $C = x_0 \oplus x_1$. Since x_0 and x_1 are fixed, we get

$$\begin{aligned}
\langle A, B \rangle \approx \langle A, R \rangle &\Rightarrow \langle x_0, x_1, A, B \rangle \approx \langle x_0, x_1, A, R \rangle \\
&\Rightarrow \langle x_0, x_1, A, B \oplus x_0 \rangle \approx \langle x_0, x_1, A, R \rangle \\
&\Rightarrow \langle x_0, x_1, A, B \oplus x_0, C \rangle \approx \langle x_0, x_1, A, B \oplus x_0 \oplus (x_0 \oplus x_1), C \rangle \\
&\Rightarrow \langle x_0, x_1, A, B \oplus x_0 \rangle \approx \langle x_0, x_1, A, B \oplus x_1 \rangle
\end{aligned}$$

The only non-trivial implication is the second to last one that uses Lemma 2 with $C = x_0 \oplus x_1$. ■

Indistinguishability Relative to an Oracle. In some applications we would like to say that A and B are indistinguishable, even if some side information is leaked to the distinguisher D. Typically, we place some restrictions on the type of side information, but allow the distinguisher to choose which particular side information

of this types he wants to see. Typically, this is modeled by letting D have "oracle access" to some function or some process (that depends on A and B or the way A and B were generated). For example, when talking about security of encryption schemes, we might allow D to have oracle access to the decryption oracle, allowing D to decrypt any messages of his choice under minimal restriction that D cannot decrypt the "target ciphertext". It is easy to see that the notion of indistinguishability and all the simple results we talked about *relativize* to this setting.

2.3 Semantic Security vs. Indistinguishability

Here we define and prove the equivalence of the notions of semantic security and indistinguishability, originally introduced by Goldwasser and Micali [33] in a particular context of encryption. We define these notions in a much more general context of any *"experiment"*. As a result, we show that these notions and their equivalence have nothing to do with encryption schemes, computational assumptions or anything else. Rather, this is just a basic fact about equivalence of two probabilistic experiments. Because we abstract away all the unnecessary complications, the proof of equivalence we present is very simple (despite its generality), and seems to be much simpler and understandable than most similar proofs that appeared in the literature.

Our general setup is the following. Assume we have some PPT experiment[1] $\mathbf{E}$ that takes a k-bit string x and transforms it into some y. We want to say that "y gives no information about x".

Definition 2 *We say that PPT experiment* $\mathbf{E}$ *is* semantically secure *in the computational sense if for any efficiently samplable* D *on* $\{0,1\}^k$*, any polynomial time computable binary relation* $\mathcal{R}$ *and any PPT adversary* A*, there exists a PPT* B *such that if* $X \leftarrow D$*,* $Y \leftarrow \mathbf{E}(X)$*,* $a \leftarrow A(Y, 1^k)$*,* $b \leftarrow B(1^k)$*, we get that*

$$\Pr(\mathcal{R}(X,a)=1) \leq \Pr(\mathcal{R}(X,b)=1) + negl(k) \tag{2.2}$$

[1] It is easy to see that the results we present do not hold in the computational setting if $\mathbf{E}$ is not polynomial time computable.

As usual, for the statistical setting we relax A, B, D, $\mathcal{R}$ from being polynomial time, and in the perfect setting we also require the advantage of A to be 0.

In other words, the odds of A (when given Y) producing a such that $\mathcal{R}(X, a)$ is satisfied are only negligibly more than the odds of B (when given *nothing*!) producing b satisfying $\mathcal{R}(X, b)$. Thus, whatever "useful information" about X one can get from Y, one can get without Y as well.

Definition 3 *We say experiment* $\mathbf{E}$ *is* indistinguishable for any two inputs *if for any* $x_0, x_1 \in \{0,1\}^k$, *we have*

$$\langle x_0, x_1, \mathbf{E}(x_0)\rangle \approx \langle x_0, x_1, \mathbf{E}(x_1)\rangle \tag{2.3}$$

Indistinguishability simply says that the adversary cannot distinguish the experiment performed on any fixed x_0 and x_1. We notice that the above condition is equivalent to saying that if i is chosen at random from $\{0,1\}$, $y \leftarrow \mathbf{E}(x_i)$, then no adversary can guess i given y significantly better than with probability $\frac{1}{2}$. Indeed,

$$\begin{aligned}\Pr(A(x_0, x_1, y) = i) &= \frac{1}{2}\left[\Pr(A(x_0, x_1, \mathbf{E}(x_0)) = 0) + \Pr(A(x_0, x_1, \mathbf{E}(x_1)) = 1)\right] \\ &= \frac{1}{2} + \frac{1}{2}\left[\Pr(A(x_0, x_1, \mathbf{E}(x_1)) = 1) - \Pr(A(x_0, x_1, \mathbf{E}(x_0)) = 1)\right]\end{aligned}$$

We will use this observation below.

Theorem 1 *The notions of semantic security and indistinguishability are equivalent.*

Proof: For simplicity, let us concentrate on the more interesting computational case. First, assume $\mathbf{E}$ is semantically secure. Take any x_0, x_1 (wlog, assume that $x_0 \neq x_1$, since otherwise $\mathbf{E}(x_0) = \mathbf{E}(x_1)$). Let D be the uniform distribution on $\{x_0, x_1\}$, and $\mathcal{R}(x, i) = 1$ if and only if $x = x_i$. Notice, D and R are describable just by x_0 and x_1. Assume $x \leftarrow \{x_0, x_1\}$, i.e. $x = x_i$ for a random $i \in \{0,1\}$. Notice that for *any* B, since B is not given any information about i and succeeds only when he outputs $b = i$ (since $x_0 \neq x_1$), we get that $\Pr(B(x_0, x_1) = i) = \frac{1}{2}$. Hence, semantic

security here implies that for any PPT A we get that if $y \leftarrow \mathbf{E}(x_i)$, then

$$\Pr(A(x_0, x_1, y) = i) \leq \frac{1}{2} + negl(k)$$

As we argued, this is equivalent to the definition of indistinguishability on x_0 and x_1.

Let us show the converse. Assume that $\mathbf{E}$ is not semantically secure for some polynomial time D, $\mathcal{R}$ and A having advantage ε over any PPT B. Let good be the probability that A succeeds, i.e. that when $X \leftarrow D$, $Y \leftarrow \mathbf{E}(X)$, $a \leftarrow A(Y)$, we get $\mathcal{R}(X, a) = 1$. And let bad denote the probability that when in addition we pick a brand new $X' \leftarrow D$, we get $\mathcal{R}(X', a) = 1$. First, we claim that

$$\mathsf{good} - \mathsf{bad} \geq \varepsilon \tag{2.4}$$

Indeed, we only need to construct a PPT B that achieves success probability bad. For that we consider B who himself samples $X \leftarrow D$, makes $Y \leftarrow \mathbf{E}(X)$, and outputs $a \leftarrow A(Y)$ (here we use that D and $\mathbf{E}$ are polynomial time). Clearly, when we sample brand new $X' \leftarrow D$, the success probability of B is exactly bad.

Recall that we need to find x_0, x_1 and adversary F who can distinguish $\mathbf{E}(x_0)$ from $\mathbf{E}(x_1)$. We start from F (here we use that $\mathcal{R}$ is polynomial time).

$F(x_0, x_1, y)$:

1. Let $a \leftarrow A(y)$, $\alpha = \mathcal{R}(x_1, a)$, $\beta = \mathcal{R}(x_0, a)$.
2. If $\alpha = \beta$, then output a random coin flip.
3. Otherwise ($\alpha = 1 - \beta$), output α.

In other words, F uses A to produce a "witness" a. Since A is supposedly good in producing "relevant" witnesses, a should be "more likely" to satisfy $\mathcal{R}(x_i, a)$ than $\mathcal{R}(x_{1-i}, a)$. And this is exactly what F checks. He computes both $\mathcal{R}(x_0, a)$ and $\mathcal{R}(x_1, a)$. If the results are the same, F did not learn anything, so he flips a coin. Otherwise, he outputs that single i that "produced" $\mathcal{R}(x_i, a) = 1$. Let us turn this intuition into a formal argument.

We analyze the behavior of F when X_0 and X_1 are sampled independently from D and $Y = Y_1 \leftarrow \mathbf{E}(X_1)$. We notice that in this setting we exactly have $\Pr(\alpha = 1) = \mathsf{good}$ and $\Pr(\beta = 1) = \mathsf{bad}$. Thus, by Lemma 1 and Equation (2.4) we get

$$\begin{aligned}\Pr(F(X_0, X_1, Y_1) = 1) &= \frac{1}{2} + \frac{1}{2}(\Pr(\alpha = 1) - \Pr(\beta = 1)) \\ &= \frac{1}{2} + \frac{1}{2}(\mathsf{good} - \mathsf{bad}) \\ &\geq \frac{1}{2} + \frac{\varepsilon}{2}\end{aligned}$$

In particular, since the above holds on average over X_0 and X_1, there exist some *particular* x_0 and x_1 such that we get

$$\Pr(F(x_0, x_1, \mathbf{E}(x_1)) = 1) \geq \frac{1}{2} + \frac{\varepsilon}{2}$$

Since the algorithm F is symmetric in x_0 and x_1, this means that

$$\Pr(F(x_0, x_1, \mathbf{E}(x_0)) = 1) \leq \frac{1}{2} - \frac{\varepsilon}{2}$$

Overall, $\Pr(F(x_1, x_0, \mathbf{E}(x_1)) = 1) - \Pr(F(x_1, x_0, \mathbf{E}(x_0)) = 1) \geq \varepsilon$. ∎

EXAMPLES. The notion of public-key encryption is a special case, where the experiment $\mathbf{E}$ samples a pair (pk, sk) of a random public and secret keys, computes encryption z of x, and returns $y = (pk, z)$. The same holds for private-key encryption, except there is no public key above (see Section 2.5). We will also have another definition of All-Or-Nothing Transforms in Section 3.3 that would fall into this category, justifying the usefulness of this general view.

USEFULNESS. The usefulness of the above equivalence is in the following. Semantic security is somewhat messy to define and to verify. However, it captures very well our intuition that $\mathbf{E}(x)$ does not convey any information about x. On the other hand, indistinguishability of any two inputs is a much simpler condition to verify and to work with. However, it is not immediately clear if it really says that $\mathbf{E}(x)$ does not convey any information about x. So the equivalence asserts that intuitive and

convenient definitions actually coincide. As the result, it is much more customary to work with indistinguishability.

VARIATIONS. There are several variations of the notion of semantic security, all of which turn out to be equivalent because of the equivalence above. For example, we could relax the definition by replacing a relation $\mathcal{R}$ with only a function f and target the adversary to produce a such that $f(X) = a$. Since indistinguishability corresponds to having a relation which is actually a function (i.e., $f(x_i) = i$), the equivalence follows. Also, rather than requiring that for any A there is some B, we could run A twice: first time on the correct Y, and second time on a brand new $Y' \leftarrow \mathbf{E}(X')$, where $X' \leftarrow D$. The definition then says that A did not learn anything because for any D and $\mathcal{R}$ he could not even see the difference when the correct X was replaced by a brand new X'. This captures our intuition of semantic security slightly less and starts to remind the indistinguishability definition. But the equivalence is clear since the B we constructed in the proof really simulates the run of A on a brand new Y', which is exactly what we are doing. A slight advantage of that definition, though, is that we do not need to restrict D and $\mathbf{E}$ to polynomial time, which is not that important.

Another traditional definition (originated all the way in [33]) is to say that the best B could do anyway without the knowledge of X, is to produce a fixed b_{max} maximizing the probability of satisfying $\mathcal{R}(X, b_{max})$ when X is chosen from D. Calling this maximum probability $p(\mathcal{R}, D)$, we say that the probability of A's success is at most $p(\mathcal{R}, D) + negl(k)$. In some sense, this could be slightly unfair since PPT B may not be able to produce the required b_{max} (in polynomial time). But, first of all, since we chose to have non-uniform adversaries and the adversaries depend on $\mathcal{R}$, we could hardwire this b_{max}. Alternatively, the notions are again equivalent since in the indistinguishability based definition we had $p(\mathcal{R}, D) \equiv \frac{1}{2}$ which B can achieve by outputting a random coin. Again, this modification has a slight advantage of not requiring D and $\mathbf{E}$ to be polynomial time. To summarize, there are several small variations of the definitions, all of which turn out to be equivalent, justifying the

"universality" of our notion.

EXPERIMENTS WITH A SETUP. Sometimes the definition of the experiment $\mathbf{E}$ can be split up into two natural phases: the setup phase, and the actual experiment phase. A classical example is public-key encryption, where the setup can chose a random public/private key pair, and the actual experiment just encrypts the given message (see [42] for more details on definitions of public-key encryption). In this case we might want to let the adversary observe the "public part" of the setup, and based on that try to come up with: a) some x_0 and x_1 that he claims to distinguish for the case of indistinguishability, or b) distribution D and relation $\mathcal{R}$ (which are automatically polynomial time in the computational setting) that he can "defeat" for the case of semantic security. Clearly (at least for non-uniform adversaries), if the experiment with the setup is secure, combining the setup and the experiment into a single "super-experiment" is also secure, but the converse is false in general (as it is easy to see). The reason is that the "public information" from the setup may help the adversary to select x_0 and x_1 (or D and $\mathcal{R}$), that he cannot select at the very beginning. However, the *equivalence* between semantic security and indistinguishability still holds for experiments with the setup.

We already remarked that setup has a natural meaning for public-key encryption (and results in a stronger definition). For private-key encryption the setup can be defined as the process of choosing a secret key, but there is no public output, so there is no reason to do the setup separately. For the definition of an All-Or-Nothing Transform that we give later, there is no natural meaning for the setup (except for an "empty" setup).

2.4 Cryptographic Basics: OWF, PRG and PRF

We now define some basic cryptographic notions. We refer the reader to [29] for a more detailed exposition, references and proofs of some of the basic facts presented here.

Definition 4 *A polynomial time computable function $g : \{0,1\}^* \to \{0,1\}^*$ is called a* one-way function *(*OWF*) if for any PPT adversary A, if x is chosen at random from $\{0,1\}^k$, we have*

$$\Pr(A(g(x), 1^k) \in g^{-1}(g(x))) = negl(k) \tag{2.5}$$

In other words, g is easy to compute but hard to invert on a random input x. A relaxation of the notion of a OWF is a notion of a *weak* OWF, where for some polynomial $p(k)$ the condition 2.5 is replaced by

$$\Pr(A(g(x), 1^k) \in g^{-1}(g(x))) < 1 - \frac{1}{p(k)} \tag{2.6}$$

In other words, no adversary succeeds with probability negligibly close to 1, so g is "slightly" hard to invert. A folklore result that we will use later is that existence of weak OWF's imply the existence of regular OWF's.

Definition 5 *A deterministic polynomial time computable function*[2] $G : \{0,1\}^k \to \{0,1\}^{m(k)}$ *is called a* pseudorandom generator *(*PRG*) stretching from k to $m(k)$ bits (where $m(k) > k$) if the following are computationally indistinguishable:*

$$\langle G(r) \mid r \in_R \{0,1\}^k \rangle \cong_c \langle R \mid R \in_R \{0,1\}^{m(k)} \rangle$$

In other words, $G(r)$ for a *random* $r \in \{0,1\}^k$ (this r is called a *seed of* G) is indistinguishable to a PPT algorithm from a truly random $R \in \{0,1\}^{m(k)}$. The following important result (the hard part of which was proved by [34]) shows that "OWF's $\iff$ PRG's".

Theorem 2 OWF*'s exist* $\iff$ PRG*'s stretching to $k+1$ bits exist* $\iff$ PRG*'s stretching to $m(k)$ bits exist for any polynomial $m(k) > k$.*

One of the consequences is that we can talk about PRG's "in general" without worrying about the particular stretch factor. Finally, we introduce another classical

[2] Technically speaking, ensemble of functions: one for each k.

notion of *pseudorandom function (*PRF*) families*. For that we need to define the notion of an algorithm A having an *oracle access* to some function f. By that we mean that at any point during its execution, A can learn in a single step the value $f(x)$ for any x of A's choice in the domain of f. We denote such an A by A^f.

Definition 6 *A function family*[3] $\mathcal{F} = \{F_s : \{0,1\}^k \to \{0,1\}^k \mid s \in \{0,1\}^k\}$, *is called a* pseudorandom function family *(*PRF *family), if each F_s is polynomial time computable, and for any PPT A, we have*

$$|\Pr(A^{F_s}(1^k) = 1) - \Pr(A^F(1^k) = 1)| = negl(k)$$

The first probability is taken over a random choice of the seed $s \in \{0,1\}^k$ *and random coins of A, while the second — over coins of A and the choice of a totally random function F from $\{0,1\}^k$ to $\{0,1\}^k$.*

In other words, no PPT A can distinguish between having an oracle access to a pseudorandom function F_s (where only the *seed* s is chosen at random), form having access to a truly random function F (where each $F(x)$ is random for each $x \in \{0,1\}^k$). Thus, a singly exponential function family $\mathcal{F}$ is indistinguishable from a doubly exponential family of all functions. The following generalization of Theorem 2 is one of the fundamental results in cryptography unifying the notions of OWF's, PRG's and PRF's and suggesting that a lot of cryptography can be built on a single assumption: **existence of one-way functions**.

Theorem 3 OWF*'s exist* $\iff$ PRG*'s exist* $\iff$ PRF *families exist.*

2.5 Symmetric-Key Encryption

We briefly review the notion of private-key cryptography, in particular, symmetric-key encryption. Here two parties share a secret key and wish to perform secure encryption.

[3]Again, technically speaking we have an ensemble of such families: one for each k. Also, the choice of domain and range to be $\{0,1\}^k$ is arbitrary, any domain and range with description of a point polynomial in k will work as well.

First, let us give the simplest possible definition of private-key encryption.

Definition 7 *A* symmetric-key (or private-key) encryption scheme $\mathcal{C}$ *is given by three PPT algorithms* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Gen*, on input* 1^k*, generates a secret key* sk*. Given* $x \in \{0,1\}^k$*, the encryption algorithm* $\mathsf{Enc}_{sk}(x)$ *generates a (random) encryption* y *of* x*. The decryption algorithm* Dec *is deterministic, and for any* $y \in \mathsf{Enc}_{sk}(x)$*, we have* $\mathsf{Dec}_{sk}(y) = x$*. Encryption scheme* $\mathcal{C}$ *is called* indistinguishable for two inputs *if for any* $x_0, x_1 \in \{0,1\}^k$*, we have*

$$\langle x_0, x_1, \mathsf{Enc}_{sk}(x_0) \rangle \approx \langle x_0, x_1, \mathsf{Enc}_{sk}(x_1) \rangle \tag{2.7}$$

We see that the definition follows the general paradigm of Section 2.3. In particular, one can define an equivalent semantic security definition. We now give a few classical examples.

One-time pad. This is the first "cryptographic scheme" ever proposed by Shannon [56]. Here the secret key is a random string R of length k, and the encryption of $x \in \{0,1\}^k$ is just $x \oplus R$. This scheme is clearly perfectly secure. However, the length of the secret key is the same as the length of the message. It is easy to show that this is unavoidable if we wish to achieve perfect secrecy [56].

Pseudorandom one-time pad. This is one of the first applications of pseudorandom generators. Assume we have a PRG G stretching from n to k bits. The key is a random $r \in \{0,1\}^n$ we encrypt $x \in \{0,1\}^k$ by a pseudorandom "one-time pad" $G(r)$, i.e. $\mathsf{Enc}(x;\ r) = x \oplus G(r)$. Here the length of the key, n, could be much smaller than the length of the message, k. Also, the security is necessarily computational and follows immediately from Corollary 1 and the definition of a PRG.

We notice that the definition of indistinguishability gurantees security of encrypting a single message, but nothing really is guaranteed about encrypting more messages. In fact, the one-time pad encryptions above are clearly bad if one is to use them twice (that is why they are called "one-time") on x and x': the XOR of the encryptions reveals $x \oplus x'$. A more interesting definition arises if we allow the adversary (who

has to distinguish encryptions of x_0 and x_1) to have *oracle access* to the encryption oracle. That is, he can obtain encryptions of any messages of his choice. The following generalization of the "one-time pad" schemes above is well known to achieve such security.

"STANDARD" SYMMETRIC-KEY ENCRYPTION. $\mathcal{F} = \{F_s : \{0,1\}^k \to \{0,1\}^k \mid s \in \{0,1\}^k\}$ be a PRF family. We select a random shared secret key $s \in \{0,1\}^k$ and encrypt x by a pair $\langle x \oplus F_s(R), R\rangle$, where R is chosen at random from $\{0,1\}^k$.

Beside their simplicity, we notice the following important feature of the above examples: the secret key is a *uniform random value*, i.e. does not have any special structure like representing a k-bit prime, etc. We remark that there are many other such examples in private-key cryptography where a secret is just a random value: pseudorandom permutations, block ciphers, messages authentication codes, various keyed hash functions. In fact, even in the public-key cryptography we frequently have simple systems where the secret is just a random value. For example, various schemes based on the Discrete Logarithm or the Diffie-Hellman Assumptions (e.g., [26, 53]) pick a random x and publish its exponent. When defining exposure-resilient functions in Section 3.1, we will see how to make all these systems "exposure-resilient" (see Section 3.2).

2.6 Linear Error-Correcting Codes

An *error-correcting code* is a deterministic mapping from k-bit strings to n-bit strings (the latter called *codewords*) such that any two codewords are very different from each other, i.e. very "far apart" in terms of the *Hamming distance*.[4] Thus, even if one misses or has corrupted a relatively few bits of some codeword, it is possible to recover these bits.

[4]The *Hamming distance* between $x, y \in \{0,1\}^n$ is the number of coordinates they differ in. The *Hamming weight* of $x \in \{0,1\}^n$ is the number of non-zero coordinates of x, i.e. its Hamming disntance from 0.

We will consider *binary linear* $[n, k, d]$ *error-correcting codes*. Such a 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 a $k \times n$ *generator matrix* M over $GF(2)$. For any vector $v \in \{0,1\}^k$ (which is viewed as a column vector, i.e. a $k \times 1$ matrix, and $v^\top$ denotes the corresponding $1 \times k$ row vector), the *codeword* corresponding to v is $v^\top M$. A code is said to have *minimum distance* d if for every two distinct vectors $u, v \in \{0,1\}^k$, $u^\top M$ and $v^\top M$ 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.

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). The ratio k/n is called the *rate* of the code, while d/n is called the *relative distance*. A standard result in coding theory shows that a random linear code (corresponding to a random M) is asymptotically good with high probability (provided the rate and relative distance are not very large constants). Many explicit constructions for asymptotically good codes (e.g., the Justesen code) exist.

We remark on two well-known bounds on error-correcting codes. First, it is always the case that $k \leq n - d + 1$ (Singleton bound). Second, $d \leq n/2$ for $k > \log n$. In other words, the distance cannot be more than $n/2$. On the positive side, we can make d arbitrarily close to $n/2$ (i.e., $(\frac{1}{2} - \varepsilon)n$ for any $\varepsilon > 0$) at the expense of making n large compared to block-length k.[5]

Finally, we mention the famous "parity" linear code, called the *Hadamard code*,[6] stretching k bits to $n = 2^k - 1$ bits. Here a k-bit message $u = u_1 \ldots u_k$ is encoded into $(2^k - 1)$-long bit message c by taking all $(2^k - 1)$ non-empty XOR's of the u_i's. Namely, for each non-empty $J \subseteq [k]$, the bit c_J of the encoding is $\oplus_{i \in J} u_i$. Viewed another way,

[5]For example, $n = poly(k)$ if we use the so called Reed-Solomon code concatenated with the Hadamard code described below, $n = O(k/\varepsilon^3)$ if we use the so called algebraic-geometric code concatenated with the Hadamard code, and $n = k/\varepsilon^2$ if we use a random code. Also, the Hadamard code by itself has $n = 2^k - 1$ and $d = (n+1)/2$.

[6]In coding theory lingo, this is also known as the "dual" of another famous code — the *hamming code* — which is a perfect code of distance 3. Thus, the corresponding generator matrix M is simply the "parity check" matrix of the hamming code.

for every non-zero $a \in \{0,1\}^k$, the a-th coordinate of c is the inner product modulo 2 of u and a: $c(a) = u \cdot a$. We omit $a = 0^k$ (i.e., $J = \emptyset$) since it always produces 0 as a parity, so it is not useful for decoding purposes. The Hadamard code is clearly linear and its generator matrix M is obtained by writing column-by-column all $(2^k - 1)$ non-zero k-bit strings. It has distance $2^{k-1} = (n+1)/2$ since if $u \neq u'$, exactly 2^{k-1} subsets J have $c_J \neq c'_J$ (equivalently, exactly 2^{k-1} vectors a have $(u - u') \cdot a = 1$).

For the proofs of these results and further information on error-correcting codes, see [40].

2.7 Strong Extractors

Extractors were first formally introduced in a seminal paper by Nisan and Zuckerman [46]. An *extractor* is a family of hash functions $\mathcal{H}$ such that when a function h_i is chosen at random from $\mathcal{H}$ (by choosing a random i), and is applied to a random variable X that has "enough randomness" in it, the resulting random variable $Y = h_i(X)$ is 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 index i (used in specifying h_i) is revealed! (Perhaps the best known example of a strong extractor is given in the Leftover Hash Lemma of [36], where standard 2-universal hash families are shown to be strong extractors.) This is illustrated in Figure 2-1.

We now define the notion of extractor more precisely. We say that random variable X distributed over $\{0,1\}^n$ has *min-entropy* m if for all $x \in \{0,1\}^n$, $\Pr(X = x) \leq 2^{-m}$. High min-entropy will turn out to be a good formal indicator for X having "a lot of randomness".

Definition 8 ([46]) *A family of efficiently computable hash functions* $\mathcal{H} = \{h_i : \{0,1\}^n \to \{0,1\}^k \mid i \in \{0,1\}^d\}$ *is called a* strong (m, ε)-extractor, *if for any random variable* X *over* $\{0,1\}^n$ *that has min-entropy* m, *if* i *is chosen uniformly at random from* $\{0,1\}^d$ *and* R *is chosen uniformly at random from* $\{0,1\}^k$, *the following two*

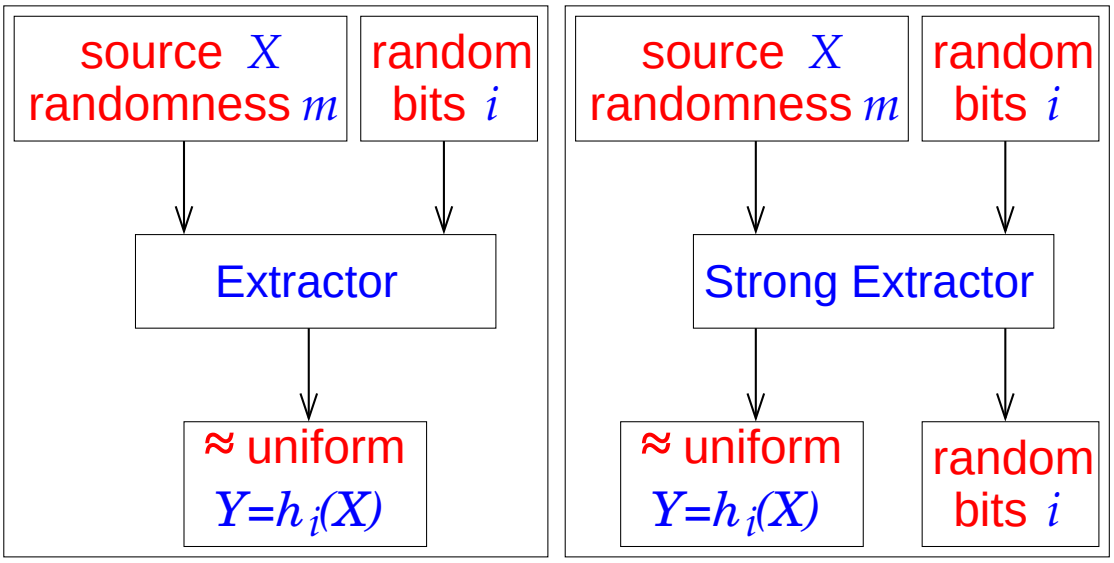

Figure 2-1: Extractors and Strong Extractors.

distributions are within statistical distance ε from each other:

$$\langle i, h_i(X)\rangle \cong_\epsilon \langle i, R\rangle \tag{2.8}$$

Throughout, we will only talk about efficient *extractor families $\mathcal{H}$. That is, given any $i \in \{0,1\}^d$ and $x \in \{0,1\}^n$, one can efficiently (i.e., in time poly(n)) compute $h_i(x)$.*

Thus, investing enough true randomness, namely the amount needed to select a random member of $\mathcal{H}$, one can "extract" something statistically close to a truly random string from the randomness in a given distribution X. Much work has been done in developing this area (e.g. [46, 32, 59, 64, 45, 61, 50, 49]). 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 $\mathcal{H}$).

We will use the following efficiently constructible families of strong extractors developed by [59, 49].

Theorem 4 ([59, 49]) *For any n, m and ε such that $n > m > 2\log(1/\varepsilon)$, there exist efficient strong (m,ε)-extractor families $\mathcal{H} = \{h_i : \{0,1\}^n \to \{0,1\}^k \mid i \in \{0,1\}^d\}$ satisfying:*

1. $k = m - 2\log(1/\varepsilon) - O(1)$ *and* $d = 4(m - \log(1/\varepsilon)) + O(\log n)$. [59]

2. $k = (1-\delta)m - O(\log(1/\varepsilon))$ *and* $d = O(\log^2 n + \log(1/\varepsilon))$ *(*$\forall$ *const.* $\delta > 0$*).* [49]

3. $k = m - 2\log(1/\varepsilon) - O(1)$ *and* $d = O((\log^2 n + \log(1/\varepsilon))\log m)$. [49]
   (provided $\varepsilon > \exp(-n/(\log^* n)^{O(\log^* n)})$*, which will hold in our applications).*

Notice that if a source has min-entropy m, we cannot hope to extract more than m (even statistically close to) random bits, i.e. we must have $k \leq m$. In fact, [48] show that $k \leq m - 2\log(1/\varepsilon) + O(1)$. The remarkable fact about the above extractor families is that they really almost achieve this seemingly impossible bound for *any* X having min-entropy m. However, in the first extractor the amount of extra-randomness d is roughly $4m$, so we invest more truly random bits than the amount of random bits that we extract! Of course, the catch is that we do not "lose" the extra-randomness, so the extractor is still very useful. On the other hand, the last two extractors are much more randomness efficient (provided m is large enough).

For more information on these topics, see the excellent survey articles [44, 45].

2.8 Deterministic Extractors and t-wise Independent Functions

In the previous section we saw that we can extract almost all the randomness from *any* distribution of min-entropy m by investing very few extra truly random bits. On the other hand, it would be very desirable not to invest any additional randomness at all, i.e. to have just a single deterministic function $f : \{0,1\}^n \to \{0,1\}^k$ that would extract all the randomness from our source X. However, it is very easy to see that this task is too ambitious.[7] In other words, we cannot hope that one function f will be good for *all* sources with min-entropy m, and therefore have to invest some

[7] For example, concentrate all the mass of X uniformly on the preimages of 2^{m+k-n} "most frequent" points in the range of f. There are at least 2^m such preimages, so X has min-entropy at least m. On the other hand, $f(X)$ can induce a distribution statistically close to uniform on $\{0,1\}^k$ only if $m + k - n > k - 1$, i.e. $m > n - 1$, so X was almost uniform to begin with.

extra randomness. However, in many applications we have some set $\mathcal{X}$ of "allowed" sources of min-entropy m, and we only need f to extract randomness from sources $X \in \mathcal{X}$ (and do not "care" about other sources; we will see an example of this later in Section 4.4). In this section we discuss how to construct such f culminating in Theorem 6 and Corollary 3. While written with the current emphasis for the first time, all the ideas of this section were largely suggested to us by Trevisan and Vadhan [62], who were the first to consider "general-purpose" deterministic extractors.

2.8.1 Deterministic and δ-sure Extractors

Towards Deterministic Extractors. As we observed, it is conceivable to have this *single* deterministic f "tuned up" to work just for the sources in $\mathcal{X}$. Such f is called a *deterministic extractor for* $\mathcal{X}$. Unfortunately, the *explicit constructions* of such f for a given $\mathcal{X}$ often turn out to be difficult. Therefore, we settle for the next best option. We will design an efficiently samplable family of hash function $\mathcal{F}$ such that when f is chosen at random from this family, f will be a good (deterministic) extractor for *every* $X \in \mathcal{X}$ with high probability. Moreover, we will not use anything about $\mathcal{X}$ except for its cardinality $|\mathcal{X}|$ and the fact that every $X \in \mathcal{X}$ has min-entropy m. In other words, for any X of min-entropy m, with high probability (much better than $1/|\mathcal{X}|$) a random f in $\mathcal{F}$ will be a good extractor function for X. Then we will simply take the union bound over all $X \in \mathcal{X}$. This justifies the following definition and its immediate corollary.

Definition 9 (implicit in [62]) *A family of efficiently computable hash functions* $\mathcal{F} = \{f_i : \{0,1\}^n \to \{0,1\}^k \mid i \in \{0,1\}^d\}$ *is called a* δ-sure (m,ϵ)-extractor, *if for any random variable* X *over* $\{0,1\}^n$ *that has min-entropy* m, *with probability at least* $(1-\delta)$ *over the choice of random* $f = f_i$ *from* $\mathcal{F}$, *we have that the distribution induced by* $f(X)$ *is* ϵ*-close to the uniform distribution over* $\{0,1\}^k$.

Corollary 2 *For any collection* $\mathcal{X}$ *of distributions of min-entropy* m *over* $\{0,1\}^n$, *if* $\mathcal{F} = \{f_i : \{0,1\}^n \to \{0,1\}^k\}$ *is a* δ*-sure* (m,ϵ)*-extractor, then with probability at least*

$(1 - |\mathcal{X}|\delta)$ *a random* $f = f_i$ *chosen from* $\mathcal{F}$ *will be a* deterministic *extractor (with statistical deviation* ϵ*) for all* $X \in \mathcal{X}$.

Thus, we are interested in constructing δ-sure extractors, where δ will be really small (typically, much smaller than ϵ), so that we can take a large union bound over all sources in $\mathcal{X}$.

Comparing with regular extractors. It is interesting to compare this definition of $\mathcal{F}$ with the definition of a strong (m, ε)-extractor $\mathcal{H}$ from Section 2.7. It is easy to check that if $\mathcal{F}$ is a δ-sure (m, ϵ)-extractor, then it is also a strong $(m, \epsilon + \delta)$-extractor. In some sense, we "fine-tuned" ε into ϵ and δ. On the other hand, for any ϵ and δ satisfying $\varepsilon = \epsilon\delta$, a strong (m, ε)-extractor $\mathcal{H}$ is also a δ-sure (m, ϵ)-extractor. However, the usage of $\mathcal{F}$ is typically very different from that of $\mathcal{H}$. $\mathcal{H}$ is designed to work for *all* X, but for *each* particular X we have to invest extra randomness and sample a *brand new* hash function $h \in \mathcal{H}$. $\mathcal{F}$ is designed to work for arbitrary but fixed collection $\mathcal{X}$ of sources of min-entropy m. We sample a function $f \in \mathcal{F}$ only *once* and with overwhelming probability this *particular* f will be a good *deterministic* extractor for *all* $X \in \mathcal{X}$. In other words, once we have chosen f we do not invest any more randomness later, no matter how many times and which sources $X \in \mathcal{X}$ are given to us (however, there is a negligible chance that our f is "bad").

As we said, the above is achieved by making δ very small (much smaller than $1/|\mathcal{X}|$) and implies that we cannot make the size of $\mathcal{F}$ very small. In the very least, we must have $|\mathcal{F}| \gg |\mathcal{X}|$, since we have to take the union bound over all $X \in \mathcal{X}$. Since $|\mathcal{X}|$ is often exponential in n, we need at least $poly(n)$ random bits to sample f from $\mathcal{F}$, which is much more than the polylogarithmic number of bits that were sufficient for regular extractors. Thus, even though it would be nice to minimize the number of bits to describe $f \in \mathcal{F}$, a more immediate concern will be to make sure that the number of bits is polynomial in n, so that f is efficiently describable and computable.

To summarize, a strong (m, ε)-extractor $\mathcal{H}$ is designed to work for any distribution X (with min-entropy m) and the emphasis is to use very few extra random bits, since

we have to use new random bits for every such X. A δ-sure (m, ϵ)-extractor is designed to work for a particular (albeit arbitrary) *collection* of sources $\mathcal{X}$ (of min-entropy m each), and the emphasis is to be able to efficiently sample a single f from $\mathcal{F}$ that will be a good deterministic extractor for all $X \in \mathcal{X}$.

2.8.2 t-wise Independent Function Families

We will give a simple efficient construction of δ-sure (m, ϵ)-extractor families based on the construction and the analysis of [62], but first we need to recall the notion of t-wise independence.

Definition 10 *A collection of random variables $Y_1, \ldots, Y_N$ over some space S is said to be* t-wise independent *if for any t distinct indices $i_1, \ldots, i_t$ we have that the variables $Y_{i_1}, \ldots, Y_{i_t}$ are independent[8] from each other. A family $\mathcal{F}$ of functions from $\{0,1\}^n$ to $\{0,1\}^k$ is said to be* t-wise independent *if when a function f is chosen from $\mathcal{F}$ at random, the values of f are t-wise independent, i.e. the random variables $\{f(x) \mid x \in \{0,1\}^n\}$ are t-wise independent over $\{0,1\}^k$.*

When talking about t-wise independent families of function we will always additionally assume that for any $x \in \{0,1\}^n$, the distribution of $f(x)$ is uniform over $\{0,1\}^k$. In other words, any t values of f are independent and *uniform*. There exist efficient t-wise independent function families from n to $k \leq n$ bits, where it takes $O(tn)$ random bits to describe a function in $\mathcal{F}$ (see [19]). The simplest such family is a family of polynomials of degree t over $GF(2^n)$ "truncated" to k bits. In other words, given a polynomial p of degree t over $GF(2^n)$ and a point $x \in GF(2^n)$, we evaluate $p(x)$ over $GF(2^n)$ and output the first k bits of the canonical n-bit representation of the answer. The t-wise independence follows from the fact that any t values of a random polynomial of degree t are independent and random.

We will need the following "tail inequality" for the sum of t-wise independent random variables proven by Bellare and Rompel [9]. There they estimate $\Pr[|Y - \mathbf{E}[Y]| >$

[8] I.e., $\Pr(Y_{i_1} = y_1 \wedge \ldots \wedge Y_{i_t} = y_t) = \Pr(Y_{i_1} = y_1) \cdot \ldots \cdot \Pr(Y_{i_t} = y_t)$, for any $y_1, \ldots, y_t$.

$A]$, where Y is the sum of t-wise independent variables. We will only be interesting in $A = \epsilon \cdot \mathbf{E}[Y]$, where $\epsilon \leq 1$. In this case it is easy to trace the proof of Lemma 2.3 (and Lemma A.5 that is used to prove it) of [9] and get the following result.

Theorem 5 ([9]) *Let t be an even integer, and assume $Y_1, \ldots, Y_N$ are t-wise independent random variables in the interval $[0,1]$. Let $Y = Y_1 + \ldots + Y_N$, $\mu = \mathbf{E}[Y]$ and $\epsilon < 1$. Then*

$$\Pr(|Y - \mu| \geq \epsilon\mu) \leq C_t \cdot \left(\frac{t}{\epsilon^2 \mu}\right)^{t/2} \tag{2.9}$$

where the constant $C_t < 3$ and in fact $C_t < 1$ for $t \geq 8$.

2.8.3 t-wise Independent Functions as Extractors

We now argue that any family of t-wise independent functions is a very good δ-sure (m, ε)-extractor family. For that we need the following crucial lemma.

Lemma 3 *Let $\mathcal{F}$ be a family of t-wise independent functions (for even $t \geq 8$) from n to k bits, let X be a distribution over $\{0,1\}^n$ of min-entropy m, and let $y \in \{0,1\}^k$. Assume for some $\alpha > 0$*

$$k \leq m - \left(2\log\frac{1}{\epsilon} + \log t + 2\alpha\right) \tag{2.10}$$

Let f be chosen at random from $\mathcal{F}$ and x be chosen according to X. Then

$$\Pr_{f \in \mathcal{F}}\left(\left|\Pr_x(f(x) = y) - \frac{1}{2^k}\right| \geq \epsilon \cdot \frac{1}{2^k}\right) \leq 2^{-\alpha t} \tag{2.11}$$

In other words, for any $y \in \{0,1\}^k$, if f is chosen from $\mathcal{F}$ then with overwhelming probability we have that the probability that $f(X) = y$ is $\frac{1}{2^k}(1 \pm \epsilon)$.

Proof: Let p_x denotes the probability that $X = x$ and let q denote the random variable (only over the choice of f) which equals to the probability (over the choice of x given f) that $f(x) = y$, i.e.

$$q = \sum_{x \in \{0,1\}^n} p_x \cdot I_{\{f(x)=y\}}$$

where $I_{\{f(x)=y\}}$ is an indicator variable which is 1 if $f(x) = y$ and 0 otherwise. Since for any x the value of $f(x)$ is uniform over $\{0,1\}^k$, we get that $\mathbf{E}_f[I_{\{f(x)=y\}}] = 2^{-k}$, and thus $\mathbf{E}_f[q] = 2^{-k}$. Notice also that the variables $I_{\{f(x)=y\}}$ are t-wise independent, since f is chosen at random from a family of t-wise independent functions. And finally notice that since X has min-entropy m, we have that all $p_x \leq 2^{-m}$.

Thus, if we let $Q_x = 2^m \cdot p_x \cdot I_{\{f(x)=y\}}$, and $Q = \sum_{x\in\{0,1\}^n} Q_x = 2^m q$, we get that the variables Q_x are t-wise independent, all reside in the interval $[0,1]$, and $\mathbf{E}[Q] = 2^m \mathbf{E}[q] = 2^{m-k}$. Now we can apply the tail inequality given in Theorem 5 and obtain:

$$\begin{aligned}
\Pr_f\left[\left|q - \frac{1}{2^k}\right| \geq \epsilon \cdot \frac{1}{2^k}\right] &= \Pr_f\left[\left|Q - 2^{m-k}\right| \geq \epsilon \cdot 2^{m-k}\right] \\
&\leq \left(\frac{t}{\epsilon^2 \cdot 2^{m-k}}\right)^{\frac{t}{2}} = \left(\frac{1}{2^{m-k-2\log\frac{1}{\epsilon}-\log t}}\right)^{\frac{t}{2}} \\
&\leq 2^{-\alpha t}
\end{aligned}$$

where the last inequality follows from Equation (2.10). ∎

The above lemma almost immediately suggests that a family of t-wise independent functions is a good δ-sure extractor. Indeed, if we take a union bound over all $y \in \{0,1\}^k$, we get that with probability at least $(1 - 2^{k-\alpha t})$ *all* y have $\Pr_x(f(X) = y) = \frac{1}{2^k}(1 \pm \epsilon)$, which easily implies that $f(X)$ is ϵ-close to uniform on $\{0,1\}^k$. Thus, to make $\mathcal{F}$ δ-sure, we need $2^{k-\alpha t} \leq \delta$. Since we will have $k < m$ anyway, it suffices to have $\alpha t \leq m + \log\frac{1}{\delta}$. We set $\alpha = 1$ for simplicity[9] and get $t = m + \log\frac{1}{\delta}$, while k could be set to $m - (2\log\frac{1}{\epsilon} + \log t + O(1)) \geq m - (2\log\frac{1}{\epsilon} + \mathrm{loglog}\frac{1}{\delta} + \log m + O(1))$. Thus, we proved

Theorem 6 *Fix any n, m, ϵ and δ. Set*

$$t = m + \log\frac{1}{\delta} \quad , \quad k = m - \left(2\log\frac{1}{\epsilon} + \mathrm{loglog}\frac{1}{\delta} + \log m + O(1)\right)$$

[9] In fact, the "optimal" choice of α is $\log(m + \log\frac{1}{\delta})$, but this will not make much difference.

Then any family $\mathcal{F}$ of t-wise independent functions from n bits to k bits is a δ-sure (m, ϵ)-extractor.

We see two crucial features of this result that make it extremely useful. First, t is logarithmic in $1/\delta$, which means that we can afford to have exponentially small δ and still have efficient $\mathcal{F}$. On the other hand, the "entropy loss" for k (the expression subtracted from m) is logarithmic in $1/\epsilon$ and *doubly* logarithmic in $1/\delta$. Thus, δ can be exponentially smaller than ϵ. This means that we can set ϵ to a desirable level (say, only slightly negligible in n) and again can easily afford to make δ exponentially small. In particular, if we take any collection $\mathcal{X}$ of M distributions of min-entropy m, we can apply Corollary 2 with $\delta = 1/M^2$ (we also replace α in the proof of Theorem 6 from 1 to 2 to get rid of the factor of 2 in $\log(1/\delta) = 2\log M$), and easily handle exponentially large M:

Corollary 3 *Fix any n, m, ϵ, M and any collection $\mathcal{X}$ of M distributions over $\{0,1\}^n$ of min-entropy m each. Define*

$$t = m + \log M \quad , \quad k = m - \left(2\log\frac{1}{\epsilon} + \log m + \operatorname{loglog} M + O(1) \right)$$

and let $\mathcal{F}$ be any family of t-wise independent functions from n bits to k bits. Then with probability at least $(1 - \frac{1}{M})$ a random function $f \in \mathcal{F}$ will be a good deterministic extractor for $\mathcal{X}$, i.e. $f(X)$ will be ϵ-close to uniform over $\{0,1\}^k$ for any $X \in \mathcal{X}$. In particular, such deterministic f exists.

Interestingly enough, one can check that we would get almost the same bound on k if we were to choose the function f completely at random (using exponentially many random bits, and making it infeasible to use). Thus, efficiently samplable and computable family of t-wise independent functions (where we can make t reasonably small) does essentially as well as a family of all functions.

2.9 Quadratic Forms and Fourier Analysis

In this section we give some background from linear algebra. Further explanation can be found in many textbooks, e.g. M. Artin's *Algebra*. In this section most of the arithmetic will be over the reals, and we will try to use boldface when talking about vectors in $\mathbb{R}^m$, to separate them from vectors over $\{0,1\}^m$ which we talked about earlier. Let $\mathbf{u} = \{u_1, \ldots, u_m\}$, $\mathbf{v} = \{v_1, \ldots, v_m\}$ be two vectors in $\mathbb{R}^m$. We will use the notation $\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u}^\top \mathbf{v} = \sum_i u_i v_i$ to denote the inner product of $\mathbf{u}$ and $\mathbf{v}$, and let $\|\mathbf{u}\|^2 = \langle \mathbf{u}, \mathbf{u} \rangle = \sum_i u_i^2$ denote the square of the Euclidean norm of $\mathbf{u}$.

Recall that a set of vectors $\{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ forms an *orthonormal basis* of $\mathbb{R}^m$, if $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ is 0 for $i \neq j$ and is 1 for $i = j$ (this automatically implies that these vectors are linearly independent and span $\mathbb{R}^m$).

Finally, recall that a non-zero vector $\mathbf{v}$ is an *eigenvector* of a square $m \times m$ matrix A corresponding to an *eigenvalue* λ, if $A\mathbf{v} = \lambda \mathbf{v}$.

QUADRATIC FORMS. A *quadratic form* (over $\mathbb{R}$) in m variables is a multivariate polynomial where each term has degree exactly 2. One can always write it as a map from real-valued vectors to real numbers such that a vector $\mathbf{w} \in \mathbb{R}^m$ maps to $\mathbf{w}^\top Q \mathbf{w} = \sum_{i,j=1}^m w_i w_j Q_{i,j}$, where Q is a symmetric $m \times m$ matrix (i.e., $Q_{i,j} = Q_{j,i}$). For example, when Q is the identity matrix, we get $\mathbf{w}^\top Q \mathbf{w} = \sum_i w_i^2 = \|\mathbf{w}\|^2$.

Now a symmetric matrix will always have m real eigenvalues (counted with multiplicity) and, moreover, it will always be *diagonalizable* over $\mathbb{R}$ (see Artin for a proof). Explicitly, this means that we can find an orthonormal basis $\{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ of $\mathbb{R}^m$ and a set of eigenvalues $\lambda_i \in \mathbb{R}$ such that $Q\mathbf{v}_i = \lambda_i \mathbf{v}_i$ for all i. In addition,

Fact 1 *For any vector* $\mathbf{u}$*, the orthonormality of the* $\mathbf{v}_i$*'s implies*

$$\mathbf{u}^\top Q \mathbf{u} = \sum_{i=1}^m \lambda_i \langle \mathbf{u}, \mathbf{v}_i \rangle^2 \quad \text{and} \quad \|\mathbf{u}\|^2 = \langle \mathbf{u}, \mathbf{u} \rangle = \sum_i \langle \mathbf{u}, \mathbf{v}_i \rangle^2$$

Assume now that $\langle \mathbf{u}, \mathbf{v}_i \rangle = 0$ for all i corresponding to the large eigenvalues of Q. Then the Fourier decomposition above allows us to obtain the following upper bound on $\mathbf{u}^\top Q \mathbf{u}$.

Corollary 4 *Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m$ be the eigenvalues of Q. And assume that $\langle \mathbf{u}, \mathbf{v}_1 \rangle = \ldots = \langle \mathbf{u}, \mathbf{v}_j \rangle = 0$ for some $j \geq 0$. Then*

$$\mathbf{u}^\top Q \mathbf{u} \leq \lambda_{j+1} \cdot \|\mathbf{u}\|^2 \tag{2.12}$$

In particular, for any $\mathbf{u}$ *we have* $\mathbf{u}^\top Q \mathbf{u} \leq \lambda_1 \cdot \|\mathbf{u}\|^2$.

Proof: By Fact 1,

$$\mathbf{u}^\top Q \mathbf{u} = \sum_i \lambda_i \langle \mathbf{u}, \mathbf{v}_i \rangle^2 = \sum_{i>j} \lambda_i \langle \mathbf{u}, \mathbf{v}_i \rangle^2 \leq \lambda_{j+1} \sum_{i>j} \langle \mathbf{u}, \mathbf{v}_i \rangle^2 = \lambda_{j+1} \|\mathbf{u}\|^2$$

■

We notice that the Corollary above follows from a more general Courant-Fischer theorem, which states that in fact $\lambda_{j+1} = \max_{\mathbf{u} \in U_j} \frac{\mathbf{u}^\top Q \mathbf{u}}{\|\mathbf{u}\|^2}$, where U_j is the space of all vectors orthogonal to the first j eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_j$.

Fourier Decomposition of the Hypercube. We will be using a particular matrix A — the adjacency matrix of an n-dimensional hypercube $\mathcal{H} = \{0,1\}^n$. That is, A is a $2^n \times 2^n$ dimensional 0-1 matrix, with entries

$$A_{x,y} = \begin{cases} 1 & \text{if } x \text{ and } y \text{ differ in exactly one position} \\ 0 & \text{otherwise} \end{cases}$$

We consider A as an operator on the 2^n-dimensional vector space V consisting of vectors with positions indexed by the strings in $\mathcal{H}$. Typically, we will use $\mathbf{u}(y)$ to refer to position $y \in \mathcal{H}$ in the vector $\mathbf{u}$.

For two strings in y, z in $\{0,1\}^n$, let $y \cdot z$ denote their inner product modulo 2, i.e. the parity of the number of positions on which they are both 1. We denote by $weight(z)$ the number of positions of z which are equal to 1. We need the following fact explicitly telling us the eigenvectors and the eigenvalues of A.

Fact 2 *A has an orthonormal basis of eigenvectors $\{\mathbf{v}_z : z \in \{0,1\}^n\}$, where the*

eigenvalue of $\mathbf{v}_z$ *is* $\lambda_z = n - 2 \cdot weight(z)$*, and the value of* $\mathbf{v}_z$ *at position* y *is*

$$\mathbf{v}_z(y) = \frac{1}{\sqrt{2^n}} \cdot (-1)^{z \cdot y} \tag{2.13}$$

The basis $\{\mathbf{v}_z\}$ is often called the *Fourier basis* related to the matrix A. The coefficients $\langle \mathbf{u}, \mathbf{v}_z \rangle$ are then called the *Fourier coefficients* of $\mathbf{u}$. From the above fact and from Corollary 4, we get the following useful lemma.

Lemma 4 *Assume* $\{\mathbf{v}_z : z \in \{0,1\}^n\}$ *are the eigenvectors of* A *as above, and let* $\mathbf{u}$ *be a vector orthogonal to all the* $\mathbf{v}_z$*'s corresponding to* z *with* $weight(z) < t$*. Then*

$$\mathbf{u}^\top A \mathbf{u} \leq (n - 2t) \cdot \|\mathbf{u}\|^2$$

In particular, for any $\mathbf{u}$ *we have* $\mathbf{u}^\top A \mathbf{u} \leq n \cdot \|\mathbf{u}\|^2$.

Chapter 3

Definitions and Discussion

In this section, we define and discuss the central concepts in our study: Exposure-Resilient Functions (ERF's) and All-Or-Nothing Transforms (AONT's). Our definitions are extremely natural and simple. We also show that ERF's and AONT't have numerous applications in many different areas, making them indeed fundamental cryptographic primitives.

3.1 Exposure-Resilient Functions

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 (see Figure 1-2). Formally,

Definition 11 *A polynomial time computable function $f : \{0,1\}^n \rightarrow \{0,1\}^k$ is ℓ-*ERF (exposure-resilient function) *if for any $L \in \{\ell^n\}$ and for a randomly chosen $r \in \{0,1\}^n$, $R \in \{0,1\}^k$, the following distributions are indistinguishable:*

$$\langle [r]_{\bar{L}}, f(r) \rangle \approx \langle [r]_{\bar{L}}, R \rangle \tag{3.1}$$

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

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 in-

distinguishable from a random value. There are three parameters of interest here: ℓ, n, and k. All of them are very important. First of all, the smaller ℓ is, the harder is to satisfy the condition above, since fewer bits are left unknown to the adversary. Thus, we wish to make ℓ as small as possible for a given n. Secondly, k is the number of pseudorandom bits that we get out when the adversary does not see ℓ bits of the input, which we would like to make as large as possible. Thus, 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 *"exposure-resilience"*); and the size of k with respect to ℓ, which we want to be as large as possible (this shows the *"randomness efficiency"*).

ADAPTIVELY SECURE ERF. In the definition of ERF above, the adversary has to "decide in advance" which $(n - \ell)$ bits he is going to observe. This is captured by requiring the security for all *fixed* sets L of cardinality ℓ. However, in many situations (e.g., the problem of gradual key exposure explained in the next section), the adversary has more power. Namely, he can decide which $(n-\ell)$ bits of the secret to learn *adaptively* based on the information that he has learned so far. In the most extreme case, the adversary would decide which bits to observe "one-bit-at-a-time". As we will see, this adversary is indeed much more powerful than the static adversary who decides on the subset L of bits to "miss" in advance. But now we just define formally the corresponding notion of *adaptively secure* ℓ-ERF that would protect even against such adaptive adversaries.

First, an adversary $\mathcal{A}$ having oracle access to a string r is said to be *ℓ-bounded* if he is allowed to adaptively read all but some ℓ bits of r one-bit-at-a-time, depending on his input and the bits of r that he read so far. We denote such an adversary by $\mathcal{A}^r(\cdot)$.

Definition 12 *A polynomial time computable function $f : \{0,1\}^n \to \{0,1\}^k$ is a (perfect, statistical or computational)* adaptive ℓ-ERF (adaptive exposure-resilient function) *if for any ℓ-bounded adversary $\mathcal{A}$, when r is chosen at random from $\{0,1\}^n$*

and R is chosen at random from $\{0,1\}^k$,

$$|\Pr(\mathcal{A}^r(f(r))=1) - \Pr(\mathcal{A}^r(R)=1)| \leq \varepsilon$$

where

- *In the perfect setting $\varepsilon = 0$.*
- *In the statistical setting $\varepsilon = negl(n)$.*
- *In the computational setting $\varepsilon = negl(n)$ for any PPT $\mathcal{A}$.*

Thus, in the above definition $\mathcal{A}$ would try to adaptively examine $(n-\ell)$ bits of r to determine at least something about $f(r)$. And if f is an ℓ-ERF, no ℓ-bounded $\mathcal{A}$ would succeed in distinguishing $f(r)$ from a random string.

We observe that in the perfect setting this definition is *equivalent* to that of an ordinary perfect ℓ-ERF. Indeed, no matter how, why and which $(n-\ell)$ bits of r were examined by $\mathcal{A}$, once the remaining ℓ bits of r are chosen at random, the definition of perfect ℓ-ERF says that $f(r)$ is truly random over $\{0,1\}^k$ (even conditioned on the observed $(n-\ell)$ bits). Thus, adaptivity does not help the adversary in the perfect setting (because the definition of a perfect ERF is by itself very strong!). As we will see, in the statistical setting there is a very big difference between the adaptive and the non-adaptive notions: if not that much with the parameters achieved, but with the difficulty of constructing adaptive ERF's as compared to ordinary ERF's. And once we have good statistical ERF's, computational ERF's will be easy to construct both in the static and in the adaptive settings.

Computational ERF vs. PRG. Assume we have a (computational) ℓ-ERF $f : \{0,1\}^n \to \{0,1\}^k$, where $k > n$. This can be viewed as a particularly strong form of a pseudorandom generator (PRG, see Section 2.4). In other words, not only f stretches n random bits into k pseudorandom bits, but the k output bits remain pseusorandom even when *any* $(n-\ell)$ *bits of the seed are revealed.* Thus, such ERF can be called an "exposure-resilient PRG". Not surprisingly, we will use regular PRG's as one of the building blocks in constructing such computational ERF's.

3.2 Applications of ERF

PROTECTING RANDOM SECRETS. As an immediate general application to the partial key-exposure problem, ℓ-ERF $f : \{0,1\}^n \to \{0,1\}^k$ allows one to represent a random secret $R \in \{0,1\}^k$ in an "exposure-resilient" way. Namely, instead of storing and using R as the secret, we pick and store a random $r \in \{0,1\}^n$, but use $f(r)$ as our secret. Since $f(r)$ and R are indistinguishable, our underlying application is not going "to know the difference". In fact, even if the adversary learns all but ℓ bits of r, the secret $f(r)$ is still indistinguishable from a random value.

On a theoretical level, we can almost always assume that our secret is a truly random string (for example, the random coins of the key generation algorithm). Thus, in principle ℓ-ERF's can solve the general partial key exposure problem. In practice, however, this is going to be efficient only if a "natural representation" of the secret is a truly random string. As we saw in Section 2.5, this indeed often happens in the setting of private-key cryptography (and sometimes even in public-key cryptography), and gives rise to many more specific applications, some of which we describe next.

EXPOSURE-RESILIENT PRG'S AND ONE-TIME PAD. As another immediate application which we already observed at the end of last section, ERF's allow us to obtain a much stronger form of pseudorandom generator (especially when $k > n$), which not only stretches n bits to k bits, but remains pseudorandom even when *any* $(n - \ell)$ *bits of the seed are revealed.* As a natural extension of the above application, we can apply it to the one-time private-key encryption. Recall that one-time pad encryption over $\{0,1\}^k$ chooses a random shared secret key $r \in \{0,1\}^n$ and encrypts $x \in \{0,1\}^k$ by a pseudorandom "one-time pad" $G(r)$ (where G is a PRG), i.e. $\mathsf{Enc}(x;\ r) = x \oplus G(r)$. We can make it resilient to the partial key exposure by replacing a PRG G with a ERF f.

EXPOSURE-RESILIENT PRF'S AND SYMMETRIC ENCRYPTION. For the next several application, we assume for convenience that ERF $f : \{0,1\}^k \to \{0,1\}^k$ is length-preserving (we will show in Section 4.3 how to build them based on any one-way

function). Using such f, we show how to obtain *exposure-resilient form of a pseudorandom function family.* 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. Recall that a 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 new secret key is $s \in \{0,1\}^k$, but $f(s)$ is used as an index to a regular PRF.

Other examples of random keys. As we pointed in Section 2.5, there are many other natural examples where the secret key is just a random string: message authentication codes, pseudorandom permutations and block ciphers, keyed hash functions, many discrete-log based cryptosystems.

Gradual exposure of random keys. In fact, we can achieve security even against what we call the *gradual key exposure* problem in the setting with shared random keys. Namely, assume several parties (say, two) want to share a secret key which is just a k-bit random value. And 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. As before, let us let us strore a random $r \in \{0,1\}^k$ and use $f(r)$ in place of R. We know that as long as the adversary misses ℓ bits of r, the system is secure.[1] However, pretty soon (in about $(k-\ell)/b$ weeks) there is a danger that the adversary may know more than $(k-\ell)$ bits of r, which would make $f(r)$ no longer "completely secret". To circumvent thsi problem and to avoid ever changing the secret key, it seems sufficient that both parties periodically (say, with period slightly less than $(k-\ell)/b$ weeks) update their stored key by setting $r_{new} = f(r_{old})$. Since at

[1] Here it makes more sense to talk about *adaptively* secure ERF's.

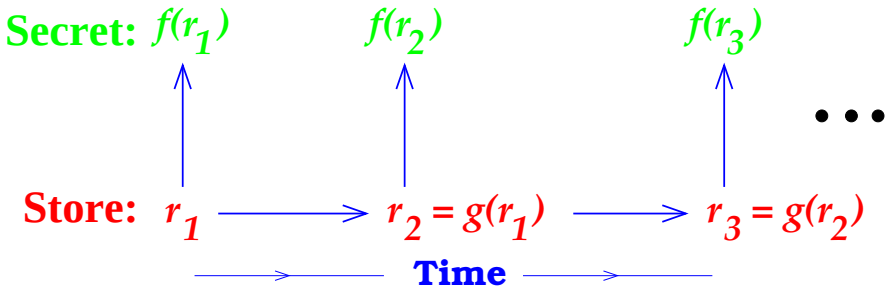

Figure 3-1: Gradual exposure of random keys, or how to maintain a random secret.

the time of each update the adversary did not know at least ℓ bits of our current key r, the value $f(r)$ is still pseudorandom, and thus secure. This idea "almost works".

The problem is that after we changed our "stored" key from r to $f(r)$, and once the adversary starts learning the bits of $f(r)$, he gets the bits of the "actual" key $f(r)$ we used several weeks ago. For some applications, where old transactions are quickly erased or become irrelevant, this might be good, but in general we definitionely do not want the adversary to learn information about our old keys. In some sense, the only thing we achieved is shifting the immediate problem of key exposure by $(k-\ell)/b$ weeks. Luckily, there is a simple fix that makes this idea work as we initially intended. Namely, assume we have a length-doubling ℓ-ERF $\tilde{f} : \{0,1\}^k \to \{0,1\}^{2k}$ (again, we will show in Section 4.3 how to build them). Call f the length-preserving function returning the first k bits of $\tilde{f}$, and by g — the one returning the last k bits of $\tilde{f}$. Now we store a random r and use $f(r)$ as our actual secret. However, at the time we need to synchronously update our key (say, in $(k-\ell)/b$ weeks), we replace r by $g(r)$. Now at the time of each update the adversary misses at least ℓ bits of our current secret, so he has no information about both $f(r)$ and $g(r)$. Moreover, even when he later learns some information about our new secret $g(r)$ (even *all of it*), he still gets no information about $f(r)$, i.e. the actual secret used in all the current transactions. Hence, parties agree on a random secret key *only once*, even if the adversary continuously learns more and more of the (current) secret! This mechanism is illustarted in Figure 4-2, where the timeline shows what is being currently stored, and what is used as a current "actual" secret.

MAINTAINING A (PSEUDO)RANDOM SECRET. The solution above has another application. Namely, it allows one party to maintain a (pseudo)random secret that keeps changing (while staying pseudorandom to the adversary), despite the adversary able to continuously learn more and more bits of whatever we store (but at a bounded rate). As before, we store r, use $f(r)$ as our maintained pseudorandom secret, and before the adversary learns too many bits of r, we let $r_{new} = g(r_{old})$. We will see one application of this is Section 3.4.

AGREEING ON A SECRET KEY. This is one of the applications of t-resilient functions (e.g., perfect ERF's) suggested by [10], which extends to any ERF. Assume that two parties Alice and Bob want to agree on a random string of length k. Ordinarily, Alice can choose a random string R and send it to Bob. Unfortunately, there is an eavesdropper Eve who can listens to the communication channel and may learn some of the bits transmitted by Alice. Alice and Bob do not know which bits were observed by Eve, but they know that with high probability Eve did not learn more than a δ fraction of the transmitted bits. Assume we have an ℓ-ERF $f : \{0,1\}^n \to \{0,1\}^k$ with $\ell/n \leq 1 - \delta$. Then Alice can pick a random $r \in \{0,1\}^n$ and send it to Bob. The actual shared random string will $f(r)$, about which Eve will have "no information" since he misses at least $(1 - \delta)n \geq \ell$ bits of r.

COIN-FLIPPING IN SYNCHRONOUS NETWORKS WITH BROADCAST. This is one of the original motivations of Chor et al. [20] . Unfortunately, if only applies to perfect ℓ-ERF's. Consider a synchronous network where n players wish to collectively flip a random k-bit string, and only a broadcast channel[2] is available for communication. Assume also that up to $(n - \ell)$ of the players can be faulty, and our protocol should be resilient against that. The simplest possible solution would be for each player i to flip a random bit r_i and to broadcast it to all the other players. The resulting k-bit output will be $f(r_1, \ldots, r_n)$ for some fixed $f : \{0,1\}^n \to \{0,1\}^k$. It is easy to see that the protocol is resilient to any ℓ faulty players if and only if f is a perfect ℓ-ERF (i.e.,

[2]This means that a player can send a message to all other players, and all the players are assured of getting the same message.

$(n-\ell)$-resilient function of [20]).

We also remark that while this application does not apply to statistical and computational ERF's, it does apply to their stronger counter-parts. Namely, we can use almost $(n-\ell)$-resilient functions of [39] (the output will then be statistically close to uniform) that we construct in Section 4.4.1. Alternatively, if we use what we call computational $(n-\ell)$-resilient functions (that we define and construct in Remark 1), the resulting output will be computationally close to uniform. Overall, we can say that this application applies to (perfect, statistical or computational) $(n-\ell)$-*resilient functions*, that is: a) the adversary can *fix* any $(n-\ell)$ bits of r to any string he desires, b) the remaining ℓ bits of r are set at random, and c) the resulting output $f(r)$ is still "close" to a random k-bit string (where the meaning of "close" depends on the notion of $(n-\ell)$-resilient function we use).

All-Or-Nothing-Transforms. Finally, in Section 5 we show how to construct AONT's using ERF's.

3.3 All-Or-Nothing Transforms

Definition 13 *A* randomized *polynomial time computable function* $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^p$ *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 $L \in \binom{s}{\ell}$, any $x_0, x_1 \in \{0,1\}^k$ we have*

$$\langle x_0, x_1,\ [T(x_0)]_{\bar{L}} \rangle \approx \langle x_0, x_1,\ [T(x_1)]_{\bar{L}} \rangle \tag{3.2}$$

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.

The above definition is "indistinguishability" based and follows the general methodology from Section 2.3. Indeed, for each fixed L the experiment on x consists simply of outputting $[T(x)]_{\bar{L}}$. In particular, 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 D), cannot compute β satisfying some relation $\mathcal{R}(x, \beta)$ "significantly better" than without z at all. Thus, 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. We also remark that Boyko [16] gave two separate definitions of semantic security and indistinguishability for AONT (with random oracles), and proved essentially identical theorems for both of his definitions. The general equivalence of the definitions (together with the efficiency of both reductions in Section 2.3) shows that one of these proofs was not necessary, and further justifies the usefulness of Theorem 1.

Comparison with earlier definitions. The definition given above generalizes and simplifies (because there are no random oracles) the formal definition for secret-only AONT given by Boyko [16] (refining an earlier definition of Rivest [51]) 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 y_2 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 also observe that if $y = (y_1, y_2)$ and $L \in \binom{s}{\ell}$, we have notationally that $[y]_{\bar{L}} = ([y_1]_{\bar{L}}, y_2)$. This is informally illustrated in Figure 3-2 and should be compared with the special case of *secret-only* AONT from Figure 1-1.

We note that our generalized notion of AONT solves the problem of partial key

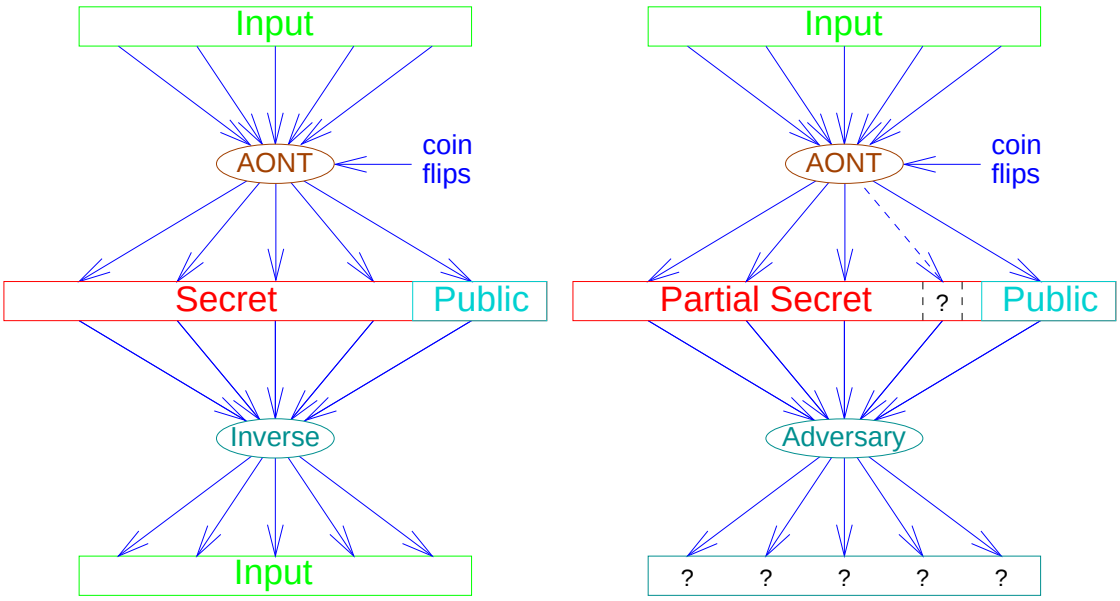

Figure 3-2: All-Or-Nothing Transform (with secret and public outputs).

exposure and also remains equally applicable to all the other known uses of the secret-only AONT. In addition, we will argue that it gives us more flexibility and also allows us to characterize the security of our constructions more precisely. More specifically, the motivations for potentially having a public part is the following. In the earlier definitions of AONT (which were secret-only), 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 first half of the transform, we are completely secure, but become totally insecure if the adversary learns the entire first half. This seems like a very reasonable solution to the key leakage problem; we will simply protect as hard as we can the first half of the transform, while the second half

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 first 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 first half remain hidden. To summarize, in our definition 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 only the secret part that is important to protect.

We now 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 well protected memory (like a smartcard), and still give us 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 only a secret part. We also point again that our generalized notion of AONT naturally suffices for all the applications of AONT that we are aware of.

We also remark that the special case of perfect AONT's was implicitly mentioned by Bennett et al. [10]. See Section 5.1.

Previous constructions. Boyko [16] showed that, *in the random oracle model*, the following so called "optimal asymmetric encryption padding" (OAEP) construction of [8] is a (secret-only) ℓ-AONT (where ℓ can be chosen to be super-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

$$u \quad = \quad G(r) \oplus x \tag{3.3}$$

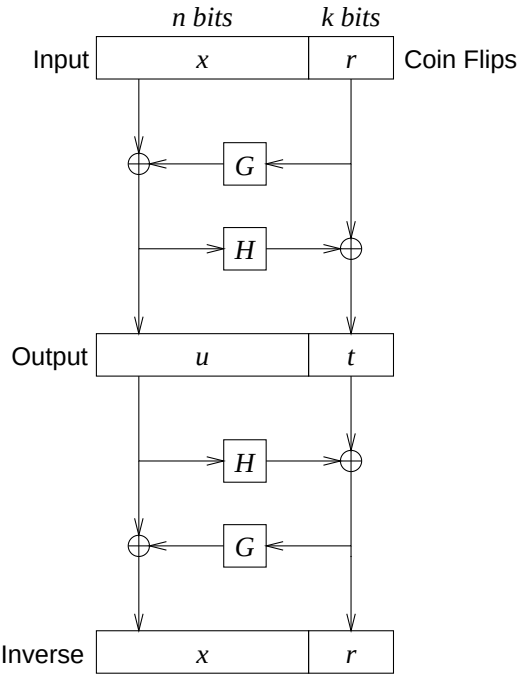

Figure 3-3: Optimal Asymmetric Encryption Padding.

$$t \;=\; H(u) \oplus r \tag{3.4}$$

We note that the inverse $I(u,t) = G(H(u) \oplus t) \oplus u$. This construction is illustrated in Figure 3-3. We also remark that Boyko's work was the first formal treatment of the AONT, stimulated a lot of subsequent research and achieved essentially the best possible AONT's in the Random Oracle model. No provable constructions of AONT based on standard assumptions were previously known.

Adaptively Secure AONT. As for the definition of ERF, we can talk about *adaptively secure* AONT*'s*. In other words, in the ordinary AONT's the adversary has to "decide in advance" which $(s-\ell)$ bits of the (secret part of) the output he is going to observe. This is captured by requiring the security for all *fixed* sets L of cardinality ℓ. While interesting and non-trivial to achieve, in many applications the adversary potentially has the power to choose which bits to observe *adaptively*. Namely, the choice of which bit(s) to observe next may partially depend on which bits the adversary has already observed. For example, if a secret is a large document, the adversary may

try to first steal the first couple of pages that have the table of contents. Based on that, the adversary will know which parts of the document are really important, and then target his attention to stealing the few "important" pages, whose identity the adversary would not know if he could only steal several pages "in one shot". Taken to the most extreme, we can allow this adaptive adversary to read the bits of the secret "one-bit-at-a-time", as long as he misses at least ℓ bits.

As before when talking about ERF's, we will capture this by having an *ℓ-bounded* adversary $\mathcal{A}$, who will have oracle access to a string $y = (y_s, y_p)$ (which could be generated by some probabilistic experiment). $\mathcal{A}$ can read entire "public" part y_p and is allowed to adaptively read all but some ℓ bits of the "secret" part y_s one-bit-at-a-time (possible based on his regular input). As before, we denote such an adversary by $\mathcal{A}^y(\cdot)$.

Definition 14 *A* randomized *polynomial time computable function* $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^p$ *is a (perfect, statistical or computational)* adaptive ℓ-AONT (adaptive 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 $x_0, x_1 \in \{0,1\}^k$ and any ℓ-bounded adversary $\mathcal{A}$,*

$$\left|\Pr(\mathcal{A}^{T(x_0)}(x_0, x_1) = 1) - \Pr(\mathcal{A}^{T(x_1)}(x_0, x_1) = 1)\right| \leq \varepsilon$$

where

- *In the perfect setting $\varepsilon = 0$.*
- *In the statistical setting $\varepsilon = negl(s + p)$.*
- *In the computational setting $\varepsilon = negl(s + p)$ for any PPT $\mathcal{A}$.*

Equivalently, in the above definition we can have $\mathcal{A}$ adaptively examine some bits of $T(x_i)$ to determine at least something about a randomly chosen i. And if T is

an ℓ-AONT, no ℓ-bounded $\mathcal{A}$ would succeed in predicting i correctly with probability significantly more than $1/2$.

Again, we observe that in the perfect setting this definition is *equivalent* to that of an ordinary perfect ℓ-AONT. Thus, adaptivity does not help the adversary in the perfect setting (because the definition of a perfect AONT is by itself very strong!). As with ERF's, however, in the statistical and computational settings there is a very big difference between the adaptive and the non-adaptive notions.

AONT's vs. ERF's. The notions of ERF and AONT are closely related with the following crucial difference. In 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. Thus, the security is "average-case", which allows us to have a *deterministic* f. In AONT, the secret is an *arbitrary* x, which can be represented in an "exposure-resilient" way by storing $T(x)$ instead. Thus, the security is "worst-case", and, as a result, AONT *must be* randomized. To summarize, 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, we will show that 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. These issues are summarized once again in Table 3.1 (see also Figures 1-1 and 1-2).

We also remark that *perfect* AONT's and ERF's are related even more closely, with AONT being more general. See Section 5.1 for more on this relation.

AONT's vs. Error-Correcting Codes. It is also interesting to compare the notion of an AONT with a somewhat opposite notion of an *error-correcting code* (ECC). Recall, an error-correcting code of minimal distance $2d$ *deterministically* stretches k input bits to n output bits (called the *encoding* of the input), such that erasing any d bits of the output still allows one to recover the k-bit input. In particular, observing any $(n-d)$ bits of the output allows one to recover the input. Thus, the larger the distance $2d$ is (and one always tries to maximize the distance when con-

Issue	AONT's	ERF's
Secret	Any x	(pseudo)Random $\mathsf{ERF}(r)$
Store	$\mathsf{AONT}(x)$	Random r
Function	(must be) Randomized	Deterministic
Security	"Worst-Case"	"Average-Case"
Length	$\lvert\mathsf{AONT}(x)\rvert \geq \lvert x\rvert$	Can have $\lvert r\rvert < \lvert\mathsf{ERF}(r)\rvert$
Common	Store secret in an "exposure-resilient" way	

Table 3.1: Comparison of AONT's and ERF's.

structing ECC's), the less "exposure-resilient" the ECC is. And this is what we want from ECC's, since their main use is to tolerate a lot of errors in the encoding. In contrast, the objective of an AONT is to give *no information* about the input when one misses just few bits of the output. Namely, missing any few bits of the (secret part of the) output gives no information about the input. Thus, a good AONT would be terrible for error-correction purposes and vice versa. Curiously enough, though, in Section 5.1.1 we will construct perfect AONT's using some good ECC's.

3.4 Applications of AONT

In the applications below, let x be the "secret entity" (clear from each application), T be an ℓ-AONT and $y = (y_1, y_2) \leftarrow T(x)$.

Partial Key Exposure. This is our original motivation. Given a secret x, our can store y instead. This way, the adversary who can learn all but ℓ bits of (the secret part of) y has no information about the "actual" secret x. This works for secrets of arbitrary structure and applies as a black-box to *any cryptographic application*. The price we pay is a slight storage blow-up (which is inevitable since we should be able to recover x from y), and the computation of x from y that we have to perform when

we use x. If the underlying AONT is efficient and not very long, while the application is very "intensive", this extra-work will not be significant, but the system becomes "exposure-resilient".

GRADUAL KEY EXPOSURE OF ANY KEYS. This is a similar scenario to the gradual key exposure of random key, that was considered in Section 3.2. There two parties wanted to maintain a common random key despite the adversary learning more and more bits of whatever we store (but doing so at a limited *rate*). We presented a solution to that problem where the parties synchronously update their current key using a good ERF. Now assume that one party (this clearly extends to more parties) has a *particular* secret x, and the adversary, as before, can learn more and more bits of the secret, but at a bounded *rate*. We do not want to *ever change* x despite this capability of the adversary. As in the regular key-exposure setting above, we simply store the AONT y of x. However, after there is a danger that the adversary knows all but ℓ bit of (the secret part of) y, we erase y and store a brand new AONT y' of x, and so on. In other words, we store a brand new AONT of x for the period of time when it is safe, and then simply redo it from scratch. Again, we choose our secret x only once and maintain it for an arbitrarily long period of time!

Notice the differences with the solution for the shared random keys. There we kept changing our current key, while maintaining it pseudorandom. Since the key changes over time, this makes sense only for two or more parties, and the difficulty was for the parties to synchronously do "the same thing", so that they still share the same (pseudorandom) secret. Here the philosophy is *not to ever change a specific secret* x *despite the gradual key exposure*. Thus, it makes sense even for one party and does not require any synchronization: the parties can apply the AONT to x independently with different local randomness and at different times. In particular, we can solve the gradual key exposure for random secrets using the same methodology and without ever (synchronously) changing the (random) secret.

On the other hand, the solution for shared secret keys can also be easily modified to solve gradual key exposure of *any* secret x (even for one player). Namely, we

noticed in Section 3.2 that the solution allowed us to maintain a random secret r that keeps changing (while staying pseudorandom), but the adversary has no information about the "current" r. Well, in addition to "storing" this r, we also store $x \oplus r$ (which allows us to reconstruct x from r and $x \oplus r$). Moreover, $x \oplus r$ can even be made *public*. Now, each time we change r we also change $x \oplus r$ accordingly. Despite conceptually being slightly more difficult than the solution using AONT's, it has the advantage of being randomness efficient. In other words, after we select the initial random bits to store r, we do not need any more random bits (e.g., to compute a brand new AONT) in order to "maintain" x.

ENHANCING THE SECURITY OF BLOCK CIPHERS. This was the original motivation of Rivest [51]. Rivest observed that typical encryption schemes (e.g., most block ciphers) have a fixed "block length" ℓ that they operate on. In other words, a (potentially long) message x is split into blocks of size ℓ and each block is encrypted (independently, or in some kind of "chain" mode). Unfortunately, in most such schemes it is possible to recover a block of the original message by correctly decrypting a single block of the ciphertext. As a result, the most typical attack on such block ciphers is a brute-force attack. The adversary goes through all the keys and tries to decode the encryption block corresponding to a block of the priginal message. If he gets something "meaningful",[3] he knows that the key is correct and breaks the whole cipher (in particular, decrypts the whole message by only looking at a single block).

Informally speaking, AONT's seem to allow us to slow down this brute-force attack by a factor equal to the number of blocks b. In particular, it forces the adversary to look at *all* the blocks. Namely, we first transform x by applying an AONT into $y = (y_1, y_2)$. We then send the public part y_2 "in the clear", and apply the standard block-by-block encryption to the secret part y_1. The recipient simply decrypts all the blocks and recovers x by inverting the AONT. Now, unless the adversary tries to decrypt *all the blocks*, no information about x is revealed. Hence, the brute-force

[3] In practice, it typically very easy to see if the decryption makes sense. Alternatively, the adversary often can make the system encrypt some message that adversary knows. Then he knows the decryption and simply tries all possible keys until a match is found.

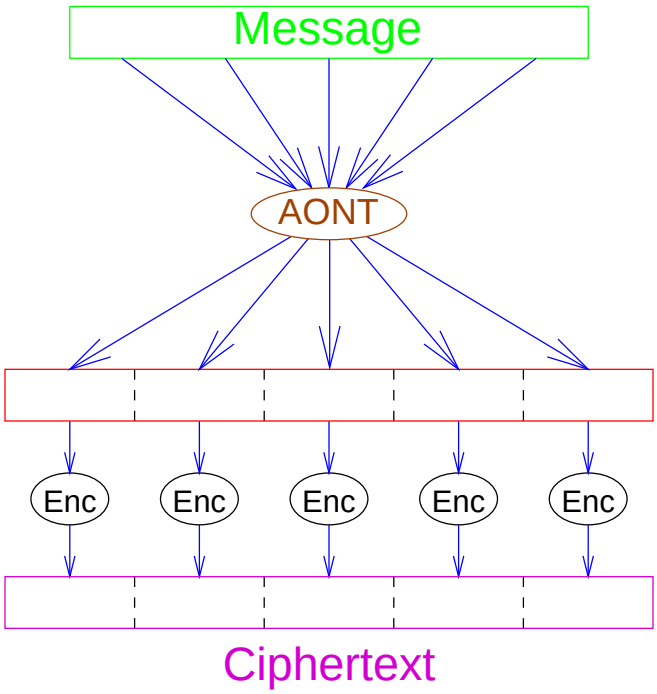

Figure 3-4: AONT's to enhance security of block ciphers.

attack is indeed slowed down by the factor of b, without changing the size of the secret key. This is particularly important when the key length is constrained by the underlying application to be only marginally secure, making it feasible for the adversary to go through all the keys. This paradigm is illustrated in Figure 3-4.

Desai [21] looked at Rivest's suggestion from a formal standpoint. On the one hand, he formally defined the notion of "non-separability of keys" which models the adversary's inability to gain any information about the secret key without decrypting every block of the ciphertext. On the other hand, Desai observed the the "standard" notion of the AONT does not seem to suffice (if used as suggested above) in order to achieve non-separability of keys. For example, if the AONT's has a lot of redundancy[4], by decripting a block of the cyphertext it might be easy to tell if the decryption is part of the AONT's output or not. If it is not, the adversary knows that the key he is examining is incorrect, and can move to the next key very quickly. On a positive

[4]which a good AONT should *not* have, since this decreases its exposure-resilience.

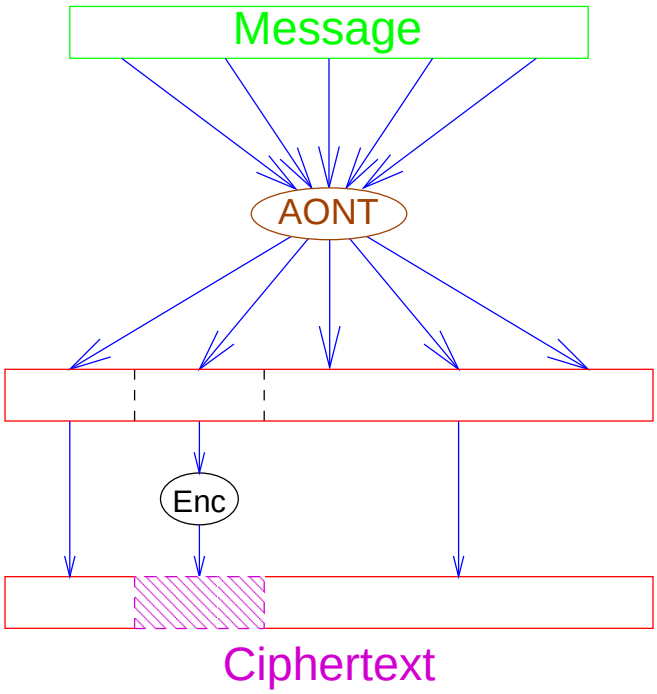

Figure 3-5: AONT's to enhance efficiency of block ciphers.

side, Desai proves that if one uses slightly stronger notion of an AONT (roughly, the output of the AONT in inistinguishable from a random string), the resulting block cipher indeed enjoys non-separability of keys.

We remark on extra flexibility of not restricting ourselves to using a secret-only AONT. Indeed, now the length of y_1 could be much smaller than the length of x (which is impossible for a secret-only AONT), but much larger than ℓ. This way we need to perform *fewer* encryptions/decryptions, while the security is much *higher* than earlier. For more efficiency considerations, see the next application.

Enhancing the efficiency of block ciphers. This is a "dual" application to the above proposed by Matyas, Peyravian and Roginsky [41]. As before, instead of splitting x into blocks, we compute the AONT $y = (y_1, y_2)$ of x and send the public part y_2 in the clear. Now, however, we encrypt only *one* (arbitrary) ℓ-block of y_1 and send all the other bits of y_1 in the clear. The recepient decrypts the single encryption and recovers x as before. This paradigm is illustrated in Figure 3-5.

If the encryption is really secure, the adversary gets no information about x unless he decrypts the single encrypted block. As the result, we securely encrypted a *long message* by performing an actual enctypion of only a *single short* message (of length ℓ). This is particularly useful in several situations. The obvious such situation is when the "base" block encryption is slow or expensive to perform. Another such situation is when the base encryption greatly expands the output (in particular, significantly more in proportion to the AONT) [52]. This way the overall output of our encryption will be considerably less than before. Yet another situation is that of remotely keyed encrypion, when the part of the system that contains the secret key is separate (for example, it resides on a smartcard), and bandwidth considerations make it prohibitive to send the entire long message to be encrypted block by block [14, 37]. Now, irrespective of the length of x, the system needs to encrypt a single short block which dramatically reduces the communication. This is illustrated in Figure 3-6.[5]

Comparing the latter two applications, there is a clear efficiency/security tradeoff that we can exploit if we use AONT in the manner suggested above. The more blocks of y_1 we encrypt, the more secure the system is (where in any event it is at least as secure as the original "naive" block cipher), but the less efficient the system potentially becomes (where in any event it is almost as efficient as it used to be). However, essentially any setting of parameter (provided we use a good AONT) will imporve both the efficiency and the security.

We also remark that these applications require *computational* AONT's. Indeed, they only make sense when ℓ is smaller than the length of the original message x, and we will show that this is possible only in the computational setting.

"Chaffing and Winnowing". Another interesting way to encrypt the data using an AONT and a special kind of message authenticaion code (MAC) way suggested by Rivest [52]. Namely, assume that the sender and the recipient share a very short authentication key, which is used by the sender to authenticate the messages sent to

[5]We remark that Bellare and Boldyreva [4] made a formal "sanity check" and showed that if we use a semantically secure encryption scheme to encrypt a block of the AONT(x), the resulting scheme remains semantically secure.

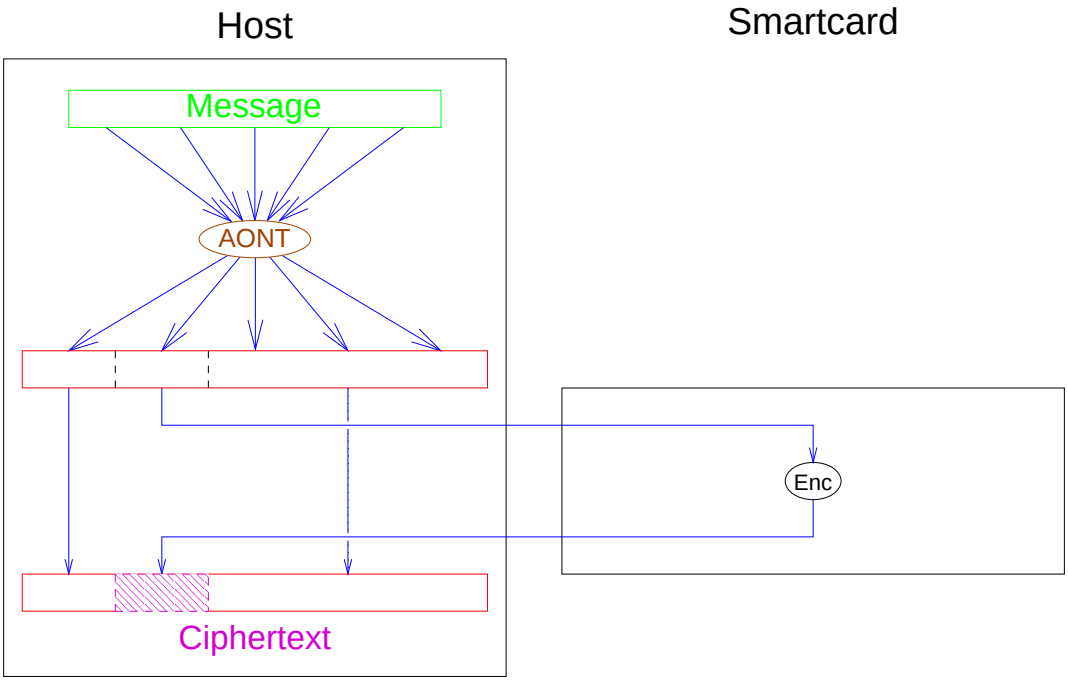

Figure 3-6: AONT's to perform remotely keyed encryption.

the receiver. In other words, the sender applies some hash function (MAC) to the message and the short secret key to get a "tag" for the message. It then sends in the clear the message and the tag. The recipient can then verify that the tag really corresponds to the message. On the other hand, we require that the eavesdropper can neither generate a tag for a given message, nor (which is more important here) verify if a given tag corresponds to a given message. Very efficient MAC's satisfying these properties exist (e.g., [5, 12, 6, 27]), and, in fact, any pseudorandom function (PRF) will work as well.

As before, we apply an AONT to our message x, send the public part in the clear, and split the secret part into b blocks of length ℓ. Call these block $v_1, \ldots, v_b$, and let $t_i = tag(v_i)$. The sender picks b random blocks $v'_1, \ldots, v'_b$ and makes "bogus" tags $t'_1, \ldots, t'_b$ which the adversary cannot tell apart from the correct tags, but the recipient can. Now, for each i the sender sends (in random order) both $\langle i, v_i, t_i \rangle$ and $\langle i, v'_i, t'_i \rangle$. The recipient can "throw away" all the random blocks (since their tags do not check) and recover the message by inverting the AONT. The adversary, on

the other hand, has 2^b choices of which subset of blocks is "relevant". It sounds plausible that unless the adversary tries the unique correct combination, he obtains no information about the message. This suggested encryption is interesting in a sense of not performing a "conventional" encryption, and sending all the information "in the clear" (the problem for the eavedropper is to tell which information is "relevant").

While an interesting suggestion that might be useful in practice, it was observed by Bellare and Boldyreva [4] that the security of this encryption does not seem to follow from the mere definition of an AONT. On the other hand, they showed that if Rivest's suggestion is applied on the *bit-wise* rather than the block-wise level (which was also one of the suggestions of Rivest), this will indeed produce a good encryption. More specifically, we send in the clear all the blocks $v_2, \ldots, v_b$ except for the first block v_1. We then split v_1 into individual bits and for each such bit c send $\langle c, tag(c)\rangle$, $\langle 1-c, \text{"garbage"}\rangle$ in random order. Viewed from a different angle, this is yet another application of the paradigm of Matyas [41] to apply an AONT to the message and to encrypt only the first block (see Figure 3-5). In this case the encryption of the first block is performed by using Rivest's "chaffing and winnowing" (which was shown to be a semantically secure encryption by Bellare and Boldyreva [4]).

GAP SECRET SHARING. This connection was noticed by Rivest [51], even though for a much weaker definition of an AONT. 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" secret sharing scheme*. For some secret x, we apply the AONT to obtain a secret output y_1 and a public output y_2. Here, we think of y_2 as being a common public share that is being 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 the 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 information" about the secret. 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. Notice also that by splitting (in an arbitrary way) our s 1-bit shares into $n = s/\ell$ groups of size ℓ each, we get a traditional $(n-1, n)$-secret sharing scheme, where all n participants can recover the secret, but no information is leaked even if one ℓ-bit share is missing. While it is quite easy to construct such "threshold" schemes (especially, $(n-1, n)$-secret sharing; e.g., [54, 38]), our setting is considerably more difficult since the shares are only 1-bit long, and we should be secure against *any* coalition of $(s - \ell)$ players.

SECURE COMMUNICATION. This application is similar to the application of the ERF's for establishing a shared random key in Section 3.2. As in that case, Eve learns some of the bits that Alice sends to Bob, Alice knows that he learns at most a fraction δ of the bits, but does not know which. To send a k-bit message x, Alice simply applies an AONT to x and sends the result y to Bob. Bob recovers x by inverting the AONT. Eve, on the other had, gets no information about x provided $\ell/s \leq 1 - \delta$ (where s is the length of the secret part of the AONT).

SIMULTANEOUS AND FAIR MESSAGE EXCHANGE. This informal application was suggested by Boyko [16]. Assume Alice and Bob want to simultaneously exchange secrets x_a and x_b of the same length over an asynchronous communication channel. However, none of them wants to send his or her secret first. Here we assume that the players are honest except they can stop the protocol at any point of their choice. Alternatively, one can think that there is a danger that the communication channel is not reliable and may fail at an arbitrary point during the protocol. Thus, Alice and Bob do not want to have a point in the protocol where one of them reveals significantly more about his or her secret than the other. Here is a nice solution using AONT's. Alice and Bob compute AONT's of their secrets: $y_a = T(x_a)$, $y_b = T(x_b)$. They exchange the public parts first in any order. Then they start exchanging the secret parts of y_a and y_b bit by bit. Assuming informally that the only attack on the AONT is the exhaustive search over the bits not yet received, at any point the search space of Alice and Bob differ by at most a factor of 2.

Chapter 4

Exposure-Resilient Functions (ERF)

In this section we give constructions of exposure-resilient functions (ERF's). First, we describe perfect ERF's and their limitations. In particular, ℓ must be at least $n/2$ for $k > \log n$. Then, on our way to building computational ERF's with very strong parameters, we build statistical ERF's, achieving essentially the best possible parameters (i.e. $\ell \approx k$ for any k) and surpassing the impossibility results for perfect ERF's. This construction is perhaps our main technical contribution and uses strong randomness extractors defined is Section 2.7. The construction also demonstrates an *exponential separation* between perfect and statistical ERF's. Indeed, in the perfect setting we are limited to have $\ell > n/2$, while here we can achieve $\ell \approx k$, which can be just slightly super-logarithmic! Finally, we show how to combine our statistical 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 about ERF's are summarized in the following theorem:

Theorem 7 *Assume $\ell \geq n^\epsilon$ (for some arbitrary $\epsilon > 0$). Then*

1. *There is no perfect ℓ-ERF $f : \{0,1\}^n \to \{0,1\}^k$ with $\ell \leq n/2$ and $k > \log n$.*
2. *There exist statistical ℓ-ERF's $f : \{0,1\}^n \to \{0,1\}^k$ with $k = \ell - o(\ell)$, and no statistical ℓ-ERF's with $k > \ell$.*

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

We will also consider a more challenging question of constructing *adaptive* ERF's. We mentioned that in the perfect setting they are equivalent to ordinary ERF's. It will also be clear that we can construct computationally secure adaptive ERF's from statistically secure adaptive ERF's in the same way as for ordinary ERF's (using pseudorandom generators). Therefore, the main difficulty in constructing adaptively secure ERF's will be in the statistical setting. Unfortunately, our statistical construcion of ordinary ERF's will not work in the adaptive setting. However, we show a very efficient *probabilistic construction* of statistical adaptive ERF's. Namely, we will construct an efficiently samplable and computable family of functions, such that a random function in this family will be a good statistical adaptive ERF with overwhelming probability. Once this function is chosen, it *never* has to be changed again and can be published. Similar to the non-adaptive setting, we will be able to achieve essentially optimal $\ell \approx k$ even in the adaptive statistical setting. Overall, we show the existence as well as an efficient probabilistic construction of optimal adaptive ERF's (with essentially the same parameters and implications as in the non-adaptive case summarized in the theorem above).

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 ℓ bits of r 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 the so called $(n - \ell)$-*resilient functions* considered in [20, 10]. As an example, if $k = 1$, the exclusive OR of all n input bits is a trivial perfect 1-ERF (or an $(n-1)$-resilient function). As we will see, for larger values of k it is much harder to construct perfect ERF's.

4.1.1 Construction

Using *binary linear error correcting codes* (see Section 2.6), one can construct the following perfect ℓ-ERF.

Theorem 8 ([20, 10]) *Let M be a $k \times n$ matrix. Define $f(r) = Mr$, where $r \in \{0,1\}^n$. Then f is a perfect ℓ-ERF if and only if M is the generator matrix for a binary error-correcting code of distance $d \geq n - \ell + 1$.*

Proof: Every codeword is a linear combination of some rows of M (i.e., codewords are of the form $u^\top M$ 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 (i.e., having at least d non-zero coordinates). Hence, even after removing *any* $(d-1)$ columns of M, the resulting k "punctured" rows of M are still linearly independent (as they cannot produce the zero vector). Therefore, the remaining $n-(d-1)$ columns have rank k and, as such, span the entire $\{0,1\}^k$. In other words, *the code generated by M has distance d if and only if any $(n-d+1)$ columns of M span $\{0,1\}^k$.*

Now assume that the adversary reads some $(n-\ell)$ bits of a randomly chosen $r \in \{0,1\}^n$. This means that Mr is equal to some particular vector y_0 (known to the adversary) plus the "punctured" matrix M' (obtained by removing $(n-\ell)$ columns of M) multiplied by a random ℓ-bit vector r' (formed by ℓ random bits of r not observed by the adversary). Then f is an ℓ-ERF if and only if $(y_0 + M'r')$ is random in $\{0,1\}^k$, which happens if and only if M' has full rank k. Thus, *f is an ℓ-ERF if and only if any ℓ columns of M span $\{0,1\}^k$.* The lemma follows by comparing the above two equivalences. ∎

Applying this result to any *asymptotically good* (recall, this means $n = O(k)$ and $d = \Omega(n)$) linear code (e.g., the Justesen code), we can get $\ell = (1-\varepsilon)n$, $k = \delta n$, where ε and δ are (very small) constants.

Recall also that by the Singleton bound, we have (for any code) $k \leq n - d + 1$. Thus, we get $k \leq n - (n - \ell + 1) + 1 = \ell$, as expected. Also, it is known that $d \leq n/2$ for $k > \log n$. This implies that we are limited to have $\ell \geq n/2$. On the other hand,

we mentioned in Section 2.6 that at the expense of making n large compared to k, we can achieve $\ell = n - d + 1$ to be arbitrarily close to $n/2$, but can never cross it. We show now that this is not a limitation of our particular construction, but rather an inherent limitation of *perfect* ERF's.

4.1.2 Strong Impossibility Result

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. [20] 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! Friedman [28] and later Bierbrauer et al. [11] generalized this result to any k showing that

Theorem 9 ([28]) *If $f : \{0,1\}^n \to \{0,1\}^k$ is a perfect ℓ-ERF, then*

$$\ell \geq 1 + n \cdot \frac{2^{k-1} - 1}{2^k - 1} = \frac{n}{2} + \left(1 - \frac{n}{2(2^k - 1)}\right) \tag{4.1}$$

In particular, for $k > \log n$ we get $\ell > \frac{n}{2}$, so at least half of the input has to remain secret!

We remark that this result of Friedman was a big breakthrough, affirmatively resolving a famous conjecture posed by [20]. In Section 5.1 we will non-trivially extend this impossibility result to a much more general setting of perfect AONT's. But now we illustrate its tightness. Notice, the bound in Equation (4.1) changes non-trivially only for $k \leq \log n$. For $k > \log n$ the bound stays around $\ell > n/2$. This is not surprising, since we can indeed essentially achieve it by using Theorem 8 with a binary code of distance arbitrarily close to $n/2$, say $d = n(\frac{1}{2} - \delta)$. Such codes exits and can achieve k as large as $n\delta^2$. And in any event, for $k > \log n$ we have $\ell > n/2$ which is quite a strong lower bound. Therefore, it suffices to show the tightness for $k \leq \log n$, and we start from $k \approx \log n$; namely, $n = 2^k - 1$. In this case we show the tightness by applying Theorem 8 to the Hadamard code introduced in Section 2.6. The Hadamard code indeed stretches from k bits to $n = 2^k - 1$ bits and has distance

2^{k-1}. By Theorem 8, we get

$$\ell = n - d + 1 = 2^k - 1 - 2^{k-1} + 1 = 2^{k-1} = 1 + n \cdot \frac{2^{k-1} - 1}{2^k - 1}$$

matching the bound in Equation (4.1). We remark on the explicit form of this function $f : \{0,1\}^{2^k-1} \to \{0,1\}^k$. For $i = 1 \ldots k$, let B_i be the subset of all $j \in [n]$ whose i-th digit in their binary expansion is equal to 1. Then the i-th bit of $f(r_1, \ldots, r_n)$ is simply $\oplus_{j \in B_i} r_j$.

For smaller k, i.e. $k < \log n$, split the n input bits into $n/(2^k - 1)$ blocks of size $(2^k - 1)$.[1] We apply the above (2^{k-1})-ERF to each of the blocks, and output the XOR of the results. Now, if the adversary misses $1 + \frac{n}{2^{k-1}} \cdot (2^{k-1} - 1)$ input bits, he misses 2^{k-1} bits from at least one of $n/(2^k - 1)$ blocks. Therefore, the entire k-bit output of this block is random, and thus, the overall XOR. Hence,

Lemma 5 ([20]) *For $k \leq \log n$, there exist (optimal) perfect ℓ-ERF's $f : \{0,1\}^n \to \{0,1\}^k$, where*

$$\ell = 1 + n \cdot \frac{2^{k-1} - 1}{2^k - 1} = \frac{n}{2} + \left(1 - \frac{n}{2(2^k - 1)}\right)$$

4.2 Statistical ERF

We saw that perfect ERF cannot achieve $\ell < n/2$. 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 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 is is the best we can hope to achieve (up to the lower order term) in the statistical setting due to the following simple lemma:

Lemma 6 *Assume $f : \{0,1\}^n \to \{0,1\}^k$ is a statistical ℓ-ERF with statistical deviation $\varepsilon < \frac{1}{2}$. Then $\ell \geq k$.*

[1] Here we assume for simplicity that $(2^k - 1)$ divides n. If not, we have to take "floors".

Proof: Assume $\ell < k$. Take any L, say $L = [\ell]$. We describe a simple (computationally unbounded) distinguisher D which distinguishes $\langle [r]_{\bar{L}}, f(r) \rangle$ from $\langle [r]_{\bar{L}}, R \rangle$ with probability at least $\frac{1}{2} > \varepsilon$, a contradiction. Given $\langle [r]_{\bar{L}}, B \rangle$, D tries all possible 2^ℓ completions r' for r, and for each of them checks if $f(r') = B$. If the equality holds at least once, D accepts. Clearly, D always accepts when $B = f(r)$, as he will eventually try $r' = r$. If $B = R$, a random string of length k, D succeeds with probability at most $2^{\ell-k} \leq \frac{1}{2}$, since there are at most 2^ℓ possible values $f(r')$ that he tries. The claim follows. ■

We notice that our result (achieving $k = \Omega(\ell)$ or even $\ell - o(\ell)$) is indeed quite surprising. It says that if $r \in \{0,1\}^n$ is chosen at random, no matter which $(n - \ell)$ input bits of r are observed, the value of $f(r)$ (which is $k = \Omega(\ell)$ bits long) is statistically close to the uniform distribution over $\{0,1\}^k$. For example, no non-trivial function of the output bits can depend on only $(n - \ell)$ input bits. We see that we need some very special machinery to solve this problem. It turns out that we need *strong extractors*, as defined in Section 2.7.

4.2.1 Intuition

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 almost ℓ 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 amount of randomness is small enough (say, sufficiently smaller than ℓ, call it d), *we can take it from the input itself!* Hence, we view the first d bits of r (which we will call i) as the randomness used to select the hash function h_i, and the rest of r we call x. The output of our function will be $h_i(x)$. Then observing $(n - \ell)$ bits of r

leaves at least $2^{\ell-d}$ equally likely possible values of x (since $|i| = d$). Now, provided our extractor is good enough, we indeed obtain $k \approx (\ell - d)$ bits statistically close to uniform (in particular, if $d = o(\ell)$, we will get $k = \ell - o(\ell)$!).

A few important remarks are in place before we give precise parameters. First, the adversary may choose to learn the entire i (i.e. it knows h_i). 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 i) *must* be chosen uniformly at random (and then possibly observed by the adversary). For example, Kurosawa et al. [39] consider almost $(n-\ell)$-resilient functions, which are "in between" perfect and statistical ℓ-ERF's. And the distinction was exactly this: they required that for *every* setting of the $(n - \ell)$ bits of r, the value $f(r)$ is statistically close (in fact, even slightly stronger that this) to random when the remaining ℓ bits are chosen at random. In our case, this has to happen only "on average" over the setting of $(n - \ell)$ bits known to the adversary. Partially because of that, the construction of [39], while somewhat better than the construction of perfect ERF's given in Theorem 8, still requires $\ell > n/2$.[2] We, on the other hand, are able to achieve essentially optimal $\ell = k + o(k)$.

4.2.2 Construction using Strong Extractors

Theorem 10 *Assume* $\mathcal{H} = \{h_i : \{0,1\}^n \to \{0,1\}^k \mid i \in \{0,1\}^d\}$ *is a strong* (m, ε)*-extractor, and assume*

$$m + d \leq \ell \tag{4.2}$$

Then there exist efficiently computable statistical ℓ-ERF*'s* $f : \{0,1\}^n \to \{0,1\}^k$ *with statistical deviation* ε*.*

Proof: Define $f : \{0,1\}^n \to \{0,1\}^k$ as follows. Given $r \in \{0,1\}^n$, let $i \in \{0,1\}^d$ be the first d bits of r, and let $x \in \{0,1\}^n$ be equal to r except the first d bits of x

[2] Nevertheless, we later show in Section 4.4 that one *can* have almost $(n - \ell)$-resilient functions with $\ell \approx k$, substantially improving the results of [39] (but our construction is probabilistic).

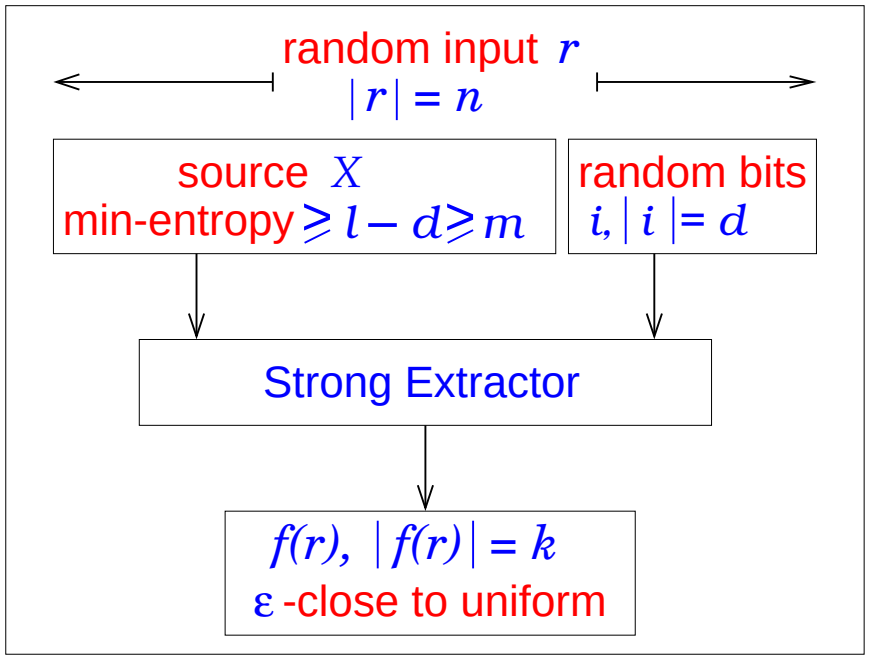

Figure 4-1: Statistical ERF from a Strong Extractor.

are fixed to 0 (we call the last $(n-d)$ bits of x the *valid* bits of x). Let $f(r) \stackrel{\text{def}}{=} h_i(x)$. This simple construction and the sketch of its analysis are illustrated in Figure 4-1.

We now argue that f is an ℓ-ERF with statistical deviation ε. Pick any $L \in \binom{n}{\ell}$. Let r be chosen at random from $\{0,1\}^n$, and define i and x as above. Notice that i and x are independent since we set the first d bits of x to 0. Assume we give the adversary A the value $[r]_{\bar{L}}$. Let w be the valid bits of x in L (i.e., unknown to A), and z be the remaining valid bits of v (i.e., deducible from $[r]_{\bar{L}}$). Notationally we will write $x = w \circ z$. First, since there are $(n-d)$ valid bits of x and at most $(n-\ell)$ of them could be given in $[r]_{\bar{L}}$, we get that $|w| \geq (n-d)-(n-\ell) = \ell - d \geq m$. In other words, A misses at least m valid bits of x. Also notice that z and i subsume $[r]_{\bar{L}}$, so it suffices to show (recalling that $f(r) = h_i(x)$) that for a random $R \in \{0,1\}^k$,

$$\langle i, z, h_i(x) \rangle \cong_\epsilon \langle i, z, R \rangle$$

We will show a slightly stronger statement that this holds for any *fixed* $z = z_0$ (but not i). To summarize, i, w and R are chosen at random, x is set to $w \circ z_0$, and we

want to show that

$$\langle i, h_i(x)\rangle \cong_\epsilon \langle i, R\rangle$$

We are almost done now. Since $|w| \geq m$ and w was chosen at random, we have $x = w \circ z_0$ has min-entropy at least m. The above result now follows from the facts that $\mathcal{H}$ is a strong (m, ε)-extractor family, i is independent from x, and x has min-entropy at least m. ■

The above Theorem gives a very simple connection between extractors and exposure-resilient functions. Since good extractors allow us to almost achieve $k \approx m$, in order to have $k \approx \ell$ Equation (4.2) requires d to be very small. Thus, the most important requirement on $\mathcal{H}$ is that the hash function in $\mathcal{H}$ should be describable by a very short random string. Luckily, strong extractors given in Theorem 4 have this property and yield the following result.

Theorem 11 *There exist statistical ℓ-*ERF *$f : \{0,1\}^n \to \{0,1\}^k$ satisfying:*

1. $k = \ell/6$, *for any* $\omega(\log n) \leq \ell \leq n$.
2. $k = (1-\delta)\ell$, *for any* $\omega(\log^2 n) \leq \ell \leq n$ *(and any constant* $\delta > 0$*).*
3. $k = \ell - o(\ell)$, *for any* $\omega(\log^2 n \cdot \mathrm{loglog}\, n) \leq \ell \leq n$.

Proof: We simply apply Theorem 10 to each of the three extractors from Theorem 4.

1. Take any $\omega(\log n) \leq \ell \leq n$, set $m = \ell/5$ and pick any negligible error ε such that $\omega(\log n) \leq \log(1/\varepsilon) \leq o(\ell)$. Notice that

   $$d + m = 4(m - \log(1/\varepsilon)) + O(\log n) + m \leq 5m = \ell$$

   so we can apply Theorem 10. We get

   $$k = m - 2\log 1/\varepsilon - O(1) \geq \ell/6$$

2. Take any $\omega(\log^2 n) \leq \ell \leq n$, pick any $\delta > 0$, set $\varepsilon = n^{-O(\log n)}$ (i.e., $\log(1/\varepsilon) = O(\log^2 n)$), and $m = (1-\delta)\ell$. Notice that

$$d + m = O(\log^2 n + \log(1/\varepsilon)) + m = O(\log^2 n) + (1-\delta)\ell < \ell$$

so we can apply Theorem 10. We get

$$k = (1-\delta)m - O(\log(1/\varepsilon)) = (1-\delta)^2\ell - O(\log^2 n) \geq (1-2\delta)\ell$$

(now replace δ by $\delta/2$.)

3. Take any $\omega(\log^2 n \cdot \operatorname{loglog} n) \leq \ell \leq n$, set $\varepsilon = n^{-O(\log n)}$, and $m = \ell - O(\log^2 n \log \ell)$. Notice that since $\log m < \log \ell$, we get

$$d + m = O((\log^2 n + \log(1/\varepsilon)) \log m) + m \leq O(\log^2 n \log \ell) + m = \ell$$

so we can apply Theorem 10. We get

$$k = m - 2\log(1/\varepsilon) - O(1) = \ell - O(\log^2 n \log \ell) = \ell - o(\ell)$$

(the latter follows since $\ell = \omega(\log^2 n \cdot \operatorname{loglog} n)$.)

■

Note that, in particular, in the first construction we can choose ℓ to be anything super-logarithmic is n, which is clearly the best we can hope for (if we want to achieve a negligible error). Indeed, otherwise we could do exhausive search in polynomial time, so the statistical distance could not be negligible. Seen another way, we can choose n to be essentially any size larger than ℓ, providing excellent security against partial key exposure. We also remark that we get an *exponential separation* between perfect and statistical ERF's. Indeed, by Theorem 9 in the perfect setting we were limited to have $\ell > n/2$, while here we can achieve $\ell \approx k$, which can be slightly super-logarithmic!

Finally notice that our statistical constructions (especially the last one) have essentially the best possible k, since any statistical ℓ-ERF must have $k \leq \ell$ by Lemma 6.

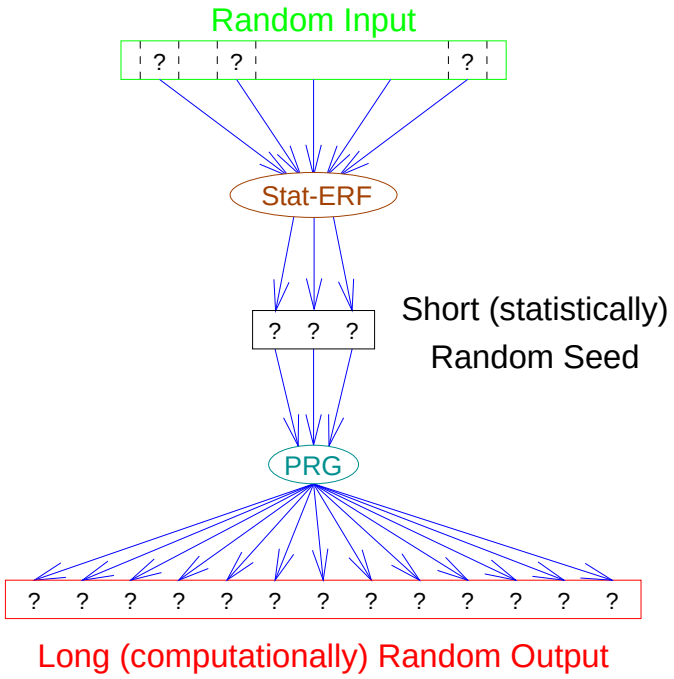

Figure 4-2: Statitical ERF + PRG $\Rightarrow$ Computational ERF.

4.3 Computational ERF

The only limiting factor of our statistical construction is that the output size is (inevitably) limitted to $k \leq \ell$. By finally relaxing our requirement to *computational* security, we are able to achieve an arbitrary output size (in addition to essentially arbitrary exposure-resilience), 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.

We start from the following almost immediate "composition" lemma. In essense, statistical ERF provide us with good exposire-resilience but limited output size, while pseudorandom generators stretch short (statistically) random input into a long computationally random output. By combining the two, we get optimal computational ERF's, which is also illustrated in Figure 4-2.

Lemma 7 *Let $m, n = poly(k)$, $f : \{0,1\}^n \to \{0,1\}^k$ be a statistical ℓ-ERF and $G : \{0,1\}^k \to \{0,1\}^m$ be a PRG. Then $g : \{0,1\}^n \to \{0,1\}^m$ mapping $r \mapsto G(f(r))$ is*

a computational ℓ-ERF.

Proof: Let $L \in \binom{n}{\ell}$. Suppose there was a distinguisher D distinguishing between $A = \langle [r]_{\bar{L}}, G(f(r)) \rangle$ and $B = \langle [r]_{\bar{L}}, R \rangle$ with non-negligible 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 $A = \langle [r]_{\bar{L}}, G(f(r)) \rangle$ and $C = \langle [r]_{\bar{L}}, G(K) \rangle$ are within statistical distance ε of one another, where K is the uniform distribution on $\{0,1\}^k$ and ε is negligible. Thus, D distinguishes C from B with non-negligible advantage $(\delta - \varepsilon)$, 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 $\delta - \varepsilon$), by simply picking a random $r \in \{0,1\}^n$, and providing D with $[r]_{\bar{L}}$ as the first component. This contradicts the security of the pseudorandom generator G, completing the proof. ■

Theorem 12 *Assume one-way functions exist. Then for any* ℓ*, any* $n = poly(\ell)$ *and* $k = poly(n)$*, there exists a computational* ℓ-ERF $g : \{0,1\}^n \to \{0,1\}^k$.

Proof: Since $k = poly(\ell)$, one-way functions imply (by Theorem 2) the existence of a PRG $G : \{0,1\}^{\ell/6} \to \{0,1\}^k$. Theorem 11 implies the existence of a statistical ℓ-ERF f from $\{0,1\}^n$ to $\{0,1\}^{\ell/6}$ with negligible statistical deviation. By Lemma 7, $g(r) = G(f(r))$ is the desired computational ℓ-ERF. ■

The above result clearly provides the strongest possible computational construction we can hope to achieve: we *deterministically* stretch our input by an arbitrary amount, and yet the output is psedorandom even when the adversary misses just a tiniest fraction of input bits of his choice!

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

Lemma 8 *If there exists an* ℓ-ERF $f : \{0,1\}^n \to \{0,1\}^k$*, for* $k > \ell$ *(for infinitely many different values of* ℓ, n, k*), then one-way functions exist.*

Proof: Take any L, let $r \in \{0,1\}^n$ and $R \in \{0,1\}^k$ be chosen uniformly at random, and let $A = \langle [r]_{\bar{L}}, f(r) \rangle$, $B = \langle [r]_{\bar{L}}, R \rangle$. From the proof of Lemma 6, the statistical

distance between A and B is at least $1/2$ (intuitively, A has at most n "bits of randomness", while B has $n - \ell + k \geq n + 1$ "bits of randomness"). By the result of Goldreich [30], 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 7 now follows by combining Theorem 9, Theorem 11, Lemma 6, Theorem 12 and Lemma 8.

4.4 Adaptively Secure ERF

We now address the question of constructing *adaptively* secure ERF's, where the adversary can adaptively decide which $(n - \ell)$ bits of the input to examine. As we have pointed out, in the perfect setting adaptively and non-adaptively secure ERF's are the same thing. In particular, all the limitations of perfect ERF's still hold. On the other hand, it is very easy to see that Lemma 7 holds in the adaptive setting as well. Namely, if we have a statistical adaptive ℓ-ERF f from n bits to k bits, and a pseudorandom generator G from k bits to m bits, then the composition $G(f(\cdot))$ is a computational adaptive ℓ-ERF from n bits to m bits (the proof is identical to that of Lemma 7). Since in the statistical setting we will be able to achieve $\ell \approx k$ (at least existentially), we would get computational adaptive ℓ-ERF's with the same parameters as in the regular non-adaptive setting (e.g., the anolog of Theorem 12 will hold). Thus, the main interesting setup we have to consider is that of *statistical* adaptive ERF's.

4.4.1 Statistical Adaptive ERF

We will present an efficient probabilistic construction of statistical adaptive ERF's with $\ell \approx k$.[3] In other words, we will achieve the same (essentially optimal) bound

[3] As an indirect consequence of our construction, we will construct the so called almost $(n - \ell)$-resilient functions that dramatically beat the parameters achieved for these functions by Kurosawa et al. [39]. However, our construction will be probabilistic.

as we had with ordinary ERF's. In particular, such adaptively secure ERF's exist and are very efficient to evaluate (ones we have found one). This should be contrasted with choosing a truly random function. One can show that a truly random function is indeed a great adaptive ERF (which is already interesting), but it would require an exponential number of bits to store and exponential time to evaluate. In contrast, the representation of the functions we construct will be very short, and they can be evaluated in almost linear time in n. The only drawback, however, is that we cannot give an *explicit* functon that is guaranteed to work. Rather, we give *a family* of functions *most* of which are *guaranteed* to be great adaptive ℓ-ERF, but we cannot prove this about any *specific* finction in this family. In practice, however, one can pick such a function at random once and for all, and be sure almost certainly that it works.

VARIOUS NOTIONS OF ADAPTIVE ERF'S. Having said this, let us turn back to statistical adaptive ERF's. One can think about at least the following four scenarios for an adaptive adversary $\mathcal{A}$, stated in terms of requiring more and more from our function $f : \{0,1\}^n \to \{0,1\}^k$. (We stress again that the adversary $\mathcal{A}$ below is computationally unbounded.)

S1. $r \in \{0,1\}^n$ is chosen at random. $\mathcal{A}$ can adaptively learn one-bit-at-a-time any $(n-\ell)$ bits of r, call them w. $\mathcal{A}$ is then given a challenge Z which is either $f(r)$ or a totally random $R \in \{0,1\}^k$. $\mathcal{A}$ has to distinguish between these two cases with non-negligible advantage.

S2. $r \in \{0,1\}^n$ is chosen at random. $\mathcal{A}$ is then given a challenge Z which is either $f(r)$ or a totally random $R \in \{0,1\}^k$. Based on Z, $\mathcal{A}$ can adaptively learn one-bit-at-a-time any $(n-\ell)$ bits of r, call them w. $\mathcal{A}$ has to distinguish if $Z = f(r)$ or $Z = R$ with non-negligible advantage.

S3. $\mathcal{A}$ chooses any set $L \in \binom{n}{\ell}$ and any $w \in \{0,1\}^{n-\ell}$. $\mathcal{A}$ requests that $[r]_{\bar{L}}$ is set to w. The remaining ℓ bits of r in L are set ar random. $\mathcal{A}$ is then given a challenge Z which is either $f(r)$ or a totally random $R \in \{0,1\}^k$. $\mathcal{A}$ has to

distinguish these two cases with non-negligible advantage. Put another way, $\mathcal{A}$ loses if for any $L \in \binom{n}{\ell}$ and any $w \in \{0,1\}^{n-\ell}$, the distribution induced by $f(r)$ when $[r]_{\bar{L}} = w$ and the other ℓ bits of r chosen at random, is statistically close to the uniform on $\{0,1\}^k$.

S4. $\mathcal{A}$ chooses any set $L \in \binom{n}{\ell}$ and any $w \in \{0,1\}^{n-\ell}$. $\mathcal{A}$ requests that $[r]_{\bar{L}}$ is set to w. The remaining ℓ bits of r in L are set ar random and $Y = f(r)$ is evaluated. $\mathcal{A}$ wins if there exists $y \in \{0,1\}^k$ such that $\Pr(Y = y)$ in this experiment does not lie within $2^{-k}(1 \pm \epsilon)$, where ϵ is negligible. Put another way, $\mathcal{A}$ loses if for any $L \in \binom{n}{\ell}$, any $w \in \{0,1\}^{n-\ell}$ and any $y \in \{0,1\}^k$, the probability that $f(r) = y$ when $[r]_{\bar{L}} = w$ and the other ℓ bits of r chosen at random, is within $2^{-k}(1 \pm \epsilon)$ (for negligible ϵ).

Our "official" Definition 12 of adaptive ERF is that satisfying scenario S2. Let us briefly summarize the above four variants. Variant S1 allows the adversary to adaptively choose which $(n - \ell)$ bits to learn before he sees anything else. In applications where adaptive security is important, however, $\mathcal{A}$ will typically have some partial information about $f(r)$ through the use of the system where $f(r)$ is the secret key. Therefore, our definition settled for version S2, where the adversary chooses which bits to see after he obseves the challenge (either $f(r)$ or R). Versions S3 and S4 have a different flavor. Here not the *entire* r is chosen at random. Rather the adversary fixes any $(n - \ell)$ bits of r to some string w. The remaining bits are set at random, and we still want $f(r)$ to be "really random": in variant S3 to be statistically close to uniform and in variant S4 to hit *every* single $y \in \{0,1\}^k$ with probability roughly 2^{-k}. We can view the setups S3 and S4 as requiring that on any *ℓ-dimensional subcube* (i.e., the set of r satisfying $[r]_{\bar{L}} = w$ for a fixed L and w) of $\{0,1\}^n$ the distribution induced by $f(r)$ is close to uniform in the $\mathcal{L}^1$ and $\mathcal{L}^\infty$ norms respectively.

While it is clear that version S2 is stronger than S1 and version S4 is stronger than S3, let us show the only non-trivial implication that version S3 is stronger than S2. Assume f does not satisfy S2. This means that if $i \in \{0,1\}$, r, R are chosen at random, Z is set to $f(r)$ if $i = 0$ and to R if $i = 1$, and $\mathcal{A}$ is given Z, $\mathcal{A}$ can predict

i well. In particular, this holds *on average* over all the random choices above. By conditioning over the value $w \in \{0,1\}^{n-\ell}$ of the ℓ positions L that $\mathcal{A}$ observes, there exist some *particular* $L \in \binom{n}{\ell}$ and $w \in \{0,1\}^{n-\ell}$, such that *conditioned* on $[r]_{\bar{L}} = w$ and the fact that $\mathcal{A}$ would choose to examine $[r]_{\bar{L}}$, $\mathcal{A}$ would distinguish $f(r)$ from R. Therefore, in the experiment where $[r]_{\bar{L}}$ is set to w, the remainder of r is set at random, R is set at random, and $\mathcal{A}$ behaves at random as before, *provided that* $\mathcal{A}$ *chose to examine* $[r]_{\bar{L}}$ *and saw* $[r]_{\bar{L}} = w$, we would have that $\mathcal{A}$ distinguishes $f(r)$ from R (i.e., guesses i well). But then this L, w and "$\mathcal{A}$ conditioned on L and w" contradict definition S3. In other words, when $[r]_{\bar{L}} = w$, the remaining bits of r are set at random and Z is set to $f(r)$ or R, our new adversary $\mathcal{A}'$ will run $\mathcal{A}$ as many times as he needs to until $\mathcal{A}$ finally choses to examine positions in L (and thus, necessarily observes $[r]_{\bar{L}} = w$). When this happens, he outputs whatever $\mathcal{A}$ does for its guess, completing the proof. To summarize this in a different way, the definition S3 requires f to be good *for every possible* L and $[r]_{\bar{L}}$, and therefore subsumes anything that $\mathcal{A}$ could possibly see in the scenario S2.

Comparing with non-adaptive ERF. We show that scenario S4 is still much weaker than the notion of a perfect ERF (which requires $\epsilon = 0$, i.e. to induce perfectly uniform distribution on every subcube given by L and w), while scenario S1 is still much stronger than our non-adaptive notion of statistical ERF (which works for any *fixed* L). Thus, we really have a hierarchy of refinements between perfect and statistical ERF's. The first claim (about perfect ERF and those satisfying scenario S4) is shown later in this section by being able to achieve $\ell \approx k$ even in the scenario S4 (which is impossible with perfect ERF's by Theorem 9). This also shows an *exponential gap* between (adaptive) perfect and adaptive statistical ERF's.

The second separation between non-adaptive statistical ERF and those satisfying scnario S1 we argue now. We do it by noticing that, unfortunately, our construction of statistical ERF's in Section 4.2 (using strong extractors) does not satisfy even the weakest adaptive notion S1. Indeed, the definition of a strong extractor requires that the hash function is chosen *completely independently* of the random source. In our

construction the randomness used to generate the hash function is *part of the input*. For any fixed L, since the input r is chosen at random, the hash function h is indeed independent from our source X (the remaining $(n-d)$ input bits, where d is the length of the seed to the extractor). When the adversary is adaptive, however, he may choose to first learn the first d bits that define the hash function h. Only then will he choose which other bits of r to read. Therefore, the source X (of min-entropy $m \approx \ell - d$) that the adversary creates is *dependent* on the choice of the hash function h, so we cannot say that h will extract almost m random bits. And, indeed, in all the strong extractor families that we used, the output bits produced need not be random if the adversary can choose the random variable X *after* the choice of the hash function is made. Therefore, a new idea is needed to deal with adaptive ERF's. This idea will be to use δ-sure (m, ϵ)-extractors defined in Section 2.8 for a very small δ, so that with high probability out hash function works for *all* possible choices of the adversary.

4.4.2 Construction using t-wise Independent Functions

As we said, we will satisfy the strongest adaptive definition S4 above. So not only we will achieve adaptive security as stated in the scenario S2 (and our Definition 12 of adaptive ERF), but our f will actually induce an almost "perfect uniform distribution": for each subcube given by $L \in \binom{n}{\ell}$ and $w \in \{0,1\}^{n-\ell}$, the number of preimages of every single $y \in \{0,1\}^k$ will be within $2^{\ell-k}(1 \pm \epsilon)$ for a negligible ϵ. More importantly, we will acheive the output size $k = \ell - o(\ell)$.

We notice that the variant S4 was exactly the notion of ERF considered by Kurosawa et al. [39], who called such functions ϵ-*almost* $(n-\ell)$-*resilient* functions. They give a complicated explicit construction which is slightly better than what is possible with perfect ERF's, but still required the adversary to miss at least half of the input: $\ell > n/2$. While it might seem from that result that maybe one really cannot hope to achieve very good parameters under such a strong definition, we show that this is not the case; that we can achieve $\ell \approx k$. However, our construction is probabilistic (but efficient).

Fix any $L \in \binom{n}{\ell}$ and $w \in \{0,1\}^{n-\ell}$. Pick a random r such that $[r]_{\bar{L}} = w$. For notational convenience, denote this r by $X = X(L,w)$. Even though when we know L and w, it is trivial to extract ℓ bits out of X, we pretend that we do not know them and only know that X has min-entropy ℓ. Then, if we apply a good enough extractor to X, we should be able to extract almost ℓ random bits out of it. As before, the problem with this is that we need some extra randomness to pick a hash function from the extractor family. However, there are "only" $M = \binom{n}{\ell}2^{n-\ell}$ choices of L and w. So if a random function in our extractor family can be good (with high probability) for *all* these M choices, we will be done. But this is exactly the notion of δ-sure (ℓ,ϵ)-extractors considered in Section 2.8!

In particular, we showed in Section 2.8 that for any family of t-wise independent functions (for a high enough t), a random function from this family will succeed to extract almost all the randomness from a lot of sources of min-entropy ℓ. More specifically, we can apply Corollary 3 to our situation. The number of random sources is $M = \binom{n}{\ell}2^{n-\ell} < 2^{2n}$, all of them have min-entropy ℓ, so we can achieve $t = \ell + \log M = O(n)$ and $k = \ell - (2\log\frac{1}{\epsilon} + \operatorname{loglog} M + \log\ell + O(1)) = \ell - 2\log\frac{1}{\epsilon} - O(\log n)$. Notice, however, that a-priori this only satisfies the notion S3, since it just achieves statistical deviation from uniform equal ϵ. But a closer look at the proof of Corollary 3 (from Lemma 3) shows that we simply take a union bound over all $y \in \{0,1\}^k$, so we in fact achieve the strongest notion S4.

In fact, it will be more convenient to go directly through Lemma 3 to achieve slightly stronger parameters, even though the above parameters are already good. Let us set $t = n/\log n$ to be our independence index, $m = \ell$ to be our min-entropy, $\alpha = 3\log n$, and

$$k = \ell - 2\log\frac{1}{\epsilon} - \log t - 2\alpha \geq \ell - 2\log\frac{1}{\epsilon} - 7\log n$$

Then we get that for any $w \in \{0,1\}^{n-\ell}$ and and $L \in \binom{n}{\ell}$, if we set $[r]_{\bar{L}} = w$ and

choose the remaining bits of r at random,

$$\Pr_{f\in\mathcal{F}}\left(\left|\Pr_r(f(r)=y)-\frac{1}{2^k}\right|\geq\epsilon\cdot\frac{1}{2^k}\right)\leq 2^{-\alpha t}=2^{-3n}$$

Now we take a union bound over 2^k possible y, and $M=\binom{n}{\ell}2^{n-\ell}$ possible settings of some $(n-\ell)$ bits of r, and get the probability of error at most $\binom{n}{\ell}2^{n-\ell}2^k2^{-3n}\leq 2^{-n}$. Hence, we have shown

Theorem 13 *Fix any n, ℓ and ϵ. Let $\mathcal{F}$ be a family of t-wise independent functions from n bits to k bits, where $t=n/\log n$ and*

$$k=\ell-2\log\left(\frac{1}{\epsilon}\right)-O(\log n)$$

Then with probability at least $(1-2^{-n})$ a random function f sampled from $\mathcal{F}$ will be a statistical adaptive ℓ-ERF with error ϵ satisfying the strongest adaptive notion S4. *Namely, for every $L\in\{\binom{n}{\ell}\}$, $w\in\{0,1\}^{n-\ell}$ and $y\in\{0,1\}^k$, the number of r satisfying $[r]_L=w$ and $f(r)=y$ is within the interval $2^{\ell-k}(1\pm\epsilon)$.*

Corollary 5 *For any $\ell=\omega(\log n)$, there exists efficient statistical adaptive ℓ-ERF $f:\{0,1\}^n\to\{0,1\}^k$ with $k=\ell-o(\ell)$.*

Notice that the parameters we achieve are even marginally better than what we had in the statistical construction in Theorem 11. The catch is, of course, that the latter is an *explicit* construction, while the former is a *probabilistic* (albeit very efficient) construction. This raises the following interesting question.

Question 1 *Can we* explicitly *construct an adaptively secure statistical* ERF *achieving $\ell\approx k$ (or even having just $\ell<n/2$)?*

Notice that from an existential point of view, the result of Corollary 5 says that such adaptive ERF's exist (and can be easily sampled).

Remark 1 We already noticed that by applying a pseudorandom generator to the output of an adaptively secure statistical ℓ-ERF, we get a *computationally* secure

adaptive ℓ-ERF (with a much larger output size), obtaining a probabilistic analog of Theorem 12. In fact, since our statistical adaptive ERF's satisfy the strong notion S4 (actually, even notion S3 suffices for this comment), we can say the following stronger statement about the resulting computational ERF: for any L and $w \in \{0,1\}^{n-\ell}$, if the adversary *fixes* $[r]_{\bar{L}} = w$, then by setting the remaining ℓ bits of r at random, the resulting k-bit output will be computationally indistinguishable from uniform. Following the terminology of [20, 39], we can call such functions *computationally* $(n-\ell)$*-resilient*.

Chapter 5

All-Or-Nothing Transforms (AONT)

As we pointed out, no AONT constructions with analysis outside the random oracle model were known. We give several such constructions. We start by considering perfect AONT's. We show that they are much more general than perfect ERF's. Yet, we non-trivially extend the the lower bound of Theorem 9 for perfect ERF's and show a more general impossibility result for perfect AONT's. Namely, the adversary must miss at least half of the (secret) output of the AONT. We show that the question of constructing perfect AONT's is equivalent to a question of finding "balanced" weighted colorings of the n-dimensional hypercube, and then show that no "very balanced" such colorings exist by taking a surprising recourse into quadratic forms and Fourier analysis. Thus, similar to ERF's, perfect AONT's have strong combinatorial limitations.

We then give a very simple "universal" construction of AONT's using ERF's (which works in any setting and even in the *adaptive* case!). This yields the *first construction of* AONT*'s outside the random oracle model*, and, moreover, the statistical and computational AONT's that we construct have essentially the best possible parameters, dramatically beating the impossibility result for perfect AONT's. In particular, the statistical construction achieves an ℓ-AONT with $\ell \approx k$ (and even a *secret-only* ℓ-AONT with $\ell = O(k)$), showing an exponential separation between the perfect and the statistical settings. The computational construction (from any one-way function) also implies that for the interesting settings of parameters (essentially, $\ell < k$), the

existence of ℓ-AONT's, ℓ-ERF's and one-way functions are *all equivalent*. The other construction we give can be viewed as the special case of the OAEP construction of Bellare and Rogaway [8] which was shown to yield an AONT in the Random Oracle model [16]. Thus, this construction can be viewed as the first step towards abstracting the properties of the random oracle that suffice for this construction to be an AONT. 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 Perfect AONT

We first consider perfect ℓ-AONT's and show that they have strong combinatorial limitations. Since our main result will be an impossibility result, in this section we will ignore the restrictions which involve efficiency of the computations, even though all our constructions will be efficient. Thus, our lower bounds holds in a purely combinatorial setting and under no computational restrictions, which makes them only stronger.

First, we observe that it suffices to restrict our attention to *secret-only* ℓ-AONT's, since the public part can be ignored. Indeed, the definition of a perfect ℓ-AONT $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^p$ implies that all the distributions $\langle x, [y_1]_{\bar{L}}, y_2 \rangle$ are identically distributed for any $x \in \{0,1\}^k$ and $L \in \binom{s}{\ell}$. In particular, for any fixed $\tilde{y}_2 \in \{0,1\}^p$ (that is possible as a valid public part) we get that all the conditional distibutions $\langle x, [y_1]_{\bar{L}} \mid y_2 = \tilde{y}_2 \rangle$ are the same. Let us fix any such possible $\tilde{y}_2$. Then, $T' : \{0,1\}^k \to \{0,1\}^s$ that simply outputs a random y_1 such that $(y_1, \tilde{y}_2) \in T(x)$ is a secret-only ℓ-AONT (it might not be efficiently computable, but we said that we do not worry about the efficiency this section). Therefore, we restrict our attention to secret-only ℓ-AONT's $T : \{0,1\}^k \to \{0,1\}^n$ (we switch from s to n for convenience), and will typically omit the phrase "secret-only" in this section.

5.1.1 Perfect (secret-only) AONT vs. perfect ERF

We start by showing that perfect AONT's seem to be sufficiently more general than perfect ERF's. Namely, a perfect ℓ-ERF immediately implies a perfect ℓ-AONT, but the converse does nor appear to hold. We will also derive an alternative definition of a perfect AONT that will closely resemble the definition of a perfect ERF and simplify the subsequent discussion.

Lemma 9 *Assume $f : \{0,1\}^n \to \{0,1\}^k$ is ℓ-ERF. Then there exists a (secret-only) ℓ-AONT $T : \{0,1\}^k \to \{0,1\}^n$ (which might not be efficiently computable).*

Proof: Given $x \in \{0,1\}^k$, define $T(x)$ to be a random $r \in \{0,1\}^n$ such that $f(r) = x$. In other words, $T(x)$ is a random inverse $r \in f^{-1}(x)$. First of all, such an r always exists since f must be surjective (otherwise, $f(r)$ cannot induce a uniform distribution on $\{0,1\}^k$ for a random r). Take any $L \in \binom{n}{\ell}$. The fact that f is a perfect ERF means that when r is chosen at random, the conditional distribution of $f(r)$ given $[r]_{\bar{L}}$ is uniform. In particular, for any $x \in \{0,1\}^k$ and any $w \in \{0,1\}^{n-\ell}$ we get that the number of r such that $[r]_{\bar{L}} = w$ and $f(r) = x$ is the same (namely, $2^{\ell-k}$). Therefore, the value r for $T(x)$ can be chosen by first choosing a random $w \in \{0,1\}^{n-\ell}$ (which is done *independently of* x) and then choosing a random r such that $[r]_{\bar{L}} = w$ and $f(r) = x$. But since the adversary only observes w which was chosen independently of x, he indeed gets no information about x from $w = [r]_{\bar{L}}$, completing the proof. ∎

Constructions of AONT from ERF. In particular, we can apply Theorem 8 to obtain perfect (secret-only) AONT's. Namely, let M be the $k \times n$ generator matrix of a linear error-correcting code of distance d. Then the following randomized transformation $T : \{0,1\}^k \to \{0,1\}^n$ is an ℓ-AONT, for $\ell = n - d + 1$. Given x, we find a random solution $r \in \{0,1\}^n$ to the linear system $Mr = x$ and let $T(x) = r$. We note that such T is *efficiently computable*. In fact, the generic solution to $Mr = x$ is always given by $(n-k)$ "free bits" of r that can be set arbitrarily, while the other k bits are fixed linear combinations of the "free bits". Thus, T can be represented

as a fixed linear transformation given by some $n \times (n-k)$ matrix P. We simply choose $(n-k)$ random bits t (corresponding to the "free bits") and output $r = Pt$. In particular, we can achive both k and $(n-\ell)$ to be (small) constant factors of n by taking any assymptotically good code M. We can also push ℓ to be arbitrarily close to $n/2$ by making k significantly smaller than n (by a large constant factor), but can never cross $n/2$ this way (unless $k \leq \log n$). Finally, the upper bound of Lemma 5 holds as well.

ERF'S FROM "UNIFORM" AONT'S. We remark that perfect AONT's seem to be much more general than perfect ERF's. Given a perfect AONT, one might try to define an ERF using the inverse map I of. Unfortunately, this seems to work only in very special cases and to fail dramatically with general AONT's. For example, not every $r \in \{0,1\}^n$ has to be a possible image of some $x \in \{0,1\}^k$ under T, so I may not be even defined for many $r \in \{0,1\}^n$ (for the simplest possible example, take any AONT and add a "dummy" 0 at the end). More generally, the probabilities that $T(x) = r$ can be potentially very complicated functions of x and r (even irrational!), so even if I was "defined" everywhere, the map $I(r)$ has "no reason" to be an ERF.

To see this from a different angle, AONT's constructed from ERF's via Lemma 9 satisfy a very "regular" condition that for every $x \in \{0,1\}^k$ and every $w \in \{0,1\}^{n-\ell}$ there are exactly $2^{\ell-k}$ possible images r such that $[r]_{\bar{L}} = w$ and $x \in T^{-1}(r)$, each of which is selected with a uniform probability by T. We call such AONT's *uniform*. More specifically, a uniform AONT $T : \{0,1\}^k \to \{0,1\}^n$ has the following form: we partition $\{0,1\}^n$ into 2^k disjoint subsets $S_1, \ldots, S_{2^k}$, and every $x \in \{0,1\}^k$ is mapped to a *uniformly random* element of S_x. Not surprisingly, for these very strict and special cases of AONT's the converse of Lemma 9 holds.

Lemma 10 *Assume $T : \{0,1\}^k \to \{0,1\}^n$ is a uniform ℓ-AONT. Then there exists (efficiently computable) ℓ-ERF $f : \{0,1\}^n \to \{0,1\}^k$.*

Proof: Define $f(r) = I(r)$, where I is the (efficiently computable) inversion map for T. Let us fix any $L \in \binom{n}{\ell}$, pick a random $x \in \{0,1\}^k$ and compute $r \leftarrow T(x)$. Uniformity of T implies that we induce a uniform distribution on r. The fact that T

is an ℓ-AONT implies that observing $[r]_{\bar{L}}$ gives no information on x. But this means that we can first pick a random r, let the adversary observe $[r]_{\bar{L}}$, then compute $x = I(r) = f(r)$, and this will induce a uniform distribution on x. This is exactly the definition of an ℓ-ERF. ■

Corollary 6 *Perfect uniform ℓ-AONT $T : \{0,1\}^k \to \{0,1\}^n$ exist iff perfect ℓ-ERF $f : \{0,1\}^n \to \{0,1\}^k$ exist.*

ANOTHER VIEW OF PERFECT AONT'S. We now restate the above comparison between (perfect) AONT's and ERF's in a slightly different way. Recall that the definition of a perfect AONT T says that for any $x_0, x_1 \in \{0,1\}^k$ we have $[T(x_0)]_{\bar{L}} \equiv [T(x_1)]_{\bar{L}}$. We claim that the following is an equivalent definition of a perfect[1] AONT (again, we ignore the efficiency considerations here).

Definition 15 *A randomized function $T : \{0,1\}^k \to \{0,1\}^n$ is a perfect ℓ-AONT if T is invertible (i.e., there is an inverse transformation I such that for any $x \in \{0,1\}^k$ and any $r \in T(x)$, we have $I(r) = x$) and for any $L \in \binom{n}{\ell}$ and for a randomly chosen $x \in \{0,1\}^k$, $R \in \{0,1\}^k$, the following distributions are identical:*

$$\langle [T(x)]_{\bar{L}},\ x\rangle \equiv \langle [T(x)]_{\bar{L}},\ R\rangle$$

The definition above says that observing $[T(x)]_{\bar{L}}$ does not give any information about a randomly chosen x. Put another way, the *conditional* distribution of x given $[T(x)]_{\bar{L}}$ is still uniform.

Lemma 11 *The above definition of perfect AONT is equivalent to the original Definition 13 of (secret-only) perfect AONT.*

Proof: Assume T satisfies the original definition, i.e. the distributions $[T(x)]_{\bar{L}}$ are all the same (irrespective of x). Call this distribution p. But then, when x is chosen

[1]Notice, this equivalence does *not* hold for the statistical and the computational settings.

at random, the distribution of $[T(x)]_{\bar{L}}$ is still p, so it gives no information about x. Thus, the conditional distribution on x is the same as we started from, i.e. uniform.

Conversely, assume T satisfies the new definition. Take any $a \in \{0,1\}^k$ and any $w \in \{0,1\}^{n-\ell}$ that happens with non-zero probability as the value of $[T(x)]_{\bar{L}}$ when x is chosen at random. Since the conditional distribution of x after any conceivable observation w of $[T(x)]_{\bar{L}}$ is uniform, we know that $\Pr_{x,T}(x = a \mid [T(x)]_{\bar{L}} = w) = 2^{-k}$. By Bayes' law, we can rewrite this probability as

$$2^{-k} = \frac{\Pr_x(x = a) \cdot \Pr_{x,T}([T(x)]_{\bar{L}} = w \mid x = a)}{\Pr_{x,T}([T(x)]_{\bar{L}} = w)} = 2^{-k} \cdot \frac{\Pr_T([T(a)]_{\bar{L}} = w)}{\Pr_{x,T}([T(x)]_{\bar{L}} = w)}$$

Hence, we get $\Pr_T([T(a)]_{\bar{L}} = w) = \Pr_{x,T}([T(x)]_{\bar{L}} = w)$, which is *independent* of a. This is exactly the original defintion of a perfect AONT. ∎

Now given an AONT T, define the probability distribution D on $\{0,1\}^n$ by $D(r) = \Pr_{x,T}(T(x) = r)$, i.e. the probability that $T(x) = r$ when x is chosen at random from $\{0,1\}^k$.

Claim 1 *The distribution D and the inverse transformation I* uniquely define *T.*

Proof: Indeed, let $S_x = \{r \mid I(r) = x\}$[2] be the set of images of x under T, and let D_x be the conditional distribution on $r \in S_x$ induced by D (i.e., $D_x(r) = D(r)/\sum_{r' \in S_x} D(r')$ for $r \in S_x$). Then the invertibility of T and the definition of D immediately imply that $T(x)$ simply samples $r \in S_x$ *according to the distribution D_x*. Thus, we can replace T by a pair (D, I). ∎

Also notice that the invertibility of T implies that if we sample $r \in \{0,1\}^n$ according to D and let $x = I(r)$, we get a *uniform* distribution on $x \in \{0,1\}^k$. Namely, instead of sampling a random $x \in \{0,1\}^k$ and computing $r \leftarrow T(x)$, we can sample r according to D and compute $x = I(r)$. Applying this to Definition 15, we get yet another equivalent definition of a perfect AONT, which now really resembles that of a perfect ERF.

[2]Because $D(r) = 0$ for all r that are not images of *any* $x \in \{0,1\}^k$, it does not really matter what is the value of I on such "impossible" r. Say, we fix it to the all-zero string.

Definition 16 *A transformation T, uniquely given by a distribution D on $\{0,1\}^n$ and a deterministic function $I : \{0,1\}^n \to \{0,1\}^k$ (as in Claim 1), is an ℓ-*AONT*, if when r is sampled according to D and R is chisen uniformly from $\{0,1\}^k$, we have for any $L \in \binom{n}{\ell}$:*

$$\langle [r]_{\bar{L}}, I(r) \rangle \equiv \langle [r]_{\bar{L}}, R \rangle \tag{5.1}$$

Notice that Equation (5.1) is exactly the same as Equation (3.1) in the definition of a perfect ERF $f : \{0,1\}^n \to \{0,1\}^k$, except the uniform distribution on r is replaced by the distribution D. Thus, we see a crystal-clear relation between perfect AONT's and ERF's. For a perfect ERF we choose r *uniformly at random* from $\{0,1\}^n$, while for a perfect AONT we allow to have an *arbitrary* distribution D on r (as long as Equation (5.1) is satisfied). In other words, in designing perfect AONT's we have an extra degree of freedeom in picking the distribution D in addition to the function $I : \{0,1\}^n \to \{0,1\}^k$, while an ERF $f : \{0,1\}^n \to \{0,1\}^k$ restricts D to be uniform. In this latter case of D being the uniform distribution, the resulting AONT is exactly what we called a *uniform* AONT earlier (where we partition $\{0,1\}^n$ into sets S_x and let $T(x)$ be a uniformly random element of S_x). This again shows that ERF's are equivalent to uniform AONT's (Corollary 6).

5.1.2 Impossibility Result

We have seen that ERF's have strong limitations, given by Theorem 9, i.e. $\ell \geq n/2 + 1 - n/(2(2^k - 1))$. Moreover, this bound is tight by Lemma 5. We have also seen that AONT's immediately imply ERF's with the same parameters, while the converse holds only for very special kinds of *uniform* AONT's, and does not appear to hold in general. The natural question comes up if perfect AONT's nevertheless share the same combinatirial limitation as perfect ERF's (and uniform AONT's).

We notice that the proofs of Friedman [28] and Bierbrauer et al. [11] of Theorem 9 fail once we go to general AONT's, since they strongly use the "uniformity" of ERF's accross different ℓ-dimensional subcubes. Still, we are able to show that the exact analogs of Theorem 9 and the bound in Equation (4.1) *do hold for general perfect*

ℓ-AONT. In fact, we show that the only ℓ-AONT's that can potentially achieve these bounds *are in fact uniform*. In other words, non-uniform perfect AONT's do not seem to bring a significant advantage, despite their generality. While a very broad idea of our proof is the same as that of [28], our proof is significantly more involved and requires more care. And, of course, it subsumes the result of Theorem 9 due to Lemma 9 (i.e., that perfect ERF's imply perfect AONT's). In addition, Lemmas 5 and 9 show that this result is tight, and is in fact achived by a uniform AONT.

Theorem 14 *If $T : \{0,1\}^k \to \{0,1\}^n$ is a perfect ℓ-AONT, then*

$$\ell \geq 1 + n \cdot \frac{2^{k-1} - 1}{2^k - 1} = \frac{n}{2} + \left(1 - \frac{n}{2(2^k - 1)}\right) \tag{5.2}$$

In particular, for $n \leq 2^k$ we get $\ell > \frac{n}{2}$, so at least half of the output of T has to remain secret even if T exponentially expands its input! Moreover, the above bound can be achieved only by a uniform ℓ-AONT.

As we will see, the proof will follow from the impossibility of certain weighted "balanced" colorings of an n-dimensional hypercube.

5.1.3 Balanced Colorings of the Hypercube

In this section, we point out a natural relation between perfect AONT's and certain weighted "balanced" colorings of the hypercube $\mathcal{H} = \{0,1\}^n$. Recall that in the graph of the hypercube two strings $y, z \in \{0,1\}^n$ are adjacent if and only if they differ in a single position.

For our purposes, a *coloring* $\mathcal{C}$ of a graph with c colors is any map which associates a color from $\{1, \ldots, c\}$ to each node in the graph.[3] In a *weighted* coloring, the nodes that are colored are also assigned a non-negative real weight. Sometimes, for obvious reasons, we will call the nodes of weight 0 *uncolored*, despite them having assigned a nominal color. We will denote the weight of node y by $\chi(y)$. We will also define

[3] Often one considers colorings such that no pair of adjacent nodes has the same color. We do *not* impose such restrictions on the colorings we study.

the weight vector χ_i of each color i by assigning $\chi_i(y) = \chi(y)$ if y has color i, and 0 otherwise. We notice that for any given $y \in \mathcal{H}$, $\chi_i(y) > 0$ for at most one color i, and also $\sum \chi_i = \chi$. A coloring where all the nodes are uncolored is called *empty*. Since we will never talk about such colorings and the absolute magnitude of the weights will not be important, we agree on the normalization condition that the sum of all the weights is 1, i.e. $\sum_{y \in \mathcal{H}} \chi(y) = 1$. A *uniform* coloring is the one where all the nodes are assigned the same weight (i.e., $\chi(y) = 2^{-n}$ for all y).

We will be interested in the properties of colorings on *subcubes* of the hypercube. Recall that a subcube is the subgraph obtained by fixing some of the n positions and letting the others take on all possible values. More formally, given a set of ℓ positions $L \in \{\binom{n}{\ell}\}$ and some assignment $a \in \{0,1\}^{n-\ell}$ to the remaining variables *not in* L, the subcube $\mathcal{H}_{L,a}$ is the set of nodes r such that r agrees with a on L, i.e. $[r]_{\bar{L}} = a$. Clearly, $|\mathcal{H}_{L,a}| = 2^{\ell}$. The *dimension* of a subcube is $\ell = |L|$ — the number of variables left unfixed.

Definition 17 *We say a weighted coloring of the hypercube is* ℓ-balanced *if, within every subcube of dimension* ℓ*, each color has the same weight. That is, for each* L *and* a*,* $\sum_{y \in \mathcal{H}_{L,a}} \chi_i(y)$ *is the same for all colors* i*.*

We notice that the empty coloring trivially satisfies this condition, and that is the reason why we exclude it from our consideration. On the other hand, a balanced coloring is allowed to contain many ℓ-dimensional subcubes which are completely uncolored (as long as not all of them are uncolored), since each color has the same (zero) weight in such subcubes.

Remark 2 If a coloring is ℓ-balanced, it is also ℓ'-balanced for any $\ell' > \ell$, since an ℓ' dimensional subcube is the disjoint union of ℓ-dimensional ones.

We study balanced colorings since they capture the combinatorial properties of ℓ-AONT's and ℓ-ERF's. We get the following equivalences.

Lemma 12 *Ignoring computational efficiency, we have that the existence of the following are equivalent in the perfect setting:*

1. ℓ-AONT*'s from k bits to n bits.*

2. *ℓ-balanced weighted colorings of n-dimensional hypercube using 2^k colors.*

The following are similarly equivalent in the perfect setting:

1. uniform ℓ-AONT*'s from k bits to n bits.*

2. ℓ-ERF*'s from n to k bits.*

3. uniform *ℓ-balanced colorings of n-dimensional hypercube using 2^k colors.*

Proof: We ignore ℓ-ERF's in the proof since we know that they are (combinatorially) equivalent to uniform ℓ-AONT's by Corollary 6. We will also use a more convenient Definition 16 of an ℓ-AONT given in terms of the inverse map $I : \{0,1\}^n \to \{0,1\}^k$ and the distribution D on $\{0,1\}^n$ induced by T and the uniform distribution on $\{0,1\}^k$.

Now the equivalence between AONT's and weighted colorings is almost immediate. The function I corresponds to assigning a color $I(y)$ to a node $y \in \mathcal{H}$, while the distribution D corresponds to assigning a weight $D(y)$ to a node $y \in \mathcal{H}$. Clearly, the resulting coloring is uniform if and only if the AONT is uniform (i.e., the distribution D is uniform).

It remains to check the balancedness property. But this again immediately follows from Definition 16 of a perfect AONT. In one direction, the definition of an AONT says that for any "non-empty" (i.e., $\Pr_{y \leftarrow D}([y]_{\bar{L}} = a) > 0$) subcube $\mathcal{H}_{L,a}$ of $\mathcal{H}$, we have that the distribution induced by $I(y)$ conditional on $y \in \mathcal{H}_{L,a}$ is uniform (when y is chosen according to D). But the conditional probability of $I(y) = i$ is proportional to the total weight of nodes of color i in this subcube. So having a uniform distribution on $I(y)$ is equivalent to saying that each color has the same weight in $\mathcal{H}_{L,a}$. The converse direction is the same. ∎

We now restate our lower bound on perfect AONT's in Theorem 14 in terms of weighted ℓ-balanced colorings of the hypercube with $c = 2^k$ colors (we prove the theorem for general c).

Theorem 15 *For any (non-empty) ℓ-balanced weighted coloring of the n-dimensional hypercube using c colors,*

$$\ell \geq \frac{n}{2} + \left(1 - \frac{n}{2(c-1)}\right)$$

Moreover, equality can hold only if the coloring is uniform and no two adjacent nodes of positive weight have the same color.

We believe that the above theorem is interesting in its own right. It says that once the number of colors is at least 3, it is impossible to find a c-coloring (even weighted!) of the hypercube such that all ℓ-dimensional subcubes are "equi-colored", unless ℓ is very large.

5.1.4 Proof of the Lower Bound (Theorem 15)

We will work in the 2^n-dimensional vector space V consisting of vectors with positions indexed by the strings in $\mathcal{H}$, and will crucially use the algebraic facts about quadratic forms and Fourier analysis described in Section 2.9. In some sense, it might appear surprising to use real analysis to prove a combinatorial fact, but it turns out that the balancedness property of our coloring is best utilized when we consider an appropriate algebraic expression and bound it in two different ways.

Consider a non-empty ℓ-balanced weighted coloring χ of the hypercube using c colors. Let χ_i be the characteristic weight vector corresponding to color i (i.e. $\chi_i(y)$ is the weight of y when y has color i and 0 otherwise). As we will show, χ_i's have some nice properties which capture the balancedness of the coloring χ. In particular, we know that for any colors i and j and for any ℓ-dimensional subcube of $\mathcal{H}$, the sum of the components of χ_i and of χ_j are the same in this subcube. Hence, if we consider the difference $(\chi_i - \chi_j)$, we get that the sum of its coordinates over any ℓ-dimensional subcube is 0.

Now it turns out that a natural way to exploit the above property is to consider the quantity $(\chi_i - \chi_j)^\top A(\chi_i - \chi_j)$, where A is the adjacency matrix of the n-dimensional hypercube (see Section 2.9). As suggested in Section 2.9, we can bound this quantity by calculating the Fourier coefficients of $(\chi_i - \chi_j)$ corresponding to large eigenvalues.

We get the following lemma:

Lemma 13 *For any $i \neq j$, we have*

$$(\chi_i - \chi_j)^\top A(\chi_i - \chi_j) \leq (2\ell - n - 2) \cdot \|\chi_i - \chi_j\|^2 \tag{5.3}$$

We postpone the proof of this crucial lemma until the the end of the proof, and now just use it to prove our theorem. First, note that the lemma above only gives us information on two colors. To simultaneously use the information from all pairs, we consider the *sum* over all pairs i, j, that is

$$\Delta \stackrel{\text{def}}{=} \sum_{i,j} (\chi_i - \chi_j)^\top A(\chi_i - \chi_j) \tag{5.4}$$

We will give upper and lower bounds for this quantity (Claims 2 and 3, respectively), and use these bounds to prove our theorem. We first give the upper bound, based on Lemma 13.

Claim 2

$$\Delta \leq 2\ (2\ell - n - 2)\ (c-1) \cdot \sum_i \|\chi_i\|^2 \tag{5.5}$$

Proof: We can ignore the terms of Δ when $i = j$ since then $(\chi_i - \chi_j)$ is the 0 vector. Using Lemma 13 we get an upper bound:

$$\sum_{i,j} (\chi_i - \chi_j)^\top A(\chi_i - \chi_j) \leq (2\ell - n - 2) \cdot \sum_{i \neq j} \|\chi_i - \chi_j\|^2$$

Now the vectors χ_i have disjoint supports (since each $y \in \mathcal{H}$ is assigned only one color), so we have $\|\chi_i - \chi_j\|^2 = \|\chi_i\|^2 + \|\chi_j\|^2$. Substituting into the equation above, we see that each $\|\chi_i\|^2$ appears $2(c-1)$ times (recall that c is the number of colors). Hence we get the desired result:

$$\sum_{i,j} (\chi_i - \chi_j)^\top A(\chi_i - \chi_j) \leq (2\ell - n - 2) \cdot 2(c-1) \cdot \sum_i \|\chi_i\|^2$$

■

Second, we can expand this sum directly to obtain a lower bound.

Claim 3

$$\Delta \geq -2n \cdot \sum_i \|\chi_i\|^2 \tag{5.6}$$

Proof: Since A is symmetric we have $\chi_i^\top A \chi_j = \chi_j^\top A \chi_i$. Then:

$$\begin{aligned}\sum_{i,j} (\chi_i - \chi_j)^\top A (\chi_i - \chi_j) &= \sum_{i,j} \left(\chi_i^\top A \chi_i + \chi_j^\top A \chi_j - 2\chi_i^\top A \chi_j\right) \\ &= 2c \cdot \sum_i \chi_i^\top A \chi_i \; - 2 \cdot \sum_{i,j} \chi_i^\top A \chi_j\end{aligned}$$

Let us try to bound this last expression. On one hand, we know that $\chi_i^\top A \chi_i \geq 0$ since it is a product of matrices and vectors with non-negative entries. On the other hand, we can rewrite the last term as a product:

$$\sum_{i,j} \chi_i^\top A \chi_j = \left(\sum_i \chi_i\right)^\top A \left(\sum_i \chi_i\right)$$

This quantity, however, we can bound using the fact that the maximum eigenvalue of A is n (see Lemma 4 in Section 2.9). We get

$$\left(\sum_i \chi_i\right)^\top A \left(\sum_i \chi_i\right) \leq n \cdot \left\|\sum_i \chi_i\right\|^2$$

Since the vectors χ_i have disjoint support (again, each node y is assigned a unique color), they are orthogonal and so $\|\sum_i \chi_i\|^2 = \sum_i \|\chi_i\|^2$. Combining these results, we get the desired lower bound:

$$\sum_{i,j} (\chi_i - \chi_j)^\top A (\chi_i - \chi_j) \geq 0 - 2n \cdot \sum_i \|\chi_i\|^2 = -2n \cdot \sum_i \|\chi_i\|^2$$

■

Combining the lower and the upper bounds of Claims 2 and 3, we get

$$2(2\ell - n - 2)(c-1) \cdot \sum_i \|\chi_i\|^2 \geq -2n \cdot \sum_i \|\chi_i\|^2$$

Now since the coloring χ is non-empty, we have $\sum_i \|\chi_i\|^2 > 0$. Dividing the inequality above by this sum gives us $2(2\ell - n - 2)(c-1) \geq -2n$. This implies that

$$\ell \geq \frac{n}{2} + \left(1 - \frac{n}{2(c-1)}\right)$$

which was exactly what we had to prove.

Proof of Lemma 13. It remains to prove Lemma 13, i.e.

$$(\chi_i - \chi_j)^\top A(\chi_i - \chi_j) \leq (2\ell - n - 2) \cdot \|\chi_i - \chi_j\|^2$$

By Lemma 4 in Section 2.9 and the explicit form of the eigenvalues of A (Fact 2), it is sufficient show that all the Fourier coefficients of $(\chi_i - \chi_j)$ which correspond to eigenvalues $\lambda_z \geq 2\ell - n = n - 2(n-\ell)$ are 0. In other words, that $(\chi_i - \chi_j)$ is orthogonal to all the eigenvectors $\mathbf{v}_z$ whose eigenvalues are at least $(n - 2(n-\ell))$, i.e. $weight(z) \leq n - \ell$. But recall that on any subcube of dimension at least ℓ, the components of $(\chi_i - \chi_j)$ sum to 0! This turns out to be exactly the fact we need to in order to show that $\langle \mathbf{v}_z, \chi_i - \chi_j \rangle = 0$ whenever $\lambda_z \geq 2\ell - n$, and thus to prove Lemma 13.

Claim 4 *For any $z \in \{0,1\}^n$ with $weight(z) \leq n - \ell$ (i.e. $\lambda_z \geq 2\ell - n$), we have*

$$\langle \mathbf{v}_z, \chi_i - \chi_j \rangle = 0$$

Proof: Pick any vector $z = (z_1, \ldots, z_n) \in \{0,1\}^n$ with $weight(z) \leq n - \ell$, and let S be the support of z, i.e. $S = \{j : z_j = 1\}$. Note that $|S| \leq n - \ell$. Also, recall that $\mathbf{v}_z(y) = \frac{1}{\sqrt{2^n}} \cdot (-1)^{z \cdot y}$ (see Fact 2). Now consider any assignment a to the variables of S. By letting the remaining variables take on all possible values, we get some subcube of the hypercube, call it $\mathcal{H}_a$.

One the one hand, note that $\mathbf{v}_z$ is constant (either $1/\sqrt{2^n}$ or $-1/\sqrt{2^n}$) on that subcube, since if y and y' differ only on positions *not* in S, we will have $z \cdot y = z \cdot y'$. Call this value C_a. On the other hand, since the coloring is ℓ-balanced and since

$|S| \leq n - \ell$, the subcube $\mathcal{H}_a$ has dimension at least ℓ and so we know that both colors i and j have equal weight on $\mathcal{H}_a$. Thus summing the values of $(\chi_i - \chi_j)$ over this subcube gives 0.

Using the above two observations, we can easily show that $\langle \chi_i - \chi_j, \mathbf{v}_z \rangle$ is 0 by rewriting the inner product as a sum over all assignments to the variables in S:

$$\begin{aligned}\langle \chi_i - \chi_j, \mathbf{v}_z \rangle &= \sum_{y \in \mathcal{H}} \mathbf{v}_z(y)[\chi_i(y) - \chi_j(y)] = \sum_{\text{assignments } a} \left(\sum_{y \in \mathcal{H}_a} \mathbf{v}_z(y)[\chi_i(y) - \chi_j(y)] \right) \\ &= \sum_a C_a \cdot \left(\sum_{y \in \mathcal{H}_a} \chi_i(y) - \sum_{y \in \mathcal{H}_a} \chi_j(y) \right) = \sum_a C_a \cdot 0 = 0\end{aligned}$$

■

Equality conditions. As we proved Theorem 15 (and also Theorem 14), we might wonder which colorings can meet the bound of the theorem. Interestingly, such colorings are very structured, as we can see by tracing down our proof. Namely, consider the lower bound proved in Claim 3, i.e. that $\sum_{i,j} (\chi_i - \chi_j)^\top A (\chi_i - \chi_j) \leq -2n \sum_i \|\chi_i\|^2$. Going over the proof, we see that equality can occur only if two conditions occur.

On the one hand, we must have $\chi_i^\top A \chi_i = 0$ for all colors i. An easy calculation shows that $\chi_i^\top A \chi_i$ is 0 only when there is no edge of non-zero weight connecting two nodes of color i. Thus, this condition implies that the coloring is in fact a c-coloring in the traditional sense of complexity theory: no two adjacent nodes will have the same color.

One the other hand, the inequality $(\sum_i \chi_i)^\top A (\sum_i \chi_i) \leq n \cdot \| \sum_i \chi_i \|^2$ must be tight. This can only hold if the vector $\chi = \sum_i \chi_i$ is parallel to $(1, 1, \ldots, 1)$ since that is the only eigenvector with the largest eigenvalue n. But this means that all the weights $\chi(y)$ are the same, i.e. that the coloring must be *uniform*.

Extending the Bound to Larger Alphabets. Although the problem of constructing AONT's is usually stated in terms of bits, it is natural in many applications (e.g., secret-sharing) to consider larger alphabets, namely to consider T :

$\{0,\ldots,q-1\} \to \{0,\ldots,q-1\}^n$. All the notions from the "binary" case naturally extend to general alphabets as well, and so does our lower bound. However, the lower bound we obtain is mostly interesting when the alphabet size q is relatively small compared to n. In particular, the threshold $n/2$, which is so crucial in the binary case (when we are trying to encode more than $\log n$ bits), becomes n/q (recall, q is the size of the alphabet). This threshold becomes meaningless when $q > n$ which is not surprising at all, since in this case we can use Shamir's secret sharing [54] (provided q is the prime power) and achieve $\ell = k$ which is "incomparable" to n (and could be as small as 1). We also remark that the bound we state is tight if $q^k \leq n$ and can be achieved similarly to the binary case by using the analog of the Hadamard code over the alphabet of size q.

Theorem 16 *Let $T : \{0,\ldots,q-1\}^k \to \{0,\ldots,q-1\}^n$ be a perfect ℓ-*AONT*. Then*

$$\ell \geq \frac{n}{q} + \left(1 - \frac{q-1}{q} \cdot \frac{n}{q^k - 1}\right)$$

In particular, $\ell > n/q$ when $q^k > n$. Moreover, equality can only hold only for a uniform AONT*.*

Similarly to the binary case, we can also find a natural connection between such perfect ℓ-AONT's and weighted ℓ-balanced colorings of the "multi-grid" $\{0,\ldots,q-1\}^n$ with $c = q^k$ colors. And again, the bound of Theorem 15 extends here as well an becomes

$$\ell \geq \frac{n}{q} + \left(1 - \frac{q-1}{q} \cdot \frac{n}{c-1}\right)$$

As we said, we can use the same techniques as in our previous proof, with the following minor changes. We now work with the graph $\{0,\ldots,q-1\}^n$, which has an edge going between every pair of words that differ in a single position. For arithmetic purposes, we think of the nodes in this graph as vectors in $\mathbb{Z}_q^n$. If we let ω be a primitive q^{th} root of unity in $\mathbb{C}$ (e.g. $\omega = e^{2\pi i/q}$), then an orthonormal basis of the adjacency matrix of our graph is given by the q^n-dimensional vectors $\mathbf{v}_z$ for $z \in \{1,\ldots,q\}^n$,

where

$$\mathbf{v}_z(y) = \frac{1}{\sqrt{q^n}} \cdot \omega^{z \cdot y}$$

and now $z \cdot y$ is the standard dot product modulo q. The eigenvalue of $\mathbf{v}_z$ is $\lambda_z = n(q-1) - qj$ where $j = weight(z)$ (number of non-zero coordinates).

We define χ_i exactly as before. Claim 4 still holds (i.e. the Fourier coefficients of $(\chi_i - \chi_j)$ corresponding to large eigenvalues are 0). Constructing upper and lower bounds as above, we eventually get

$$(q\ell - n - q)(c-1) \sum_i \|\chi_i\|^2 \geq -n(q-1) \sum_i \|\chi_i\|^2$$

which implies the desired inequality. Equality conditions are the same.

This completes our study of perfect AONT's, and brings us back to the problem of constructing statistical and computational AONT's, which we do next.

5.2 Simple "Universal" 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 3.2. 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 obtain the following theorem.

Theorem 17 *Let $f : \{0,1\}^n \to \{0,1\}^k$ be a 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 $L \in \binom{n}{\ell}$, and $x_0, x_1 \in \{0,1\}^k$. We have to show that

$$\langle x_0, x_1, [r]_{\bar{L}}, f(r) \oplus x_0 \rangle \approx \langle x_0, x_1, [r]_{\bar{L}}, f(r) \oplus x_1 \rangle$$

This immediately follows from Corollary 1 and the definition of ERF (Equation (3.1)).

■

Notice that the size of the secret part $s = n$ and size of the public part $p = k$. As an immediate corollary of Theorems 7 and 17, we have:

Theorem 18 *Assume $\ell \leq s \leq poly(\ell)$. There exist probabilistic transformations $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^k$ (with secret output of length s and public output of length k) such that*

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

On the length of the public part. We notice that the length of the public part is k — the size of the message. Having a public output length necessarily equal to the size of the input seems to be a bit restrictive, but is actually quite natural. We can view the public part as the "masked original message". It takes exactly as much space for the application as the secret x used to take, and requires no protection (even though protection does not "hurt"). The size of the new secret part s is now a parameter that can be chosen pretty much arbitrarily (especially in the computational setting) depending on the level of security we desire to achieve. This level of security is now directly proportional to the extra-space that we use. To summarize, there is a very clear tradeoff between the amount of extra-space used and the exposure-resilience achieved.

Statistical AONT. Looking at the statistical construction, we get that $k = \ell - o(\ell)$ and s can be an arbitrary polynomial in ℓ. For example, we can set $\ell = s^\epsilon$ to achieve excellent exposure-resilience. The only drawback is that $k < \ell$. Unfortunately, similar to the case of ERF, this is unavoidable due to the following simple lemma, that also shows that our statistical construction is nearly optimal up to the lower order term.

Lemma 14 *If $T : \{0,1\}^k \to \{0,1\}^s \times \{0,1\}^p$ is a statistical ℓ-AONT with statistical deviation $\varepsilon < \frac{1}{2}$, then $k \leq \ell$.*

Proof: The proof is very similar to that of Lemma 6. Assume $k > \ell$. Take any $L \in \binom{s}{\ell}$, say $L = [\ell]$. To show that there exist $x_0, x_1 \in \{0,1\}^k$ contradicting Equation (3.2), we show that Equation (3.2) does not hold for *random* x_0 and x_1 by constructing a (computationally unbounded) distinguisher D who, given random x_0 and x_1, can successfully distinguish $[T(x_0)]_{\bar{L}}$ from $[T(x_1)]_{\bar{L}}$. Given $\langle x_0, x_1, w\rangle$, D simply tries out all possible 2^ℓ completions of w and inverts them using I. If he ever got x_1 back, he outputs 1, otherwise he outputs 0. Clearly, D always outputs 1 when w corresponds to x_1. When w corresponds to x_0, there are only 2^ℓ possible completions, and each can be inverted in only one way. Since x_1 is chosen at random for $\{0,1\}^k$, the probability that any of these inversions is equal to x_1 is at most $2^{\ell-k} \leq \frac{1}{2}$. Thus, the advantage of D is at least $\frac{1}{2} > \varepsilon$, a contradiction. ■

COMPUTATIONAL AONT. As with ERF's, the computational construction allows us to achieve the number of missing bits, ℓ, to be arbitrarily small compared to the input length k, beating the limitations of statistical AONT's. In essence, we can choose pretty much arbitrary ℓ and s given the input size k. For example, we can set $\ell = s^\epsilon$ to have an excellent exposure-resilience. We can also make the total output size $N = s + k$ to be dominated by the input size k, if we choose $s = o(k)$. This seems to be the best setting from a theoretical point of view. Namely, if $s = o(k)$ and $\ell = s^\epsilon$, we get that the total output size is $k + o(k)$, while the exposure-resilience is as small as we can wish.

SECRET-ONLY STATISTICAL AONT. 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, he must miss at least ℓ bits of the secret output). Let us apply this to our construction, where $p = k$. In the statistical setting, we obtain $\ell' = 2k+o(k) = O(k)$ and essentially any total output size $N = s + k > \ell'$. In fact, applying the first part of Theorem 11 to Theorem 17 and uniting the public and secret parts of the resulting ℓ-AONT, we get

Corollary 7 *For any $\omega(\log N) \leq \ell \leq N$ there exists a statistical secret-only ℓ-*AONT *$T : \{0,1\}^k \to \{0,1\}^N$, where $k = \Omega(\ell)$.*

Up to a small constant factor (which can be made as small as 2 if we relax $\ell = \omega(\log^2 N \operatorname{loglog} N)$), this is the best we can hope to achieve by Lemma 14.

Secret-only computational AONT. Let us turn now to the computational setting and get a secret-only AONT out of it. We see that $\ell' = \ell + k$ and $N = s + k$, as before, 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. Thus, if we make the total output size N small, we have only moderate exposure-resilience, and if we want to have very good exposure-resilience, we have to make the total output size large. As we demonstrated, these problems disappear if we have a public part, and there is really no reason not to. However, from a theoretic and aesthetic points of view, the following question is important to resolve:

Question 2 *Are there computational secret-only ℓ-*AONT*'s from k bits to N bits such that $N = O(k)$ and $\ell = N^\epsilon$, for any $\epsilon > 0$ (or even only $\ell = o(k)$)?*

We do not give a full answer to this question, but reduce it to the question of constructing a plausible function, which described in Section 5.3.

Adaptive AONT. We notice that Theorem 17 easily generalizes to the *adaptive* setting, where the adversary can adaptively choose which $(s - \ell)$ bits of the secret part to observe, as stated in Definition 14. This follows from the fact that Corollary 1 clearly relativizes to the setting with the oracle, who can provide the adversary any $(s - \ell)$ bits of the secret output. In particular, using the efficient probabilistic construction of adaptive ℓ-ERF's from Section 4.4, we get a probabilistic construction of statistical and computational adaptive AONT's with the same (and even slightly better) parameters as we did in the non-adaptive setting above. For example, in the statistical setting we can achieve adaptive ℓ-AONT's with $\ell \approx k$, and even secret-only

adaptive ℓ-AONT's with $\ell = O(k)$. This and Theorem 14 also show an exponential separation between statistical and perfect adaptive AONT's.

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, and would provide a positive answer to Question 2 if one constructs a function f as above. It can also be viewed as the speacial case of the OAEP construction of [8] in the Random Oracle model, and can be viewed as the first step towards abstracting the properties of random oracles that make this construction work as an AONT.

Removing random oracles from OAEP. Recall that the OAEP construction of Bellare and Rogaway [8] 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; see Figure 3-3). Boyko [16] formally showed that this is indeed an ℓ-AONT (where ℓ can be super-logarithmic in the security parameter). Let us 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 (so that we can try to replace random oracles by constructive functions). We look at the two extreme cases.

First, assume we know u completely 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 an ℓ-ERF! Thus, G must be an ERF, and this seems sufficient to handle the case when we miss ℓ bits of t.

Now assume that we know t completely and miss 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 an informally stated 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 informal property? Clearly, the easiest one is the identity function (assuming $n = k$).

OUR CONSTRUCTION. The above informal reasoning 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.

$$u = f(r) \oplus x \tag{5.7}$$

$$t = u \oplus r \tag{5.8}$$

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 $I(u,t) = u \oplus f(u \oplus t)$. This construction is illustrated in Figure 5-1 (and should be again compared with the original OAEP construction in Figure 3-3).

Theorem 19 *Assume f is such that both $f(r)$ and $(f(r) \oplus r)$ are length-preserving computational ℓ-*ERF*s. Then T above is a computational secret-only 2ℓ-*AONT*.*

Proof: Let $N = 2k$ be the size of the output, $L_1 = \{1 \ldots \ell\}$, $L_2 = \{\ell{+}1 \ldots 2\ell\}$. Take any $L \in \binom{N}{2\ell}$ 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 the 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

$$\langle x_0, x_1, [f(r) \oplus x_0]_{\bar{L}}, (f(r) \oplus r) \oplus x_0 \rangle \approx \langle x_0, x_1, [f(r) \oplus x_1]_{\bar{L}}, (f(r) \oplus r) \oplus x_1 \rangle$$

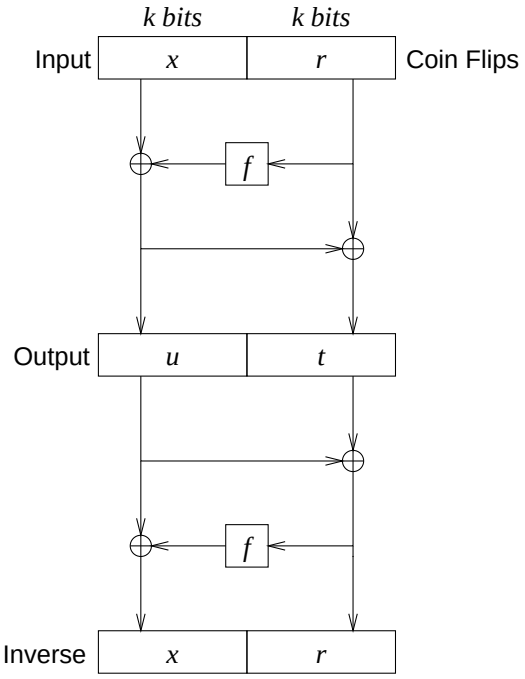

Figure 5-1: OAEP with H being the identity function and G being an ERF f.

Since $[f(r) \oplus x_i]_{\bar{L}} \oplus [f(r) \oplus r \oplus x_i]_{\bar{L}} = [r]_{\bar{L}}$ (for both $i = 0$ and $i = 1$), and $\langle A, B, C \rangle \approx \langle A, D, E \rangle$ iff $\langle A, B, C \oplus B \rangle \approx \langle A, D, E \oplus D \rangle$, the above is the same as

$$\langle x_0, x_1, [r]_{\bar{L}}, (f(r) \oplus r) \oplus x_0 \rangle \approx \langle x_0, x_1, [r]_{\bar{L}}, (f(r) \oplus r) \oplus x_1 \rangle$$

The result now immediately follows from Corollary 1 and the fact that $f(r) \oplus r$ is an ℓ-ERF (i.e., it satisfies Equation (3.1)).

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, the result follows from the fact that $f(r)$ is an ℓ-ERF. ■

We note that random oracle f clearly satisfies the conditions of the above 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 Theorem 19 is quite reasonable, even though we leave open the question of constructing such f based on standard assumptions. We also remark that non-trivial length-preserving ERF's can exist only in the computational sense, since $\ell \geq k$ for

any statistical ERF (by Lemma 6).

5.4 Computational AONT implies OWFs

We have seen in Lemma 14 that statistical ℓ-AONT's can exist only for $k \leq \ell$. We now show a strong dual statement that once $\ell < k$, computational ℓ-AONT's in fact imply the existence of one-way finctions.

Theorem 20 *Assume we have a computational* ℓ-AONT $T : \{0,1\}^k \rightarrow \{0,1\}^s \times \{0,1\}^p$ *where* $\ell < k$. *Then one-way functions exist.*

Proof: To show that OWF's exist it is sufficient to show that *weak* OWF's exist [29] (see also Section 2.4). Fix $L = [\ell] \subseteq [s]$. Define

$$g(x_0, x_1, b, r) = \langle x_0, x_1, [y]_{\bar{L}} \rangle$$

where $y = T(x_b;\ r)$. Intuitively, it might seem that the fact that g is a weak OWF should follow immediately from the fact that T is an AONT. Namely, to invert g on a random input the adversary needs to determine b correctly, which he cannot do significantly better than guessing, by the security of the AONT. While this intuition is somewhat correct in its spirit, there are some problems that have to be overcome. First, to "successfully invert" g the adversary does not have to come up with the preimage that we "started from". In particular, it could be that for most x_0, x_1, b, r we started from, it is possible to achieve the same output with $x_0, x_1, 1-b$ and some other randomness r' (so that the adversary does not necessarily *have to* produce b to succeed). To rule out this possibility, we will use the fact that $\ell < k$ and that T is *invertible*. Secondly, it will not be immeditely clear why the fact that b is hard to guess implies that g is a weak OWF, but this will follow from a careful analysis, which we present now.

Assume that g is a not weak OWF. 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 $\Pr(B) \geq \frac{3}{4}$. Also, if B does not happen, F flips a coin and succeeds with probability $1/2$. So assume A succeeds inverting. 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 $[T(x_{1-b};\ r')]_{\bar{L}} = z$ (equivalently, $g(x_0, x_1, 1-b, r') = g(x_0, x_1, b, r)$). If U does not happen and A succeeded inverting, we know that $\tilde{b} = b$ (i.e., F succeeds with probability 1), as $(1-b)$ is an impossible answer. On the other hand, if U *does* happen and A succeeds inverting, we claim that F succeeds with probability exactly $1/2$, which we argue next.

Indeed, *conditioned on* $B \wedge U$, our experiment can be view as follows. Let D be the distribution on z induced by choosing a random x and setting $z \leftarrow [T(x)]_{\bar{L}}$, and let D_z be the conditional distribution on x induced by choosing y this way. We first choose $z \leftarrow D$, and then independently sample brand new $x_0, x_1 \leftarrow D_z$. Notice, *no bit b is generated yet*! Then we give x_0, x_1, y to F, who passes them to A. If A succeeds inverting (outputs $\tilde{b}$ and $\tilde{r}$ s.t. $z = [T(x_{\tilde{b}};\ \tilde{r}]_{\bar{L}}$), we let F output $\tilde{b}$ as before. Otherwise, we repeat the whole experiment from scratch. Only after we finished this experiment (i.e. A eventually succeeded) do we choose at random the "right" bit b. It is easy to see that this experiment is completely equivalent to our original experiment conditioned on $B \wedge U$ (provided the latter has non-zero probability). On the other hand, since b is generated afterwards, $\Pr(\tilde{b} = b) = \frac{1}{2}$ indeed.

Combining the above observations and using $\Pr(X \wedge \overline{Y}) \geq \Pr(X) - \Pr(Y)$, we get:

$$\begin{aligned}
\Pr(\tilde{b} = b) &\geq \frac{1}{2}\Pr(\overline{B}) + \Pr(B \wedge \overline{U}) + \frac{1}{2}\Pr(B \wedge U) \\
&= \frac{1}{2} + \frac{1}{2} \cdot \Pr(B \wedge \overline{U}) \\
&\geq \frac{1}{2} + \frac{1}{2} \cdot (\Pr(B) - \Pr(U)) \\
&\geq \frac{1}{2} + \frac{1}{2} \cdot \left(\frac{3}{4} - \Pr(U)\right)
\end{aligned}$$

To get a contradiction, we show that $\Pr(U) \leq 2^{\ell-k}$, which is at most $\frac{1}{2} < \frac{3}{4}$ since $\ell < k$. 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 some 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'. Since x' was chosen at random, $\Pr(U) \leq 2^{\ell-k}$ indeed. ∎

We note that the result is essentially optimal (up to the lower order term), since by Theorem 18 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, as stated in Corollary 7.

5.5 Worst-case/Average-case Equivalence of AONT

In the definition of AONT we require that Equation (3.2) 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 and independent $x_0, x_1 \in \{0,1\}^k$. We call an AONT secure against (possibly only) the uniform distribution an *average-case* AONT. Note, for instance, the proofs of Theorem 20 and Lemma 14 work for average-case AONT's as well, since we used random x_0 and x_1 in both proofs. Thus, statistical average-case

ℓ-AONT's are impossible for $\ell < k$ and computational average-case ℓ-AONT's imply OWF's if $\ell < k$.

A natural question to ask is whether average-case AONT's imply (regular) AONT's with comparable parameters, which can be viewed as the worst-case/average case equivalence. We notice that in the perfect setting an average-case AONT is also a worst-case AONT (for example, this follows from the equivalent Definition 15 of a perfect AONT), so there is nothing to show here. Perhaps surprisingly, we show that up to a constant factor, the worst-case and the average-case notions are indeed identical in the statistical and the computational settings, as well. Below we assume without loss of generality that our domain $\{0,1\}^k$ is a finite field (e.g. $GF(2^k)$), so that addition and multiplication are defined.

Theorem 21 *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 from $\{0,1\}^k$ 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'.

Proof: 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'). Before arguing that T' is an ℓ-AONT, let us define some terminology. Given an output of T' of the form $y' = \langle t_1, t_2, t_3, t_4 \rangle$, we let the quadruple $\langle a_1, a_2, b, z \rangle$, where $a_1 = I(t_1)$, $a_2 = I(t_2)$, $b = I(t_3)$ and $z = I(t_4)$, be the *effective inputs* of y' (while the actual input $x' = (z - b)/(a_1 + a_2)$). In general, a_1 will typically stand for the first effective input to T', a_2 — for the second, b — for the third, and z — for the last.

Assume now that T' is not an ℓ-AONT, that is for some $L' \in \{\binom{4s}{4\ell}\}$, $x'_0, x'_1 \in \{0,1\}^k$

(obviously, $x'_0 \neq x'_1$) we have

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

And assume that an adversary A' distinguishes the above two distributions. First, let us define a subset $L \in \binom{s}{\ell}$ that would contradict the fact that T is an average-case ℓ-AONT. We construct L by looking at which part of the output of T' has the most bits in L'. Formally, let $L_j = \{m \in [\ell] \mid m + (j-1)\ell \in L'\}$, $j = 1, 2, 3, 4$. Since $|L'| = 4\ell$, some $|L_j| \geq \ell$. We let L be any ℓ-element subset of this L_j. Thus, if $j = 1$ the adversary misses "L-bits" of $T(a_1)$, if $j = 2$ — of $T(a_2)$, if $j = 3$ — of $T(b)$, and if $j = 4$ — of $T(z)$.

Let x_0, x_1 be selected at random from $\{0,1\}^k$, $i \in_R \{0,1\}$, $w \leftarrow [T(x_i)]_{\bar{L}}$, and we have to construct an adversary A (that would use A') that can determine i with probability non-trivially better than $\frac{1}{2}$ when given $\langle x_0, x_1, w \rangle$. Here is a general strategy of A. He will *implicitly* (i.e., as a thought experiment pretending that he knows i) create y' in such a way that *irrespective of i being 0 or 1*, y' will correspond to x'_i (i.e., $I'(y') = x'_i$). In addition, y' will be (statistically close to) a random output of $T'(x'_i)$. However, A would be able to *explicitly* compute $w' = [y']_{\bar{L}'}$ using his input w. By handing this w' to the assumed good distinguisher A', A would be able to determine i as well as A' does. Thus, A succeeds in "blindly translating" w to the right w'.

Before showing how to (implicitly) construct y', we see what relations it should satisfy. Let $\langle a_1, a_2, b, z \rangle$ be the effective inputs of y'. Since they should correspond to x'_i, we must have

$$(a_1 + a_2) \cdot x'_i + b = z \tag{5.9}$$

Moreover, $\langle a_1, a_2, b, z \rangle$ should be (statistically close to) *random* satisfying the corresponding equation above (subject to $a_1 + a_2 \neq 0$). To implicitly compute y', A will implicitly set one of a_1, a_2, b, z to x_i (which one depends on j that "produced" L; namely, set $a_1 = x_i$ if $|L_1| \geq \ell$, set $a_2 = x_i$ if $|L_2| \geq \ell$, etc). Assume for concreteness that $j = 1$ and so $a_1 = x_i$. The remaining three parameters (in our case, a_2, b, z) A will compute *explicitly* in such a way that it *does not matter whether i is 0 or 1* (as

long as the implicit parameter, here a_1, is equal to x_i). Assuming A can succeed in doing so, we will be done since he can explicitly produce $w' = \langle w, T(a_2), T(b), T(z)\rangle$. Similar technique holds for $j = 2, 3, 4$.

We now show how this can indeed be done for any j.

- $|L_1| \geq \ell$. We know that

$$\begin{aligned}&\langle x'_0, x'_1,\ [T(a_1)]_{\bar{L}},\ T(a_2),\ T(b),\ T((a_1+a_2)\cdot x'_0 + b)\rangle \quad \not\approx \\ &\langle x'_0, x'_1,\ [T(a_1)]_{\bar{L}},\ T(a_2),\ T(b),\ T((a_1+a_2)\cdot x'_1 + b)\rangle\end{aligned}$$

  Clearly, we should (implicitly) make $a_1 = x_i$ (which is random since x_i is random). In order to explicitly 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 the resulting a_2 is random), so we can ignore this case happening for the statistical or computational settings. Then we immediately observe that by construction, $\langle x_i, a_2, b, z\rangle$ satisfy $(x_i + a_2)\cdot x'_i + b = z$. Moreover, this is a *random* quadruple of inputs to T used in computing $T'(x'_i)$ (technically, statistically close to it). Hence, we can explicitly produce $w' = \langle w, T(a_2), T(b), T(z)\rangle$ and, by the previous argument, obtain a contradiction to the fact that T is an average-case ℓ-AONT.

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

- $|L_3| \geq \ell$. We know that

$$\begin{aligned}&\langle x'_0, x'_1,\ T(a_1),\ T(a_2),\ [T(b)]_{\bar{L}},\ T((a_1+a_2)\cdot x'_0 + b)\rangle \quad \not\approx \\ &\langle x'_0, x'_1,\ T(a_1),\ T(a_2),\ [T(b)]_{\bar{L}},\ T((a_1+a_2)\cdot x'_1 + b)\rangle\end{aligned}$$

  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 in computing $T'(x'_i)$ (technically, statistically close to it). Hence, we can explicitly produce $w' = \langle T(a_1), T(a_2), w, T(z)\rangle$ and, by the previous argument, obtain a contradiction to the fact that T is an average-case ℓ-AONT.

- $|L_4| \geq \ell$. We know that

$$\begin{aligned}&\langle x'_0, x'_1,\ T(a_1),\ T(a_2),\ T(b),\ [T((a_1+a_2)x'_0 + b)]_{\bar{L}}\rangle \quad \not\approx \\ &\langle x'_0, x'_1,\ T(a_1),\ T(a_2),\ T(b),\ [T((a_1+a_2)x'_1 + b)]_{\bar{L}}\rangle\end{aligned}$$

  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 $a_1 + a_2$)

$$a \cdot x'_0 + b = x_0$$

$$a \cdot x'_1 + b \;=\; x_1$$

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 can explicitly produce $w' = \langle T(a_1), T(a_2), T(b), w \rangle$ and, by the previous argument, obtain a contradiction to the fact that T is an average-case ℓ-AONT.

■

Chapter 6

Conclusions

We now briefly summarize the contributions of this thesis.

All-Or-Nothing Transforms. Our main motivation came from the problem of partial key exposure and related questions. We have proposed to use the *All-Or-Nothing Transform* [51], which also has many other applications, as the most direct way to solve these problems. Up to date, however, there were no provable constructions of the AONT in the standard model of computation, based on standard computational assumptions (e.g., without random oracles [16], ideal ciphers [21] and having strong enough security properties [60]). We gave very natural and simple definitions of AONT in the perfect, statistical and computational settings, together with the first provable constructions in all these settings. We have also shown almost matching lower bounds, making our constructions nearly optimal. In particular, our lower bound on perfect AONT's is of independent interest, relates to an interesting question of balanced colorings of the hypercube, and non-trivially extends the lower bound of Friedman [28] for such colorings.

Exposure-Resilient Functions. The key ingredient in our approach is an interesting new primitive which we called an *Exposure-Resilient Function*. We demonstrated that this primitive has natural applications in combating key exposure, and also has many other applications (for example, it can be viewed as a "super-secure" pseudorandom generator), making it a very interesting notion in its own right. Simi-

larly to AONT's, we have shown how to build essentially optimal ERF's in the perfect, statistical and computational settings.

Other contributions. We have also examined other properties of AONT's and ERF's (e.g., worst-case/average-case equivalence of AONT's, equivalence of "interesting" computational AONT's and ERF's to one-way functions), as well as several other results of independent interest. For example, we formally written down the notion of δ-sure extractors (which we used in constructing adaptively secure ERF's, and which have other applications) suggested to us and implicitly used by Trevisan and Vadhan [62], and also gave a simple "generic" proof that semantic security is equivalent to indistinguishability [33].

Open Problems. There are still several interesting questions remaining open. Two of them are summarized in Questions 1 and 2. Namely, to have an explicit construction of adaptively secure statistical ERF's, and to have constructions of secret-only computational AONT's with good exposure-resilience and short output length. Somewhat related to the latter problem is the question of designing ERF's, pseudorandom generators or pseudorandom functions having "nice properties" with respect to the exclusive OR operator. For example, we reduced the question of constructing "good" secret-only AONT's to the question of constructing a length-preserving ERF f such that $f(x) \oplus x$ is also an ERF. Similarly, the "ideal block cipher" construction of Desai [21] can be analyzed in the standard model if one replaces the ideal cipher with a pseudorandom function family that remains pseudorandom if some of the outputs are XORed with the random seed.

Exposure-Resilient Cryptography. To recap everything once again, we observed that standard cryptographic notions and constructions do not guarantee any security even if a *tiny* fraction of the secret entity is compromised. We then put forward the notion of *Exposure-Resilient Cryptography*, which is concerned with building cryptographic primitives that remain *provably secure* even if *most* of the secret is exposed.

Bibliography

[1] M. Abdalla, S. Miner and C. Namprempre. Forward Security in Threshold Signature Schemes Available at Cryptology ePrint Archive, Listing for 2000, report 031, `http://eprint.iacr.org/2000/031/`.

[2] M. Abdalla, L. Reyzin. A New Forward-Secure Digital Signature Scheme. Available at Cryptology ePrint Archive, Listing for 2000, report 002, `http://eprint.iacr.org/2000/002/`.

[3] R. Anderson. Invited Lecture. *Fourth Annual Conference on Computer and Communication Security*, ACM, 1997.

[4] M. Bellare and A. Boldyreva. The Security of Chaffing and Winnowing Available at Cryptology ePrint Archive, Listing for 2000, report 010, `http://eprint.iacr.org/2000/010/`.

[5] M. Bellare, R. Canetti and H. Krawczyk. Keying hash functions for message authentication. In *Proc. of Crypto*, pp. 1–15, 1996.

[6] M. Bellare J. Kilian and P. Rogaway. The security of ciper block chaining. In *Proc. of Crypto*, pp. 341–358, 1994.

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

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

[9] M. Bellare, J. Rompel. Randomness-Efficient Oblivious Sampling. In *Proc. of 35th FOCS*, 1994.

[10] C. Bennett, G. Brassard, and J. Robert. Privacy Amplification by public discussion. In *SIAM J. on Computing*, pp. 17(2):210–229, 1988.

[11] J. Bierbrauer, K. Gopalakrishnan, D. Stinson. Orthogonal Arrays, resilient functions, error-correcting codes and linear programming bounds. In *SIAM J. of Discrete Math*, 9:424–452, 1996.

[12] J. Black, S. Halevi, H. Krawczyk, T. Krovetz, P. Rogaway. UMAC: Fast and Secure Message Authentication. In *Proc. of Crypto*, pp. 216–233, 1999.

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

[14] M. Blaze. High Bandwidth Encryption with low-bandwidth smartcards. In *Fast Software Encryption*, pp. 33–40, 1996.

[15] V. Boyko. On All-Or-Nothing Transforms and Password-Authenticated Key Exchange Protocols. PhD Thesis, MIT, available at `http://theory.lcs.mit.edu/~cis/cis-theses.html`, May 2000.

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

[17] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz and A. Sahai. Exposure-Resilient Fucntions and All-Or-Nothing-Transforms. In *Proc. of EuroCrypt*, pp. 453–469, 2000.

[18] R. Canetti, O. Goldreich and S. Halevi. The Random-Oracle Model, Revisited. In *Proc. of STOC*, pp. 209–218, 1998.

[19] L. Carter and M. Wegman. Universal classes of hash functions. *JCSS*, vol. 18, pp. 143–154, 1979.

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

[21] A. Desai. The Security of All-Or-Nothing Encryption: Protecting Against Exhaustive key Search. To appear in *Crypto*, August 2000.

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

[23] W. Diffie, P. van Oorschot and M. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2:107–125, 1992.

[24] Y. Dodis, A. Sahai and A. Smith. Optimal Lower Bound for Perfect All-Or-Nothing Transforms. Preliminary version in submission, April 2000.

[25] 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.

[26] T. ElGamal. A public key cryptosystem and a signature scheme based on the discrete logarithms. *IEEE Transactions of Information theory*, 31(4):469–472, 1985.

[27] M. Etzel, S. Patel, and Z. Ramzan. SQUARE HASH: Fast Message Authentication via Optimized Universal Hash Functions. In *Proc. of Crypto*, pp. 234–251, 1999.

[28] J. Friedman. On the Bit Extraction Problem In *Proc. of FOCS*, pp. 314–319, 1992.

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

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

[31] O. Goldreich, S. Goldwasser and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):210–217, 1986.

[32] O. Goldreich, A. Wigderson. Tiny Families of Functions with Random Properties: A Quality-Size Trade-off for Hashing. In *Electronic Colloquium on Computational Complexity (ECCC)*, Rechnical Report TR94-002, 1994. Revised December 1996. `http://www.eccc.uni-trier.de/eccc/`. Preliminary version in *Proc. of STOC*, pp. 574–583, 1994.

[33] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

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

[35] 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.

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

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

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

[39] K. Kurosawa, T. Johansson and D. Stinson. Almost k-wise independent sample spaces and their cryptologic applications. Submitted to *J. of Cryptology*, preliminary version appeared in *Proc. of EuroCrypt*, pp. 409–421, 1997.

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

[41] S. Matyas, M. Peyravian and A. Roginsky. Encryption of Long Blocks Using a Short-Block Encryption Procedure. Available at `http://grouper.ieee.org/groups/1363/P1363a/LongBlock.html`.

[42] S. Micali, C. Rackoff, B. Sloan. The Notion of Security for Probabilistic Cryptosystems. In *SIAM J. on Computing*, 17(2):412–426, 1988.

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

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

[45] N. Nisan, A. Ta-Shma. Extracting Randomness: A Survery and a New Construction. In *JCSS*, 58(1):148–173, 1999.

[46] N. Nisan, D. Zuckerman. Randomness is Linear in Space. In *JCSS*, 52(1):43–52, 1996.

[47] K. Prutkov. `http://129.22.144.118/raznoe/kprutkov.htm`.

[48] J. Radhakrishnan and A. Ta-Shma. Tight bounds for depth-two superconcentrators. In Proc. of FOCS, pp. 585–594, 1997.

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

[50] R. Raz, O. Reingold, S. Vadhan. Extracting all the Randomness and Reducing the Error in Trevisan's Extractors. In Proc. of STOC, pp. 149–158, 1999.

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

[52] R. Rivest. Chaffing and Winnowing: Confidentiality without Encryption. *CryptoBytes (RSA Laboratories)*, 4(1):12–17, 1998.

[53] C. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

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

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

[56] C. Shannon. Communication Theory of Secrecy Systems. In *Bell System Technical Journal*, 28(4):656–715, 1949.

[57] 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.

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

[59] A. Srinivasan, D. Zuckerman. Computing with Very Weak Random Sources. In SIAM J. on Computing, 28(4):1433–1459, 1999.

[60] D. Stinson. Something About All or Nothing (Transforms). Available from `http://cacr.math.uwaterloo.ca/~dstinson/papers/AON.ps`, 1999.

[61] L. Trevisan. Construction of Extractors Using PseudoRandom Generators. In Proc. of STOC, pp. 141–148, 1999.

[62] L. Trevisan, S. Vadhan. Extracting Randomness from Samplable Distributions. To appear in *Proc. of FOCS*, 2000.

[63] U. Vazirani. Towards a Strong Communication Complexity Theory or Generating Quasi-Random Sequences from Two Communicating Semi-Random Sources. In *Combinatorica*, 7(4):375–392, 1987.

[64] D. Zuckerman. Randomness-optimal oblivious sampling. In *Random Structires and Algorithms*, 11(4):345–367, 1997.