Maintaining Secrecy when Information Leakage is Unavoidable

by

Adam Davison Smith

B.Sc. Mathematics and Computer Science, McGill University, 1999,
S.M. Electrical Engineering and Computer Science,
Massachusetts Institute of Technology, 2001

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

Doctor of Philosophy in Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2004

Author .....
Department of Electrical Engineering and Computer Science
December 23, 2004

Certified by .....
Madhu Sudan
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

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

Maintaining Secrecy when Information Leakage is Unavoidable

by

Adam Davison Smith

Submitted to the Department of Electrical Engineering and Computer Science
on December 23, 2004, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science

Abstract

Sharing and maintaining long, random keys is one of the central problems in cryptography. This thesis provides about ensuring the security of a cryptographic key when partial information about it has been, or must be, leaked to an adversary. We consider two basic approaches:

1. *Extracting a new, shorter, secret key from one that has been partially compromised.* Specifically, we study the use of noisy data, such as biometrics and personal information, as cryptographic keys. Such data can vary drastically from one measurement to the next. We would like to store enough information to handle these variations, without having to rely on any secure storage—in particular, without storing the key itself in the clear.

   We solve the problem by casting it in terms of key extraction. We give a precise definition of what "security" should mean in this setting, and design practical, general solutions with rigorous analyses. Prior to this work, no solutions were known with satisfactory provable security guarantees.

2. *Ensuring that whatever is revealed is not actually useful.* This is most relevant when the key itself is sensitive—for example when it is based on a person's iris scan or Social Security Number. This second approach requires the user to have some control over exactly what information is revealed, but this is often the case: for example, if the user must reveal enough information to allow another user to correct errors in a corrupted key. *How can the user ensure that whatever information the adversary learns is not* useful *to her?*

   We answer by developing a theoretical framework for separating leaked information from useful information. Our definition strengthens the notion of *entropic security*, considered before in a few different contexts. We apply the framework to get new results, creating (a) encryption schemes with very short keys, and (b) hash functions that leak no information about their input, yet—paradoxically—allow testing if a candidate vector is close to the input.

One of the technical contributions of this research is to provide new, cryptographic uses of mathematical tools from complexity theory known as *randomness extractors*.

Thesis Supervisor: Madhu Sudan
Title: Professor of Electrical Engineering and Computer Science

Acknowledgements

Several people acted as mentors—at various levels of formality—during my graduate studies. They provided advice, research ideas, encouragement and support. In chronological order: Claude Crépeau, Shafi Goldwasser, Madhu Sudan, Silvio Micali, Rafi Ostrovsky and Cynthia Dwork. Without their help I would not have (respectively) gotten started in computer science research, made it through my first year at MIT, had the flexibility to pursue my own research ideas, had the confidence to push those ideas as far as possible, gotten into cutting edge research on cryptographic protocols, or learned taste and discrimination in my choice of research projects. For their help, support and patience, my thanks.

I am also grateful to the collaborators who have helped with my research, or allowed me to help them with theirs: Andris Ambainis, Howard Barnum, Shuchi Chawla, Claude Crépeau, Yevgeniy Dodis, Giovanni Di Crescenzo, Cynthia Dwork, Matthias Fitzi, Daniel Gottesman, Martin Hirt, Thomas Holenstein, Jon Katz, Silvio Micali, Frank McSherry, Moses Liskov, Anna Lysyanskaya, Rafail Ostrovsky, Chris Peikert, Charlie Rackoff, Leo Reyzin, Amit Sahai, Ronen Shaltiel, abhi shelat, Alain Tapp, Luca Trevisan, Hoeteck Wee, and Ke Yang.

Most importantly, I thank the many friends and family members who have made my graduate studies such a pleasure. They are too numerous to mention all here, but a few stick out: abhi shelat (who taught me, by example, the value of originality and to a lesser extent of the Lower Case), Adam Holt, April Rasala, Ayesha Islam, Alexandra London-Thompson, Bobby Sampson, Eliot, Rebecca, Simone, and my parents.

Finally, I thank my future wife Sofya for her friendship and her love. Her contribution to my life is too large to fit on this page.

Bibliographic Note

The results of the first part of the thesis were published as joint work with Yevgeniy Dodis and Leonid Reyzin [30]. That work, in turn, came out of important initial discussions with Rafail Ostrovsky and later conversations with Piotr Indyk on embeddings of the edit metric.

The results of the second part of the thesis are more recent, and not yet published elsewhere. They appear for now in a pair of manuscripts which were written jointly with Yevgeniy Dodis [31, 32].

Finally, I thank the many people who helped with this research through discussions, observations and thoughtful objections: Rafi Ostrovsky, Piotr Indyk, Salil Vadhan, Avi Wigderson, Madhu Sudan, Jonathan Katz, Xavier Boyen, Noga Alon, Venkat Guruswami, Yehuda Lindell, Alex Samorodnitsky, Henry Cohn, Pim Tuyls, Jonathan Connell, and Chris Peikert.

Contents

Chapter 1

Introduction

Generating shared secret keys and passwords is one of the central tasks in cryptography. A key is typically a long, random string of bits known only to a few people, and it can be used for various cryptographic tasks; the most common examples are encrypting a message so that it cannot be read by others, authenticating a message to ensure that it comes from the correct sender, and identifying a user as the possessor of a copy of the key. A password is similar, but is used only for access control. For now, we will use "key" and "password" interchangeably.

Sharing—and maintaining—a long, random secret key is difficult: whoever generates the key must communicate it to the other parties, and the parties must have some way of storing the key (or some derived information) reliably and securely. If the "key" in question is a person's password, this means memorizing a long string and later remembering it exactly—something that most people are not easily capable of. The main issues addressed in this thesis are handling keys that cannot be stored reliably, and making use of a very short key when a longer one would normally be required.

More generally, this thesis provides new results about modeling and ensuring *provable* security when partial information about a key has been, or must be, leaked to an adversary. For example, when a user's storage is not reliable, one must reveal some information to allow correction of the errors which appear in the key; when the storage is not fully secure, then whatever information is stored may be learned by an adversary.

The main goal is to understand—and limit—exactly what information has been leaked. We consider two basic approaches:

1. *Extracting a new, shorter, secret key* from one that has been partially compromised. This is relevant when some information is leaked about a key, but the users still need it for encryption, authentication or identification.

2. *Ensuring that whatever is revealed is not actually useful.* This is most relevant when the key itself is sensitive—for example when it is based on a person's fingerprint, iris scan or Social Security Number. This second approach requires the user to have some control over exactly what information is revealed, but

this is often the case: for example, if the user must reveal enough information to allow another user to correct errors in a corrupted key.

Specifically, this thesis proposes new models and new techniques for several problems with a (perhaps surprisingly) similar flavor. The main results of this thesis can be divided into two areas, following the two approaches above:

- The first rigorous framework for using noisy data, such as biometrics and personal information, as cryptographic keys. *Biometrics* are characteristics that are inherent to a person's body. Fingerprints, face images, voice recordings, and iris scans are all common examples. Using such data as keys has been proposed to save users from the task of remembering long keys.

  The main challenge is that biometrics vary drastically from one measurement to the next. A fingerprint, for example, varies due to the position of the finger, humidity, swelling, etc. One must store enough information to correct these variations, without storing the key itself in the clear. Prior to this work, no solutions were known with satisfactory provable security guarantees.

  We solve the problem by casting it in terms of key extraction problem, as in approach (1) above. We propose a rigorous definition of exactly what "security" should mean in this setting. This allows us to analyze existing constructions and compare their performance. We also provide new, provably secure constructions which improve on existing ones, and prove lower bounds on the achievable parameters.

- A formalization of approach (2) above: we develop a theoretical framework for separating leaked information from useful information. Our definition strengthens the notion of *entropic security*, considered before in a few different contexts. Roughly, the definition goes as follows: revealing information Y about secret X is safe if there is no function of X which is easier to predict with Y than without it (i.e. Y may leak some global information about X, but there is no function of X — in particular, no substring or parity check — that becomes easier to predict).

  We characterize entropic security in terms of mathematical tools from complexity theory ("randomness extractors," explained below). This insight allows us to design (1) new encryption schemes with a short key, and (2) hash functions which leak nothing useful about their inputs, yet—paradoxically—allow testing if a candidate vector is *close to* the input.

The thesis is divided into two parts, following the bullets above. This introduction chapter also consists of two corresponding main sections (1.1 and 1.2).

The basic philosophy of both sections is the same. We focus on provably secure protocols. The contributions of the thesis are thus at three levels: rigorous definitions of security, new protocols, and proofs that those protocols satisfy the definition. Each of these pieces is important. The definitions, in particular, allow one to understand what types of attacks the protocol is known to be secure against, and what attacks

were not considered. They also allow one to understand the exact assumptions under which the proof holds. Beyond these practical advantages, precise definitions of security allow one to understand the limits of a particular notion, say by proving bounds on the resources (computing time, interaction, etc) that are needed for a protocol to satisfy a definition, or bounds on the parameters (error probabilities, attack success probabilities) that are possible.

A Few Words on Information-Theoretic Cryptography The security of many cryptographic protocols depends on assuming that a particular problem is computationally hard, for example "the probability that the adversary can factor a random product of two primes of length k is negligibly small." There are in fact two assumptions hidden in such a statement, one about the resources necessary to factor large integers, and another about the adversary's computing power. Such hypotheses are bunched under the umbrella of "computational" assumptions. While it is reasonable to assume that an adversary does not have time to execute 2^{1000} instructions on a computer (for thermodynamic reasons—there is not enough time and energy in the universe), no one has yet proven any good lower bounds on the number of steps required to solve a particular problem, and computational assumptions remain problematic.

Protocols whose security does *not* depend on computational assumptions are called "unconditionally" or "information-theoretically" secure. Such protocols are studied since they avoid assuming unproven lower bounds on the resources required to solve particular problems. They make good building blocks for larger protocols, since they can be plugged into any other context without having to make additional assumptions. There are also many situations where it is not known how to apply techniques from "computational" cryptography, and so information-theoretic tools, which are generally easier to work with, become very valuable.

There are a few mathematical tools from information-theoretic cryptography which will be used many times in the thesis. A key concept, introduced by Shannon [73], is the *entropy*, or uncertainty, of a random variable. We use several measures of entropy, but all agree in the simplest case where a random variable is distributed uniformly over some subset of its possible values. If it takes on N different values with equal probability, then the entropy of the variable is $\log(N)$ (in the thesis, logarithms are in base 2 by default); a variable which takes on a single value with probability 1 has no entropy at all, and a variable which is distributed uniformly over all binary strings of length n has entropy n.

Another key tool, much more recent, is a *randomness extractor*. Informally, an extractor is a function which takes as input some imperfect source of randomness (the various bits of which may be biased or correlated), and produces as output a short, uniformly random string, ideally with length equal to the entropy of the input. The magic of an extractor is that it should work for *any* "source" (i.e. probability distribution on inputs) which has sufficient entropy. To work on such a large class of distributions, a randomness extractor actually needs an additional input: a short, uniformly random "seed." The ideal output length is thus the sum of the length

of the seed and the entropy of the input. Extractors first arose in cryptography and in complexity theory, and have since had many applications in both areas—see [72, 85, 86] for surveys.

One of the contributions of this research is to provide new uses of the tools above in cryptography. First, we apply them to the problem of error-prone keys. We essentially construct noise-resilient extractors. The resulting protocols can provably handle any distribution on passwords that has sufficient entropy. Previous proofs of security required the system designer to know the distribution of, say, human fingerprints, exactly.

Second, the chapters on entropic security demonstrate a new use for extractors. The main result of Chap. 5 is that the output of an extractor reveals no useful information — in a precise sense — about the imperfect random source used as input. This theorem allows us to give several new protocols, lower bounds, and proofs of security. By using known constructions of extractors, we also simplify current protocols and proofs of security for encryption of high-entropy messages and "perfectly one-way" hash functions. The final chapter combines both uses of extractors, constructing entropically-secure fuzzy extractors.

The next two sections of this introduction (Sections 1.1 and 1.2) discuss the results of each part of the thesis in more detail, as well as the related literature.

1.1 Fuzzy Extractors: Cryptography with Noisy Data

Cryptography traditionally relies on uniformly distributed random strings for its secrets. Reality, however, makes it difficult to create, store, and reliably retrieve such strings. Strings that are neither uniformly random nor reliably reproducible seem to be more plentiful. For example, a random person's fingerprint or iris scan is clearly not a uniform random string, nor does it get reproduced precisely each time it is measured. Similarly, a long pass-phrase (or answers to 15 questions [39] or a list of favorite movies [48]) is not uniformly random and is difficult to remember for a human user. Part I of this thesis is about using such nonuniform and unreliable secrets in cryptographic applications.

An Application: Biometrics The focus of this research is theoretical, but much of the previous work is specific to biometrics, and so we begin with some background and terminology.

The term biometrics originally meant any measurement of biological system; the term later came to mean physiological or behavioral characteristics that could be used to identify a person.[1] The concept is by no means new—we instinctively recognize

[1]This is still the source of some confusion. The journal *Biometrics* (International Biometric Society) covers statistical issues in biology and has little to do with the term "biometrics" as used in the media today.

other people by face and voice, although doing so from a photograph or recording is harder. Signatures and handwriting (and handwriting recognition experts) have been used for a long time as a means of identifying people. Even finger- and palm-prints are quite old, having been recognized as unique identifiers in fourteenth century Persia [71].

Recognition via biometrics was traditionally performed by a human expert, and was considered sufficiently accurate to carry legal weight. Automated recognition has proven less reliable — recently, computer fingerprint matching based on poor scans lead to a man being mistakenly jailed for the Madrid train bombing in early 2004 [65]. Nevertheless, automated biometric authentication is now widely deployed. The US-VISIT program takes digital fingerprints and photographs of visitors from certain countries as they enter the United States [84, 35], and many countries will integrate electronic biometric information directly into passports by mid-2006 [15].

The most common biometrics currently used are face and voice recognition, finger- and palm-prints, hand geometry, iris scans, signatures (not just the shape but also the timing and pressure of strokes), dental records and, to a lesser extent, keystroke timing and walking gait [10]. Of these, iris scans and fingerprints have been the most successfully automated.

The basic structure of a biometric authentication system is the same with all these techniques. When a user enrolls in the system, a measurement is taken and some derived information (the "template") is stored in a database. To authenticate the user at a later time, a new measurement is taken and compared to the template. Depending on the exact application, it may be more or less easy to recover the original biometric measurement from the template.

There are many difficulties that come with the use of biometrics for keys or passwords. The main challenge is that biometric measurements are noisy: depending on the exact conditions and positioning of the measuring device, fingerprints, iris scans, face images can all come out very differently. There are also other, broader issues. Biometrics are difficult to change — fingers, irises and vocal cords cannot be revoked — so basing a key on only them is inherently problematic. Because fingerprints and face images, for example, can be measured without a person's consent, their use raises many fundamental privacy concerns. These concerns are particularly valid when the measurements (or templates) are stored in a centralized database which can later be used to identify people without their knowledge. See [75, 69, 34, 81] for longer discussions of these issues.

This research provides a mathematical framework for approaching the first issue: how to deal with noise in the measurement without storing biometric data "in the clear" on a server. (The broader issues remain wide open.)

A Motivating Example: Password Authentication To illustrate the use of random strings let us consider the task of password authentication. A user Alice has a password w and wants to gain access to her account. A trusted server stores some information $y = f(w)$ about the password. When Alice (or someone else) enters w', the server lets her in only if $f(w') = y$. In this simple application, we assume that

it is safe for Alice to enter the password for the verification. However, the server's long-term storage is not assumed to be secure (e.g., y is stored in a publicly readable `/etc/passwd` file in UNIX [63]). The goal, then, is to design an efficient f that is hard to invert (i.e., given y it is hard to find w' s.t. $f(w') = y$), so that no one can figure out Alice's password from y. Such functions f are called *one-way functions*.

Unfortunately, the solution above has several problems when used with passwords w available in real life. First, the definition of a one-way function assumes that w is *truly uniform*, and guarantees nothing if this is not the case. In contrast, human-generated and biometric passwords are far from uniform, although they do have some unpredictability in them. Second, Alice has to reproduce her password *exactly* each time she authenticates herself. This restriction severely limits the kinds of passwords that can be used. Indeed, a human can precisely memorize and reliably type in only relatively short passwords, which do not provide an adequate level of security. Greater levels of security are achieved by longer human-generated and biometric passwords, such as pass-phrases, answers to questionnaires, handwritten signatures, fingerprints, retina scans, voice commands, and other values selected by humans or provided by nature, possibly in combination (see [38] for a survey). However, two biometric readings are rarely identical, even though they are likely to be close; similarly, humans are unlikely to precisely remember their answers to multiple question from time to time, though such answers will likely be similar. In other words, the ability to tolerate a (limited) number of errors in the password while retaining security is crucial if we are to obtain greater security than that provided by typical user-chosen short passwords.

The password authentication problem described above is just one example of a cryptographic application where the issues of nonuniformity and error tolerance naturally come up. The same issues arise in any cryptographic application, such as encryption, digital signatures, or identification, where the secret key comes in the form of "biometric" data.

.

Two Simple Approaches Which Fail We would like to design a function to replace the one-way function f in the UNIX-style password authentication scheme. One approach that immediately comes to mind is *quantization*: rather than using w, use some point $\bar{w}$ close to w that comes from a small set of possibilities. For example, if w is a vector in $\mathbb{R}^n$, then one could imagine rounding off all the coordinates of w to the nearest integer and applying $f()$ to the resulting vector $\bar{w}$. The hope is that when Alice goes to authenticate herself, the data w' which has been measured will round off to the same vector as the original w (roughly, we hope $\bar{w'} = \bar{w}$). Unfortunately, this type of scheme will fail for most "interesting" types of data. In our example, some of the components of w are likely to lie near some half-integer, e.g. 0.5, 1.5, 2.5, etc. Even if w' is very close to w, those components stand a good chance of lying on the other side of the boundary in the perturbed vector w'. It is very unlikely then to have $\bar{w'} = \bar{w}$. This sort of phenomenon occurs with most high-dimensional data (it is due to the fact that most of the volume of a high-dimension shape lies near its boundary). Csirmaz and Katona [25] discuss these issues in a sligthly different context. We refer

the interested reader to [25] or a text on discrete geometry. Four our purposes, it suffices to observe that quantization does not really solve the problem at hand.

A second common approach to this problem is to change the model and use encrypted data: instead of storing $f(w)$, one could just store an encryption of w, and then store the corresponding key on a different server. The problem here is that the entity verifying Alice's identity must know the key, and if that entity's storage is compromised then Alice's data is available in the clear. The goal is to allow the verification information to be distributed easily, as in the UNIX-style storage of $f(w)$.

1.1.1 Contributions of This Research

Assumptions Our techniques apply to any key material with the following properties:

1. Any two measurements of the same input (say, a particular person's fingerprint) are "close" to each other. The meaning of "close" depends on the application, but we assume that there is some metric on possible results, and that two measurements will be close in that metric.

2. The distribution of measurement outcomes is sufficiently random. Specifically, we assume that an adversary's a priori probability of guessing the input is not too high.

   The users and designers of the system do not need to know the distribution itself, but only a lower bound on its entropy.

The New Definitions, Informally We propose two primitives, termed *secure sketch* and *fuzzy extractor*.

A secure sketch addresses the problem of error tolerance. It is a (probabilistic) function outputting a public value v about its biometric input w, that, while revealing little about w, allows its exact reconstruction from any other input w' that is sufficiently close. The price for this error tolerance is that the application will have to work with a lower level of entropy of the input, since publishing v effectively reduces the entropy of w. However, in a good secure sketch, this reduction will be small, and w will still have enough entropy to be useful, even if the adversary knows v. A secure sketch, however, does not address nonuniformity of inputs.

A fuzzy extractor addresses both error tolerance and nonuniformity. It reliably extracts a uniformly random string R from its biometric input w in an error-tolerant way. If the input changes slightly, the extracted R remains the same. To assist in recovering R from w', a fuzzy extractor outputs a public string P (much like a secure sketch outputs v to assist in recovering w). However, R remains uniformly random even given P.

Our approach is general: our primitives can be naturally combined with *any* cryptographic system. Indeed, the string R extracted from w by a fuzzy extractor can be used as a key in any cryptographic application but, unlike a traditional key, need not be stored (because it can be recovered from any w' that is close to w). We define

our primitives to be *information-theoretically* secure, thus allowing them to be used in combination with any cryptographic system without additional assumptions (however, the cryptographic application itself will typically have computational, rather than information-theoretic, security).

For a concrete example of how to use fuzzy extractors, in the password authentication case, the server can store $\langle P, f(R)\rangle$. When the user inputs w' close to w, the server recovers the actual R and checks if $f(R)$ matches what it stores. Similarly, R can be used for symmetric encryption, for generating a public-secret key pair, or any other application. Secure sketches and extractors can thus be viewed as providing fuzzy key storage: they allow recovery of the secret key (w or R) from a faulty reading w' of the password w, by using some public information (v or P). In particular, fuzzy extractors can be viewed as error- and nonuniformity-tolerant secret key *key-encapsulation mechanisms* [76].

Because different biometric information has different error patterns, we do not assume any particular notion of closeness between w' and w. As mentioned above, the definitions simply assume that w comes from some metric space, and that w' is no more that a certain distance from w in that space. We only consider particular metrics when building concrete constructions.

General Results Before proceeding to construct our primitives for concrete metrics, we describe several general observations:

- Fuzzy extractors can be built out of secure sketches using strong randomness extractors [66], such as pairwise-independent hash functions.
- The existence of secure sketches and fuzzy extractors over a particular metric space implies the existence of certain error-correcting codes in that space, thus producing lower bounds on the best parameters a secure fingerprint and fuzzy extractor can achieve.
- For a special class of (finite) metric spaces, called *transitive*, one can construct a secure sketching scheme from any error-correcting code (e.g. the construction for Hamming distance, below).

  Transitive finite metric spaces include the Hamming cube, set difference, and transposition distance on orderings of a list.
- For a metric space $\mathcal{M}'$ without nice properties such as transitivity, one can nonetheless construct fuzzy extractors from certain "metric embeddings"—these are maps from one metric space to another which approximately preserve distances. Specifically, the existence of a fuzzy extractor in the target space, combined with a biometric embedding of the source into the target, implies the existence of a fuzzy extractor in the source space.

  We need weaker properties from the embedding than those normally required in algorithmic applications. We call maps which satisfy the relaxed definition *biometric embeddings*.

These general results help us in building and analyzing our constructions.

Constructions We describe constructions of secure sketches and extractors in three metrics: Hamming distance, set difference, and edit distance.

Hamming distance (i.e., the number of bit positions that differ between w and w') is perhaps the most natural metric to consider. We observe that the "fuzzy-commitment" construction of Juels and Wattenberg [47] based on error-correcting codes can be viewed as a (nearly optimal) secure sketch. We then apply our general result to convert it into a nearly optimal fuzzy extractor. While our results on the Hamming distance essentially use previously known constructions, they serve as an important stepping stone for the rest of the work.

The set difference metric (i.e., size of the symmetric difference of two input sets w and w') comes up naturally whenever the biometric input is represented as a subset from a universe of possible features.Using the transitivity of the metric space, we demonstrate the existence of optimal (with respect to entropy loss) secure sketches, and therefore also fuzzy extractors, for this metric. However, this result is mainly of theoretical interest, because (1) it relies on optimal constant-weight codes, which we do not know how construct and (2) it produces sketches of length proportional to the universe size. We then turn our attention to more efficient constructions for this metric, and provide two of them.

First, we observe that the "fuzzy vault" construction of Juels and Sudan [48] can be viewed as a secure sketch in this metric (and then converted to a fuzzy extractor using our general result). We provide a new, simpler analysis for this construction, which bounds the entropy lost from w given v. Our bound on the loss is quite high unless one makes the size of the output v very large. We then provide an improvement to the Juels-Sudan construction to reduce the entropy loss to near optimal, while keeping v short (essentially the same length as w).

Second, we note that in the case of a small universe, a set can be simply encoded as its characteristic vector (1 if an element is in the set, 0 if it is not), and set difference becomes Hamming distance. However, the length of such a vector becomes unmanageable as the universe size grows. Nonetheless, we demonstrate that this approach can be made to work efficiently even for exponentially large universes. This involves a result that may be of independent interest: we show that BCH codes can be decoded in time polynomial in the *weight* of the received corrupted word (i.e., in *sublinear* time if the weight is small). The resulting secure sketch scheme compares favorably to the modified Juels-Sudan construction: it has the same near-optimal entropy loss, while the public output v is even shorter (proportional to the number of errors tolerated, rather than the input length).

Finally, edit distance, that is, the number of insertions and deletions needed to convert one string into the other, naturally comes up, for example, when the password is entered as a string, due to typing errors or mistakes made in handwriting recognition. We construct a biometric embedding from the edit metric into the set difference metric, and then apply our general result to show that such an embedding yields a fuzzy extractor for edit distance, because we already have fuzzy extractors for set difference. The edit metric is quite difficult to work with, and the existence of such an embedding is not a priori obvious: for example, low-distortion embeddings of the edit distance into the Hamming distance are unknown and seem hard to con-

struct [4]. It is the particular properties of biometric embeddings, as we define them, that make the construction feasible.

Lower Bounds As mentioned above, we also show that the existence of secure sketches or fuzzy extractors for a particular metric implies the existence of error-correcting codes for that metric with related parameters. Thus we can use existing bounds on the parameters achievable for codes to get bounds on what parameters are achievable for our new primitives. The end result is a set of lower bounds on how much information must be leaked about the input to allow error-correction. We prove two kinds of bounds. The first kind of bound, on the *min-entropy loss*, tells us limits on the length of a key which can be extracted from the input given that the adversary has learned the public output P (resp. secure sketch). The second kind of bound, on the *loss of Shannon entropy* gives a more intuitively significant result: the *mutual information* between the input and the public information must be high.

1.1.2 Related Work

Relation to Previous Work Since this topic combines elements of error correction, randomness extraction and password authentication, there is a lot of related work.

The need to deal with nonuniform and low-entropy passwords has long been realized in the security community, and many approaches have been proposed. For example, Kelsey et al [49] suggest using $f(w, r)$ in place of w for the password authentication scenario, where r is a public random "salt," to make a brute-force attacker's life harder. While practically useful, this approach does not add any entropy to the password, and does not formally address the needed properties of f. Another approach, more closely related to ours, is to add biometric features to the password. For example, Ellison et al. [36] propose asking the user a series of n personalized questions, and use these answers to encrypt the "actual" truly random secret R. A similar approach using user's keyboard dynamics (and, subsequently, voice [60, 61]) was proposed by Monrose et al [62]. Of course, this technique reduces the question to that of designing a secure "fuzzy encryption". While heuristic approaches were suggested in the above works (using various forms of Shamir's secret sharing), no formal analysis was given. Additionally, error tolerance was addressed only by brute force search.

Soutar et al.[78, 79, 77] took a different approach, based on techniques from signal processing. Their technique is the basis for a commercial implementation (Bioscrypt, Inc) but we don't know of a rigorous security analysis of their scheme.

A more formal approach to error tolerance in biometrics was taken by Juels and Wattenberg [47] (for less formal solutions, see [26, 27, 62, 36]), who provided a simple way to tolerate errors in *uniformly distributed* passwords. Frykholm and Juels [39] extended this solution; our analysis is quite similar to theirs in the Hamming distance case. Almost the same construction appeared implicitly in earlier, seemingly unrelated, literature on information reconciliation and privacy amplification (see, e.g.,

[7, 8, 24]). We discuss the connections between these works and our work further in Section 3.4.

Juels and Sudan [48] provided the first construction for a metric other than Hamming: they construct a "fuzzy vault" scheme for the set difference metric. The main difference is that [48] lacks a cryptographically strong definition of the object constructed. In particular, their construction leaks a significant amount of information about their analog of R, even though it leaves the adversary with provably "many valid choices" for R. In retrospect, their notion can be viewed as an (information-theoretically) one-way function, rather than a semantically-secure key encapsulation mechanism, like the one considered in this work. Nonetheless, their informal notion is very closely related to our secure sketches, and we improve their construction in Section 3.5. Clancy et al. [21] used the Juels-Sudan scheme as the basis of a smartcard-based biometric authentication system. They provide an analysis of the scheme's security against some specific attacks, one based on brute force search and the other on Berlekamp-Welch decoding. Our improvements to the Juels-Sudan scheme extend more or less directly to [21].

The work of Linnartz and Tuyls [52] is the closest to ours in approach. They define and construct a primitive called a "shielding function", similar to a fuzzy extractor (that line of work was continued in [87].) In two works of similar flavor, Cohen and Zémor [22] and Tuyls and Goseling [82] derive specific channel-capacity bounds on biometric authentication systems by assuming particular distributions on both inputs and error patterns in the biometric inputs. The main difference between those works and ours is that they assume that the exact distribution on the biometric inputs is simple and known to the designer (either the uniform distribution over strings of a given length or a known, multivariate Gaussian in $\mathbb{R}^n$). Another difference is that [52, 87] focus on the continuous space $\mathbb{R}^n$, whereas we focus on discrete metric spaces. We learned of these works after having completed the research described in this chapter.

Other, very different approaches have also been taken for guaranteeing the privacy of noisy data. Csirmaz and Katona [25] consider quantization for correcting errors in "physical random functions," essentially proving that in most metric spaces this approach is difficult, if not impossible, to use. (In the language of our paper, quantization corresponds to secure sketches with no public storage). Barral, Coron and Naccache [5] proposed a system for off-line, private comparison of fingerprints. Ratha, Connell and Bolle [68] proposed "cancelable" biometrics, the idea of using distorted versions of the same biometric on different servers, to prevent the leaking of one version from making the other versions known to an attacker. Although seemingly similar, the approaches of [5, 68] are complimentary to ours, and the solutions can be combined to yield systems which enjoy the benefits of all of them.

A nice summary of much of the previous work on biometric cryptosystems, along with an experimental evaluation of the technique of Juels and Sudan [48] appears in a survey by Uludag et al [83]. Our improvement to the Juels-Sudan scheme extends directly to the results of [83].

Finally, work on privacy amplification [7, 8], as well as work on de-randomization and hardness amplification [44, 66], also addressed the need to extract uniform ran-

domness from a random variable about which some information has been leaked (see Chap. 2). A major focus of research in that literature has been the development of (ordinary, not fuzzy) *randomness extractors* with short seeds (see Section 1 for an informal definition). We use randomness extractors, though for our purposes, pairwise independent hashing [7, 44] is sufficient.

Subsequent Work There have been several works subsequent to ours which built on these ideas. Van Dyjk and Woodruff [28] worked on improving the entropy loss of secure sketches using computational assumptions. Burnett, Duffy and Dowling [16] used fuzzy extractors as the basis of an implementation, in Java, of an identity-based signature scheme. Boyen [12] extended the definition of fuzzy extractors to allow reusability of the same biometric for generating many different keys. The results hold in a model where the errors which occur are independent of the biometric data itself. The existence of re-usable fuzzy extractors without the independence assumption remains an interesting open question.

On a different tack, our work has been applied to privacy amplification: Fuzzy extractors may be seen as noise-resilient versions of randomness extractors, and Ding [29] used this idea for noise tolerance in Maurer's bounded storage model [57].

1.2 Entropic Security: Hiding All Partial Information

The second part of the thesis considers a more general setting, in which information must be leaked, yet we want to ensure—in some information-theoretic sense—that the leaked information is not useful. We describe general conditions which guarantee that leaked information cannot be used to help predict any function of the input, and use the insights gained to find new constructions of various cryptographic primitives with this property.

Let $Y()$ be a probabilistic map, and X a random variable distributed over strings of bits. Normally, one formalizes the statement "$Y(X)$ leaks no information about X" by requiring that X and $Y(X)$ be very close to statistically independent random variables. Equivalently, one can require that the *Shannon mutual information*[2] $\mathbf{I}(X; Y(X))$ be very small.

However, we'll consider situations where information leakage is unavoidable—that is $\mathbf{I}(X; Y)$ is quite large, and X and Y are very far from being independent. For example, if X is an iris scan and Y is a secure sketch of X which can correct τ flipped bits (see the previous section), then by Proposition 4.3 there are necessarily at least τ bits of mutual information between X and Y (i.e. $\mathbf{I}(X; Y) \geq \tau$).

[2]The mutual information between two random variables measures how far they are from being completely independent. The exact definition is not important for this introduction; it is given in Section 2.1.

Entropic Security Since the usual notion of secrecy is unattainable, we propose an alternative, inspired by semantic security of encryptions [40]: For every function f, it should be hard to predict $f(X)$ given Y (that is, learning Y should not change the adversary's probability of guessing $f(X)$ by more than some small parameter ϵ). A map $Y()$ that satisfies this condition is said to *hide all functions of* X. Pictorially, the direction of the diagonal arrow should be hard to compute:

$$\begin{array}{ccc} X & \longrightarrow & Y(X) \\ \downarrow & \swarrow & \\ f(X) & & \end{array}$$

More precisely, the information leakage is at most ϵ if for all functions, seeing $Y(X)$ increases the probability of guessing $f(X)$ by at most ϵ. The map $Y()$ is entropically secure if the condition holds for all input distributions X of sufficiently high min-entropy[3]

Definition 1.1 (Entropic Security). *The probabilistic map Y hides all functions of X with leakage ϵ if for every adversary $\mathcal{A}$, there exists some adversary $\mathcal{A}'$ such that for all functions f,*

$$\left| \Pr[\mathcal{A}(Y(X)) = f(X)] - \Pr[\mathcal{A}'() = f(X)] \right| \leq \epsilon.$$

The map $Y()$ is called (t, ϵ)-entropically secure if $Y()$ hides all functions of X, whenever the min-entropy of X is at least t.

This notion of security was introduced by Canetti, Micciancio and Reingold [17, 18], and subsequently and independently by Russell and Wang [70]. Both of those works discussed a weaker-seeming version of this definition (one of our results is the near-equivalence of their notion of security to the one described here—see below).

1.2.1 Two Games for Measuring Information Leakage

In order to explain the relation between entropic security and the standard notion of security, we formulate two abstract games. Both games attempt to capture the meaning of the statement "$Y(X)$ leaks no information about X" by allowing an adversary to guess the value of a function $f(X)$ based on $Y = Y(X)$.

In this discussion, the pairs (X, Y) and (X', Y') are sampled independently according to the same joint distribution (i.e. $Y = Y(X)$ and $Y' = Y(X')$). Let $\mathcal{X}$ denote the range of X.

Game 1, on input y: The adversary receives y as input and outputs the description of a function $f_y : \mathcal{X} \to \{0,1\}^*$, and a string g. The adversary's gain is:

$$\Pr[f_y(X) = g \mid Y = y] - \Pr[f_y(X') = g]$$

[3]The min-entropy of a random variable is one of several measures of the inherent uncertainty about the value of the variable. The min-entropy of X is the negative logarithm of the probability of guessing the value of X ahead of time, that is $\mathbf{H}_\infty(X) = -log(\max_x \Pr[X = x])$.

Game 2, on input y: The adversary produces the description of a function $f : \mathcal{X} \to \{0,1\}^*$ before seeing y. The adversary then sees y and outputs a string g. The adversary's gain is:

$$\Pr[f(X) = g \mid Y = y] - \Pr[f(X') = g]$$

Now consider the adversary's expected gain in each of these games when the input Y is chosen at random. We'll say that Y leaks *a-posteriori information about* X if there exists an adversary who has a non-negligible expected advantage in Game 1, since the adversary gets to decide what he wants to learn *after* seeing Y. Similarly, we'll say that Y leaks *a priori information about* X if there's an adversary who has a non-negligible advantage in Game 2—there, the adversary must decide ahead of time what information about X is of interest.

The adversary's expected advantage in the *a-posteriori* game is the standard measure of information leakage, and is well understood. In particular, it can be bounded by the statistical difference[4] between the joint distribution (X, Y) and the product of marginals (X', Y) (where X' is independent of Y). One can also ensure that no a-posteriori information is leaked by requiring that $\mathbf{I}(X;Y) \leq \epsilon$, where ϵ is some "negligible" quantity (and $\mathbf{I}$ is the mutual information). On the other hand, the adversary's advantage in Game 2 is less well understood. One gets some insight by thinking of it as a simplified, information-theoretic reformulation of semantic security of encryptions [40].

1.2.2 Contributions on Entropic Security

This thesis considers situations in which we simply cannot prevent an adversary from having a large advantage in Game 1—that is, we cannot prevent non-negligible Shannon information about X from being leaked by Y—and yet we can still satisfy a strong definition of secrecy by ensuring that no *a-priori* information is leaked, i.e. no advantage is possible in Game 2. We provide a general technique for proving that Game 2 cannot be won—namely, it is necessary and sufficient to show that the map $Y(\cdot)$ is some kind of *randomness extractor*. We then construct special-purpose extractors to suit three different applications.

As with entropy loss for secure sketches, we bound the leakage ϵ by assuming that X itself is hard to predict. We show that in many situations, one can construct maps $Y()$ which hide all functions of X as long as $\mathbf{H}_\infty(X) \geq I(X;Y) + 2\log\left(\frac{1}{\epsilon}\right)$. The quantity $\log\left(\frac{1}{\epsilon}\right)$ is the number of "bits of security," a standard measure in cryptography.

A Strong Definition of Security The definition we propose (Definition 1.1) is stronger than previously studied formulations of entropic security [17, 18, 70], which only considered the adversary's ability to predict *predicates* instead of all possible

[4]The statistical difference, or total variation distance, between two probability distributions measures how easy it is to distinguish samples from one distribution from samples from the other. See Section 2.1.

functions of the secret input. (Recall that a predicate is a "yes"/"no" question, that is, a function that outputs a single bit.)

For example, the definition in terms of predicates does *not directly imply* that the adversary's chance of recovering the message itself remains low! The implication does in fact hold, but requires some proof. If we assume that the adversary can guess the input with non-trivial probability, we can obtain a contradiction by choosing a "Goldreich-Levin" predicate at random, that is by choosing $g_r(x) = r \odot x$ where r is a random n-bit string and $\odot$ is the binary inner product $r \odot x = \sum_i r_i x_i \mod 2$. We omit a detailed proof; we prove much more general implications below, for which the GL predicates do not suffice.

An Equivalence to Indistinguishability The key result behind all our constructions is the equivalence of the following statements:

- The map $Y()$ hides all functions of X, as long as $\mathbf{H}_\infty(X) \geq t$.
- For any two random variables X_1, X_2 which both have min-entropy at least $t-2$, the random variables $Y(X_1)$ and $Y(X_2)$ are statistically indistinguishable.

There are two main pieces to the result. First, we show that indistinguishability is equivalent to entropic security for predicates (the definition of [17, 70]). This is the easier of the two parts. Second, we show that if the adversary can gain advantage ϵ at predicting some function $f(X)$ given Y, then there exists a predicate g, which depends on f and the distribution of X such that the adversary gets nearly the same advantage at guessing $g(X)$ given Y. This last result may be of independent interest. It is an information-theoretic converse to the Goldreich-Levin hardcore bit construction, which states converts a good predictor for a particular predicate into a good predictor for some underlying function.

This equivalence provides a new application of *randomness extractors* to cryptography; namely, we show that an extractor's output reveals very little about its source. [5] (Recall that an extractor takes as input an arbitrary, high entropy random source and outputs a small number of uniformly random bits.) The equivalence simplifies many existing proofs of security and also strengthens them—existing techniques only proved that no *predicate* of the secret X was leaked (as opposed to no function of any kind).

Finally, the result parallels—and was inspired by—a similar equivalence due to Goldwasser and Micali for the case of computationally secure encryption schemes [40]—see below.

We also obtain new constructions, lower bounds and tighter analyses for the following settings:

[5] This use of extractors has been around implicitly in complexity theory for many years, for example in the use of hash functions for approximate counting. However, our abstraction, and cryptographic perspective, are novel.

Symmetric encryption with a short key Suppose one wants to encrypt a n-bit message using k bits of secret key. Shannon's famous lower bound [73] states that without computational assumptions, one cannot encrypt securely using fewer than n bits of key. Russell and Wang [70] showed that one can nonetheless hide all predicates of the message with a much shorter key. We provide several new results:

- A stronger definition of security, based on the equivalence between entropic security for all functions and indistinguishability of encryptions for message spaces with high min-entropy.

- Two general frameworks for constructing entropically secure encryption schemes, one based on expander graphs and the other on XOR-universal hash functions. These schemes generalize the schemes of Russell and Wang, yielding simpler constructions and proofs as well as improved parameters.

- Lower bounds on the key length k for entropic security and indistinguishability. In particular, we show near tightness of Russell-Wang constructions: $k > n - t$. (In fact, for a large class of schemes $k \geq n - t + \log\left(\frac{1}{\epsilon}\right)$.)

Noise-resilient Perfectly One-Way Hash Functions Canetti, Micciancio and Reingold [17, 18] showed that it is possible to construct randomized hash functions $Y()$ such that $Y(X)$ hides all predicates of X, but nonetheless $Y()$ is *verifiable*: given x and y, one can check whether or not $Y(x) = y$, yet it is not feasible to find x_1, x_2 such that $Y(x_1) = Y(x_2)$.

- We show how to construct "fuzzy"—that is, noise-resilient—perfect hash functions. The hash value for w allows one to verify whether a candidate string w' is *close* to w, but reveals nothing else about w. The main technical tools are constructions of entropically-secure sketches and fuzzy extractors.

  This result is a significant departure from the approach of Canetti *et al.* The motivation behind [17, 18] was to formalize the properties of an ideal "random oracle" which might be achievable by a real computer program. In contrast, even given a random oracle, it is not at all clear how to construct a proximity oracle for a particular value w (i.e. an oracle that accepts an input if and only if it is sufficiently close to w). In that sense, the result is also about *code obfuscation:* noise-resilient POWFs might best be viewed as obfuscated versions of a proximity oracle.

- We strengthen the results of [18] on information-theoretically-secure POWF's. Following the general results on entropic securiy, we strengthen the definition of perfect one-way-ness to preclude the adversary from improving her ability to predict any *function* of the input. We also improve the analysis of the [18] construction, obtaining better parameters (roughly half the entropy loss) and reducing the assumptions necessary for security.

1.2.3 Composability and Semantic Security

Composing Entropically-Secure Constructions—Towards Robust Security A desirable property of definitions of security of cryptographic primitives is *composability*: once some protocol or algorithm has been proven secure, you would like to be able to use it as a building block in other protocols with your eyes closed—without having to worry about effects that violate the intuitive notion of security, but which are not covered by the original definition.

Composability is difficult to guarantee, since it is not clear how to translate it into a mathemetical property. There are various formalizations of composability, most notably "Universal Composability" [19] and several frameworks based on logic algebras for automated reasoning (see [45] and the references therein). Finding protocols that are provably secure in these general frameworks is difficult, and sometimes provably impossible. A more common approach is to prove that a particular definition remains intact under a few straightforward types of composition, say by proving that it is still secure to encrypt the same message many times over.

The main weakness of entropic security, as defined above, is that it does not ensure composability, even in this straightforward sense. If $Y()$ and $Y'()$ are independent versions of the same entropically-secure mapping, then the map which outputs the pair $Y(X), Y'(X)$ may be completely insecure, to the point of revealing X completely. In the case of encryption, this may mean that encrypting the same message twice is problematic. The reason is that given the first value $Y(X)$, the entropy of X may be very low, too low for the security guarantee of $Y'()$ to hold.

For example, suppose that $Y(x)$ consists of the pair M, Mx, where M is a random $\frac{3n}{4} \times n$ binary matrix M and $x \in \{0,1\}^n$. We will see in later chapters that $Y()$ is entropically secure whenever the entropy of X is close to n. However, the pair $Y(x), Y'(x)$ provides a set of $\frac{3n}{2}$ randomly chosen linear constraints on x. With high probability, these determine x completely, and so the pair $Y(), Y'()$ is insecure under any reasonable definition.

Given these issues, entropically-secure primitives must be used with care: one must ensure that the inputs truly have enough entropy for the security guarantee to hold. When the inputs are passwords, the requirement of high entropy is natural, but the issue of composability raises a number of intersting open questions for future work.

On Semantic Security and Indistinguishability In a seminal paper, Goldwasser and Micali [40] introduced two notions of security for encryption schemes against computationally bounded eavesdroppers. They showed that seeing an encryption of a message gives a computationally bounded eavesdropper no useful information (roughly in the sense defined above) *if and only if* there is no pair of messages m_1, m_2 such that the eavesdropper can distinguish the encryption of m_1 from the encryption of m_2.

Our result can be seen as a parallel of that equivalence for high-entropy distributions. The Goldwasser-Micali equivalence can be stated as follows: if security holds for all flat distributions with 1-bit of entropy — that is, distributions over pairs

(m_1, m_2) — then no information is leaked for any distribution whatsoever. Our result provides a limited generalization: if you can prove that encryptions of messages from all distributions of min-entropy at least t are indistinguishable, then no information is leaked to the eavesdropper when the message has min-entropy at least $t+1$. There are significant differences: the original equivalence of [40] holds for both computationally bounded adversaries and unbounded ones, whereas our equivalence only holds for computationally unbounded adversaries. The proof techniques are also very different.

1.3 Organization of This Thesis

The thesis is divided into two parts: the first part describes the new definitions and constructions for the secure use of biometric passwords; the second, longer part presents the results on entropic security.

Chapter 2 presents the background definitions material and notation.

Chapter 3 deals with the secure use of biometric passwords. We define secure sketches and extractors, which give us an abstract framework for dealing with noisy data, and present the constructions for three measures of distance: Hamming distance, set difference and edit distance. We discuss improvements to the error-tolerance which are possible using "list-decoding" of codes, and conclude the chapter with the details applying our framework to the problem of password authentication.

In Chapter 4, we show a further relationship between our new primitives and error-correcting codes. We use this to establish various lower bounds on the information that must be leaked by our primitives, both in terms of min-entropy and Shannon entropy.

Chapter 5 presents our general results on entropic security. The main result of this section is the equivalence of entropic security to indistinguishability of encryptions of high-entropy message distributions.

Chapter 6 applies these general results to entropically-secure encryption schemes. We describe two new encryption schemes, each with a simple proof of security, and give lower bounds which show that the key length of our schemes is nearly optimal.

Chapter 7 describes our results on entropically-secure hash functions, called "perfectly one-way" hash functions (POWHF). First, we construct entropically-secure fuzzy extractor and secure sketches for Hamming distance, and use those to build "fuzzy" (i.e. noise-resilient) POWHF. The new construction also simplifies and improves the known constructions of ordinary (not noise-resilient) POWHF.

Finally, Appendix A gives the proofs of several lemmas on universal hash functions (variants on the "left-over hash lemma", or "privacy amplification lemma"). The proofs are collected together for easy reference, and since they are all quite similar.

Chapter 2

Mathematical Preliminaries

2.1 Probability Distributions and Entropy

Notation Unless explicitly stated otherwise, logarithms, denoted $\log(x)$ are base 2. When defining a new quantity x, we will often use $x \stackrel{def}{=} x'$. The set of binary strings lf length n is denoted $\{0,1\}^n$. Bitwise XOR of strings is denoted by $\oplus$, and inner product mod 2, that is the standard inner product on $\{0,1\}^n$, is denoted with the symbol $\odot$.

Random variables will typically take on values either in $\mathbb{R}$ or in a finite, discrete space such as $\{0,1\}^n$, and are denoted with capital letters in bith cases. The probability that a random variable X takes on a particular x. $\mathbb{E}[X]$ denotes the expectation of a real-valued random variables and $\mathsf{Var}[X]$, its variance. Conditional probabilites, exectations and variance will be denoted in the standard way: conditioned on event E, we write $\Pr[X = x|E], \mathbb{E}[X|E], \mathsf{Var}[X|E]$. In many cases, we will use the same symbol both for a random variable and the distribution according to which it is drawn.

If X is a probability distribution (or random variable), the notation $x \leftarrow X$ means "draw x according to X." We will sometimes use $x \sim X$ to denote the same thing. If S is a finite set, we write $x \leftarrow S$ to denote "draw S from the uniform distribution of S." Since it comes up very often, we use U_ℓ to a random variable distributed uniformly on ℓ-bit binary strings, i.e. the set $\{0,1\}^\ell$. The same notation will also be used for more complex experiminets: $x \leftarrow A(y)$ means that the (possibly probabilistic) algorithm A is run on input y and the output is assigned to x. If A is probabilistic and we wish to make the random bits R explicit, we write $x \leftarrow A(y;R)$.

We will often want to discuss the asymptotics of various constructions, in which case the construction will be parametrized by an integer (typically an input length). If O_n refers to the instantiation of some construction with parameter n, we call a sequence $O_{n_0}, O_{n_0+1}, O_{n_o+2}, ...$ an *ensemble*, and abuse notation slightly by writing $\{O_n\}_{n\in\mathbb{N}}$.

A non-negative function $\epsilon : \mathbb{N} \to \mathbb{R}$ is called negligible if $\epsilon(n) = O(1/n^c)$ for every constant $c \in \mathbb{N}$. The input n to ϵ is frequently omitted.

Distance Measures on Probability Distributions The *statistical distance* (sometimes called *total variation distance*, or simply L_1 distance) between two probability distributions A and B on the same space is

$$\mathbf{SD}(A, B) \stackrel{def}{=} \frac{1}{2} \sum_v \left| \Pr[A = v] - \Pr[B = v] \right|$$

(that is, half the L_1 distance between the probability mass functions). Statistical difference measures an adversary's probability of telling samples from two distributions apart: if one flips a fair coin and samples from either A or B according to the coin, then an adversary's best possible probability of guessing the value of the coin given the sample is exactly $\frac{1}{2} + \frac{1}{2}\mathbf{SD}(A, B)$.

The L_2 *distance* between two distributions A and B is

$$\|A - B\|_2 \stackrel{def}{=} \sqrt{\sum_v (\Pr[A = v] - \Pr[B = v])^2}.$$

This does not have a natural interpretation in the way that statistical difference does, but it is often easier to compute. We will use:

Fact 2.1. *If A, B are distributions on $\{0,1\}^n$ with $\|A - B\|_2^2 \leq \epsilon^2/2^n$, then the statistical difference $\mathbf{SD}(A, B)$ is at most ϵ.*

2.1.1 Three Measures of Entropy

Informally, the *entropy* of a probability distribution is the amount of inherent uncertainty in the outcome of drawing a value from the distribution. There are various ways to define entropy, and typically each formulation is useful in different contexts. In the thesis, we will work with three formulations of entropy. In a seminal work, Shannon introduced the concept of entropy to probability theory and signal processing [73]. The notion of entropy he defined is called *Shannon entropy*, and is probably the most commonly used definition. We will mostly make use of two different notions, which turn out to be more convenient when one works with randomness extraction from an unknown source: *min-entropy* and *collision entropy*. We will define these two first.

Min-Entropy The *min-entropy* $\mathbf{H}_\infty(A)$ of a random variable A is the logarithm of the most probable element in the distribution:

$$\mathbf{H}_\infty(A) \stackrel{def}{=} -\log(\max_a \Pr(A = a)).$$

This measures how hard it is to predict the outcome of a draw from A: the probability of successfully guessing the outcome ahead of time is exactly the mass of the most likely element. This is cryptographically useful: it is the probability that an adversary can guess the value of A on the first try.

For a pair of (possibly correlated) random variables A, B, a conventional notion of "average min-entropy" of A given B would be $\mathbb{E}_{b \leftarrow B}\left[\mathbf{H}_\infty(A \mid B=b)\right]$. However, for our purposes, the following slightly modified notion will be more robust: we let

$$\tilde{\mathbf{H}}_\infty(A \mid B) = -\log\left(\mathbb{E}_{b \leftarrow B}\left[2^{-\mathbf{H}_\infty(A|B=b)}\right]\right).$$

Namely, we define *average min-entropy* of A given B to be the logarithm of the average probability of the most likely value of A given B. This definition is the right one to use when one is interested in the statistical difference from uniform, as becomes clear, for example, in Lemma 3.2.

The following simple lemma explains why our choice is more convenient. The lemma is not new, but we do not know of a convenient reference and so we provide a proof here.

Lemma 2.2. *For any random variables A, B, and for any $\delta > 0$, the conditional entropy $\mathbf{H}_\infty(A|B=b)$ is at least $\tilde{\mathbf{H}}_\infty(A|B) - \log(1/\delta)$ with probability at least $1-\delta$ (the probability here is taken over the choice of b).*

Moreover, if B is an ℓ-bit string, then $\tilde{\mathbf{H}}_\infty(A \mid B) \geq \mathbf{H}_\infty(A) - \ell$.

Proof. Let $p = 2^{-\tilde{\mathbf{H}}_\infty(A|B)} = \mathbb{E}_b\left[2^{-\mathbf{H}_\infty(A|B=b)}\right]$. By the Markov inequality, $2^{-\mathbf{H}_\infty(A|B=b)} \leq p/\delta$ with probability at least $1-\delta$. Taking logarithms, we get the first claim. Next, let $q = 2^{-\mathbf{H}_\infty(A)}$, so that q is the maximum probability in A's distribution. Conditioned on any event E of some probability $\alpha > 0$, this max probability can go up by at most $1/\alpha$, since $q \geq \Pr[A=a] \geq \Pr[(A=a) \wedge E] = \alpha \cdot \Pr[A=a \mid E]$. Thus, we have:

$$\mathbb{E}_{b \leftarrow B}\left[2^{-\mathbf{H}_\infty(A|B=b)}\right] \leq \sum_b \Pr[B=b]\left(\frac{q}{\Pr[B=b]}\right)$$
$$= q \cdot \#\left\{b \mid \Pr[B=b] > 0\right\} \leq q \cdot 2^\ell.$$

But this means that $\tilde{\mathbf{H}}_\infty(A \mid B) \geq -\log(q \cdot 2^\ell) = \mathbf{H}_\infty(A) - \ell$. □

Collision Entropy The *collision probability* of a distribution A is the probability that two independent draws from the same distribution will yield the same value:

$$\mathsf{Col}(A) = \Pr_{a,b \leftarrow A}[a=b] = \sum_a \Pr[A=a]^2 = \|A\|_2^2.$$

It is well known [46] that if A has support on some set $\mathcal{A}$ and $\mathsf{Col}(A) \leq (1+\epsilon^2)/|\mathcal{A}|$, then A is ϵ-close to the uniform distribution $U_\mathcal{A}$ on $\mathcal{A}$: $\mathbf{SD}\left(A, U_\mathcal{A}\right) \leq \epsilon$. (This is a direct consequence of Fact 2.1.)

The *collision entropy* (sometimes called *Renyi entropy*) of A, is defined as the logarithm of the collision probability:

$$\mathbf{H}_2(A) \stackrel{def}{=} -\log_2 \mathsf{Col}(A)$$

Collision entropy is very closely related to the min-entropy:

$$\mathbf{H}_\infty(A) \leq \mathbf{H}_2(A) \leq 2\mathbf{H}_\infty(A).$$

By another useful lemma for which I have no good reference, any distribution with collision entropy t is within statistical difference ϵ of a distribution with min-entropy at least $t - \log(1/\epsilon)$ (proof omitted).

Shannon Entropy The *Shannon entropy* of a distribution is given by:

$$\mathbf{H}_{sh}(A) \stackrel{def}{=} \sum_a \Pr[A = a] \log \frac{1}{\Pr[A = a]}.$$

Similarly, the *conditional entropy of A given B* is $\mathbf{H}_{sh}(A \mid B) \stackrel{def}{=} \mathbb{E}_b \left[\mathbf{H}_{sh}(A \mid B = b)\right]$. Alternatively, it can computed as the difference between the entropy of the pair A, B and the entropy of B alone: $\mathbf{H}_{sh}(A \mid B) = \mathbf{H}_{sh}(A, B) - \mathbf{H}_{sh}(B)$.

Finally, the *mutual information* between A and B measures the drop in the entropy of A which is caused by learning B:

$$\mathbf{I}(A; B) \stackrel{def}{=} \mathbf{H}_{sh}(A) - \mathbf{H}_{sh}(A \mid B) = \mathbf{H}_{sh}(A, B) - \mathbf{H}_{sh}(A) - \mathbf{H}_{sh}(B).$$

The mutual information is thought of as measuring the amount of information about A which is revealed by B. The formula is symmetric, so it also represents the information about B which is revealed by A.

The Shannon information can differ a lot from collision and min-entropy. For example, consider a distribution A which takes a particular value with probability $\frac{9}{10}$, and is uniform over all strings of length n with the remaining probability $\frac{1}{10}$. Then in fact the Shannon entropy of A is quite large, about $n/10$. On the other hand, the min-entropy is only $\log(10/9) \approx 0.15$.

There are, however, several useful relationships of Shannon entropy with other measures. First, it is always an upper bound:

$$\mathbf{H}_{sh}(A) \geq \mathbf{H}_2(A) \geq \mathbf{H}_\infty(A).$$

Second, when the mutual information between two random variables X, Y is very small, then they are very close to uniform, and vice versa. We will only need the reverse implication in this thesis:

Lemma 2.3 (Theorem 16.3.2 in [23]). *If A, B are distributions over $\{0,1\}^n$, and $\mathbf{SD}(A, B) \leq \epsilon$, then $|\mathbf{H}_{sh}(A) - \mathbf{H}_{sh}(B)| \leq \epsilon(n + \log\left(\frac{1}{\epsilon}\right))$.*

Now let X, Y be two correlated random variables on $\{0,1\}^n$, and let X be sampled according to the same distribution as X, but independently of Y. Let $A = (X, Y)$ and $B = (X', Y)$, i.e. B is the product of the marginals. By the lemma above, if $\mathbf{SD}(A, B) \leq \epsilon$, then $\mathbf{I}(X, Y) \leq \epsilon\ (n + \log\left(\frac{1}{\epsilon}\right) + 1)$.

Conversely, if $\mathbf{I}(X; Y)$ is large (with even a single bit of information), then the pair X, Y cannot be very close to being independent.

2.2 Randomness Extractors

Randomness extractors were mentioned in the introduction as an important tool in information-theoretic cryptography. Informally, an extractor is a function which takes as input some imperfect source of randomness (the various bits of which may be biased or correlated), and produces as output a short, uniformly random string.

Definition 2.1. *An efficient* $(n, t', \ell, k, \epsilon)$-extractor *is a polynomial time function* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^{\ell+k}$ *such that for all min-entropy* t' *distributions* W*, we have* $\mathbf{SD}\left(\mathsf{Ext}(W;S)\ ,\ U_{\ell+k}\right) \leq \epsilon$*, where* $\mathsf{Ext}(W;S)$ *stands for applying* Ext *to* W *using (uniformly distributed) randomness* S*.*

The difference $t' - (\ell + k)$ is usually called the *entropy loss* of the extractor (it is, informally, the difference between the entropy going in and the entropy coming out). The number of extra random bits k is usually called the seed length of the extractor and S, the seed.

Strong extractors are an important special class of extractors, which include their seed explicitly in the output. We include a separate definition for ease of reference:

Definition 2.2. *An efficient* (n, t', ℓ, ϵ)-strong extractor *is a polynomial time probabilistic function* $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^\ell$ *such that for all min-entropy* t' *distributions* W*, we have* $\mathbf{SD}\left(\langle \mathsf{Ext}(W;S), S\rangle\ ,\ \langle U_\ell, S\rangle\right) \leq \epsilon$*, where* $\mathsf{Ext}(W;S)$ *stands for applying* Ext *to* W *using (uniformly distributed) randomness* S*.*

Strong extractors can extract at most $\ell = t' - 2\log(1/\epsilon) + O(1)$ nearly random bits [67]. Many constructions match this bound (see Shaltiels' survey [72] for references). Extractor constructions are often complex since they seek to minimize the length of the seed X. For our purposes, the length of S will be less important, so pairwise independent hash functions will already give us optimal $\ell = t' - 2\log(1/\epsilon)$ [44, 8]—we explain that construction now.

Pairwise Independence and XOR Universality Wegman and Carter [20, 88] defined pairwise independence of families of functions from one finite set to another. We will use a relaxation, often called XOR universality [50].

Definition 2.3. *Let* $\mathcal{H} = \{h_i\}_{i\in\mathcal{I}}$ *is a set of functions from* $\{0,1\}^n$ *to* $\{0,1\}^\ell$*, indexed by a set* $\mathcal{I}$*. Suppose we choose* H *uniformly from* $\mathcal{H}$*.*

- *The family of functions* $\mathcal{H}$ *is* pairwise independent *(or* 2-universal*) if, for all possible pairs of distinct inputs* x, y *the random variables* $H(x), H(y)$ *are independent and uniformly distributed over* $\{0,1\}^\ell$*, that is*

$$\forall x, y \in \{0,1\}^n, x \neq y, \forall a, b \in \{0,1\}^\ell : \quad \Pr_{I\leftarrow\mathcal{I}}[h_I(x) = a \text{ and } h_I(y) = b] = \frac{1}{2^{2\ell}}$$

- *The family* $\mathcal{H}$ *is called* XOR-universal *if for all pairs of distinct inputs* x, y*, the difference* $H(x) \oplus H(y)$ *is uniformly distributed over* $\{0,1\}^\ell$*.*

Pairwise independence clearly implies XOR-universality. The reverse is not quite true; for example, it could be that some particular input is left fixed by all functions in the family. In this work, we can almost always substitute pairwise independent families with XOR-universal families, and we state the main lemmas below using XOR-universality.

A simple example of a XOR-universal family is the set of *all* functions from n bits to ℓ bits. A more commonly used family is the set of all linear functions from n bits to ℓ bits (here linearity is over $\mathbb{Z}_2$ equipped with XOR, denoted $\oplus$). A linear function can be described by a $\ell \times n$ matrix of bits; the function requires $n\ell$ bits to store.

One gains some efficiency by viewing $\{0,1\}^n$ as the larger field $GF(2^n)$. The family consists of functions $h_a(x) = ax$ where multiplication is in $GF(2^n)$. This produces only n bits of output. If ℓ is shorter than n then one can simply truncate the output to the first ℓ bits. If ℓ is greater than n, the one can simply work over $GF(2^\ell)$ instead.

The easiest construction of strong extractors is given by the "left-over hash" lemma, also called the "privacy amplification" lemma [7, 44, 46, 8].

Lemma 2.4 (Left-over hash / privacy amplification). *If $\{h_i\}_{i\in\mathcal{I}}$ is a family of pairwise independent hash functions from n bits to ℓ bits, and X is a random variable in $\{0,1\}^n$ with collision entropy $\mathbf{H}_2(X) \geq \ell + 2\log\left(\frac{1}{\epsilon}\right) + 1$, then*

$$\langle I, h_i(X)\rangle \approx_\epsilon \langle I, U_\ell\rangle$$

where $\mathcal{I} \leftarrow \mathcal{I}$, $U_\ell \leftarrow \{0,1\}^\ell$ (both drawn uniformly), and I, X and U_ℓ are independent.

The lemma is stated in terms of collision entropy; this implies the same result for min-entropy since $\mathbf{H}_\infty(X) \leq \mathbf{H}_2(X)$. The seed length of this extractor is n, and the entropy loss is $2\log\left(\frac{1}{\epsilon}\right) + 1$, as stated above.

2.3 Metric Spaces and Error-Correcting Codes

Metric Spaces A metric space is a set $\mathcal{M}$ with a distance function $\mathsf{dis} : \mathcal{M}\times\mathcal{M} \to \mathbb{R}^+ = [0,\infty)$ which obeys various natural properties (the exact properties are not important; our results are specific to the three metrics below).

In this thesis, $\mathcal{M}$ will usually be a finite set, and the distance function will only take on integer values (we will also discuss continous metrics, but only briefly). The size of the $\mathcal{M}$ will always be denoted $N = |\mathcal{M}|$. We will assume that any point in $\mathcal{M}$ can be naturally represented as a binary string of appropriate length $O(\log N)$.

We concentrate on the following metrics.

1. *Hamming metric.* Here $\mathcal{M} = \mathcal{F}^n$ over some alphabet $\mathcal{F}$ (we will mainly use $\mathcal{F} = \{0,1\}$), and $\mathsf{dis}(w,w')$ is the number of positions in which they differ.

2. *Set Difference metric.* Here $\mathcal{M}$ consists of all s-element subsets in a universe $\mathcal{U} = [n] = \{1,...,n\}$. The distance between two sets A, B is the number of points in A that are not in B. Since A and B have the same size, the distance is half of the size of their symmetric difference: $\mathsf{dis}(A,B) = \frac{1}{2}|A\triangle B|$.

3. *Edit metric.* Here again $\mathcal{M} = \mathcal{F}^n$, but the distance between w and w' is defined to be one half of the smallest number of character insertions and deletions needed to transform w into w'.

All three metrics seem natural for biometric data.

Coding Since we want to achieve error tolerance in various metric spaces, we will use *error-correcting codes* in the corresponding metric space $\mathcal{M}$. A code C is a subset $\{w_1, \ldots, w_K\}$ of K elements of $\mathcal{M}$ (for efficiency purposes, we want the map from i to w_i to be polynomial-time). The *minimum distance* of C is the smallest $d > 0$ such that for all $i \neq j$ we have $\mathsf{dis}(w_i, w_j) \geq d$. In our case of integer metrics, this means that one can detect up to $(d-1)$ "errors" in any codeword. The *error-correcting distance* of C is the largest number $\tau > 0$ such that for every $w \in \mathcal{M}$ there exists at most one codeword w_i in the ball of radius τ around w: $\mathsf{dis}(w, w_i) \leq \tau$ for at most one i. Clearly, for integer metrics we have $\tau = \lfloor (d-1)/2 \rfloor$. Since error correction will be more important in our applications, we denote the corresponding codes by $(\mathcal{M}, K, \tau)$-codes. For the Hamming and the edit metrics on strings of length n over some alphabet $\mathcal{F}$, we will sometimes call $k = \log_{|F|} K$ the *dimension* on the code, and denote the code itself as an $[n, k, d = 2\tau + 1]$-code. [1]

We will use a few definitions specific to the Hamming metric when proving lower bounds. Let $A(n, d)$ to be the largest k for which their exists an (n, k, d) code, and let $A(n, d, b)$ be the largest such k for a code all of whose codewords lie within a Hamming ball of radius b (denoted $\mathsf{Ball}_n(b)$). More generally, given any set S of 2^m points in $\{0,1\}^n$, let $A(n, d, S)$ be the largest k such that all the codewords of C belong to the set S. Finally, we let $L(n, d, m)$ denote $\min_{|S|=2^m} A(n, d, S)$. Of course, when $n = m$, we get $L(n, d, n) = A(n, d)$. In general, we can say $L(n, d, m) \leq \min(A(m, d), A(n, d, b))$, where b is such that $\mathsf{Ball}_n(b)$ has volume 2^m. The last inequality follows by choosing $S = \{0,1\}^m 0^{n-m}$ and $S = B$ respectively.

The exact determination of quantities $A(n, d)$ and $A(n, d, b)$ form the main problem of coding theory (see [1] for a summary of results about the latter). To the best of our knowledge, the quantity $L(n, d, m)$ was not explicitly studied earlier, but its exact determination seems very hard as well. We conjecture it is very close to $A(n, d, b)$ above for essentially all "natural" settings of n, d, m.

Finally, we will work a lot with *linear* codes for the Hamming metric. A code in $\{0,1\}^n$ is linear if it forms a linear subspace of $\{0,1\}^n$ equipped with bitwise XOR, that is if the XOR of any two codewords is also a codeword. If C is a linear code of dimension k, then the syndrome of C, denoted $syn_C(\cdot)$ is a linear map from n bits to $n - k$ bits such that $syn_C(w) = 0^{n-k}$ if and only if $w \in C$. In the Hamming metric, the syndrome captures the *error pattern* which has been applied to a codeword. That is, if $w = c \oplus e$, where $c \in C$, then $syn_C(w) = syn_C(e)$. Thus, to find the smallest vector e for which $w \oplus e$ is a codeword, it is not necessary to have w in hand; simply knowing $syn_C(w)$ is sufficient.

[1] In this thesis, the square brackets [] do *not* mean that the code is linear, although this is sometimes used as a convention in the literature.

Part I

Cryptography with Noisy Data

Chapter 3

Secure Sketches and Fuzzy Extractors: Cryptographic Keys from Noisy Data

This chapter introduces the primitives discussed earlier for handling noisy data: *secure sketches* and *fuzzy extractors*. We begin with formal definitions of the primitives (Section 3.1), and then turn to illustrating them with several constructions.

Constructions First, we discuss generic techniques for building secure sketches and fuzzy extractors. In Section 3.2, we show that given a secure sketch for a particular metric space, one can construct a fuzzy extractor with similar parameters. Section 3.3 describes two geometric constructions of the primitives. (1) In a transitive metric space, the existence of an efficiently decodable error-correcting code immediately leads to a secure sketch with related parameters. (2) More generally, embeddings from a source metric space into a target space allow one to construct fuzzy extractors for the target space into fuzzy extractors from the source.

Second, we describe constructions for the three specific metric spaces described in the Introduction—Hamming distance, set difference, and edit distance (Sections 3.4, 3.5, and 3.6, respectively). The next chapter (Chap. 4) investigates lower bounds on the performance of our primitives. Those results show that the constructions for the Hamming and set difference metrics are optimal, at least for some settings of the parameters.

Extensions and Applications The definitions of Section 3.1 are quite strict, and require a noisy input to be accepted as long as it is close to the original data, even though the exact error pattern may have been chosen adversarially. In Section 3.7, we consider several relaxations of the error model and show in each relaxation, we can change our constructions slightly to tolerate a larger numbers of errors.

Finally, Section 3.8 gives further details of the secure use of biometric data for password authentication. This is probably the main application of the framework presented in this chapter.

3.1 New Definitions

Let $\mathcal{M}$ be a metric space on N points with distance function dis.

Definition 3.1. *An* $(\mathcal{M}, t, t', \tau)$-secure sketch *is a randomized map* $\mathsf{SS} : \mathcal{M} \to \{0,1\}^*$ *with the following properties.*

1. *There exists a deterministic recovery function* Rec *allowing to recover* w *from its sketch* $\mathsf{SS}(w)$ *and any vector* w' *close to* w*: for all* $w, w' \in \mathcal{M}$ *satisfying* $\mathsf{dis}(w, w') \leq \tau$*, we have* $\mathsf{Rec}(w', \mathsf{SS}(w)) = w$.

2. *For all random variables* W *over* $\mathcal{M}$ *with min-entropy* t*, the average min-entropy of* W *given* $\mathsf{SS}(W)$ *is at least* t'*. That is,* $\tilde{\mathbf{H}}_\infty(W \mid \mathsf{SS}(W)) \geq t'$.

The secure sketch is efficient *if* SS *and* Rec *run in time polynomial in the representation size of a point in* $\mathcal{M}$*. We denote the random output of* SS *by* $\mathsf{SS}(W)$*, or by* $\mathsf{SS}(W; X)$ *when we wish to make the randomness explicit.*

We will have several examples of secure sketches when we discuss specific metrics. The quantity $t - t'$ is called the *entropy loss* of a secure sketch. Our proofs in fact bound $t - t'$, and the same bound holds for all values of t.

Definition 3.2. *An* $(\mathcal{M}, t, \ell, \tau, \epsilon)$ fuzzy extractor *is a given by two procedures* $(\mathsf{Gen}, \mathsf{Rep})$.

1. Gen *is a probabilistic generation procedure, which on input* $w \in \mathcal{M}$ *outputs an "extracted" string* $R \in \{0,1\}^\ell$ *and a public string* P*. We require that for any distribution* W *on* $\mathcal{M}$ *of min-entropy* t*, if* $\langle R, P \rangle \leftarrow \mathsf{Gen}(W)$*, then we have* $\mathbf{SD}\left(\langle R, P \rangle, \langle U_\ell, P \rangle\right) \leq \epsilon$.

2. Rep *is a deterministic reproduction procedure which allows one to recover* R *from the corresponding public string* P *and any vector* w' *close to* w*: for all* $w, w' \in \mathcal{M}$ *satisfying* $\mathsf{dis}(w, w') \leq \tau$*, if* $\langle R, P \rangle \leftarrow \mathsf{Gen}(w)$*, then we have* $\mathsf{Rep}(w', P) = R$.

The fuzzy extractor is efficient *if* Gen *and* Rep *run in time polynomial in the representation size of a point in* $\mathcal{M}$.

In other words, fuzzy extractors allow one to extract some randomness R from w and then successfully reproduce R from any string w' that is close to w. The reproduction is done with the help of the public string P produced during the initial extraction; yet R looks truly random even given P. To justify our terminology, notice that strong extractors (as defined in Section 2.2) can indeed be seen as "nonfuzzy" analogs of fuzzy extractors, corresponding to $\tau = 0$, $P = X$ (and $\mathcal{M} = \{0,1\}^n$).

3.2 Fuzzy Extractors from Secure Sketches

Not surprisingly, secure sketches come up very handy in constructing fuzzy extractors. Specifically, we construct fuzzy extractors from secure sketches and strong extractors. For that, we assume that one can naturally represent a point w in $\mathcal{M}$ using n bits. The strong extractor we use is the standard XOR-universal hashing construction, which has (optimal) entropy loss $2\log\left(\frac{1}{\epsilon}\right)$. The lemma, combined with secure sketches, will often produce nearly optimal fuzzy extractors.

Lemma 3.1. *Assume* SS *is a* $(\mathcal{M}, t, t', \tau)$*-secure sketch with recovery procedure* Rec*, and let* Ext *be the* (n, t', ℓ, ϵ)*-strong extractor based on XOR-universal hashing (in particular,* $\ell = t' - 2\log\left(\frac{1}{\epsilon}\right)$*). Then the following* $(\mathsf{Gen}, \mathsf{Rep})$ *is a* $(\mathcal{M}, t, \ell, \tau, \epsilon)$*-fuzzy extractor:*

- $\mathsf{Gen}(W; X_1, X_2)$*: set* $P = \langle \mathsf{SS}(W; X_1), X_2 \rangle$*,* $R = \mathsf{Ext}(W; X_2)$*, output* $\langle R, P \rangle$*.*
- $\mathsf{Rep}(W', \langle V, X_2 \rangle)$*: recover* $W = \mathsf{Rec}(W', V)$ *and output* $R = \mathsf{Ext}(W; X_2)$*.*

Lemma 3.1 follows directly from the intermediate result below (Lemma 3.2), which explains our choice of the measure $\tilde{\mathbf{H}}_\infty(A|B)$ for the average min-entropy. Lemma 3.2 says that XOR-universal hashing extracts randomness from the random variable A as if the min-entropy of A given $B = b$ were always at least $\tilde{\mathbf{H}}_\infty(A|B)$. If one wants to use a generic extractor, there is some additional loss in the parameters (see the remark after the proof of the lemma).

Lemma 3.2. *If* A, B *are random variables such that* $A \in \{0,1\}^n$ *and* $\tilde{\mathbf{H}}_\infty(A|B) \geq t'$*, and* H *is a random member of a XOR-universal hash family from* n *bits to* ℓ *bits, then* $\mathbf{SD}\left(\langle B, H, H(A)\rangle \ , \ \langle B, H, U_\ell\rangle\right) \leq \epsilon$ *as long as* $\ell \leq t' - 2\log\left(\frac{1}{\epsilon}\right)$*.*

This is a special case of Lemma A.6, but we give a (direct) proof here.

Proof. The particular extractor we chose has a smooth tradeoff between the entropy of the input and the quality of the output. For any random variable X, the left-over hash/privacy amplification lemma [7, 44, 8] states:

$$\mathbf{SD}\left(\langle H, H(X)\rangle \ , \ \langle H, U_\ell\rangle\right) \leq \sqrt{2^{-\mathbf{H}_\infty(X)} 2^\ell}$$

In our setting we have a bound on the *expected* value of $2^{-\mathbf{H}_\infty(A|B=b)}$, namely

$$\mathbb{E}\left[2^{-\mathbf{H}_\infty A} \mid B)\right] \leq 2^{-t'}.$$

Using the fact that $\mathbb{E}\left[\sqrt{Z}\right] \leq \sqrt{\mathbb{E}\left[Z\right]}$, for any non-negative r.v. Z we get:

$$\mathbb{E}_b\left[\mathbf{SD}\left(\langle H, H(A|B=b)\rangle \ , \ \langle H, U_\ell\rangle\right)\right] \leq \sqrt{2^{\ell - t'}}.$$

Now the distance of $\langle B, H, H(A)\rangle$ from $\langle B, H, U_\ell\rangle$ is the average over values of B of the distance of $\langle H, H(A)\rangle$ from $\langle H, U_\ell\rangle$. This average is exactly what was bounded above:

$$\mathbf{SD}\left(\langle B, H, H(A)\rangle, \langle P, U_\ell\rangle\right) = \mathbb{E}_B\left[\mathbf{SD}\left(\langle H, H(A)\rangle\ ,\ \langle H, U_\ell\rangle\right)\right] \leq \sqrt{2^{\ell - t'}}.$$

The extractor we use always has $\ell \leq t' - 2\log\left(\frac{1}{\epsilon}\right)$, and so the statistical difference is at most ϵ. □

Remark 3.1. One can prove an analogous form of Lemma 3.2 using any strong extractor. However, in general, the resulting reduction leads to fuzzy extractors with min-entropy loss $3\log\left(\frac{1}{\epsilon}\right)$ instead of $2\log\left(\frac{1}{\epsilon}\right)$. This may happen in the case when the extractor does not have a convex tradeoff between the input entropy and the distance from uniform of the output. Then one can instead use a high-probability bound on the min-entropy of the input (that is, if $\tilde{\mathbf{H}}_\infty(X|Y) \geq t'$ then the event $\mathbf{H}_\infty(X|Y = y) \geq t' - \log\left(\frac{1}{\epsilon}\right)$ happens with probability $1 - \epsilon$).

3.3 Two Generic Constructions

Sketches for Transitive Metric Spaces We give a general technique for building secure sketches in *transitive* metric spaces, which we now define. A permutation π on a metric space $\mathcal{M}$ is an *isometry* if it preserves distances, i.e. $\mathsf{dis}(a, b) = \mathsf{dis}(\pi(a), \pi(b))$. A family of permutations $\Pi = \{\pi_i\}_{i \in \mathcal{I}}$ acts *transitively* on $\mathcal{M}$ if for any two elements $a, b \in \mathcal{M}$, there exists $\pi_i \in \Pi$ such that $\pi_i(a) = b$. Suppose we have a family Π of transitive isometries for $\mathcal{M}$ (we will call such $\mathcal{M}$ *transtive*). For example, in the Hamming space, the set of all shifts $\pi_x(w) = w \oplus x$ is such a family (see Section 3.4 for more details on this example).

Let C be an $(\mathcal{M}, K, \tau)$-code. Then the general sketching scheme is the following: given a input $w \in \mathcal{M}$, pick a random codeword $b \in C$, pick a random permutation $\pi \in \Pi$ such that $\pi(w) = b$, and output $\mathsf{SS}(w) = \pi$. To recover w given w' and the sketch π, find the closest codeword b' to $\pi(w')$, and output $\pi^{-1}(b')$. This works when $\mathsf{dis}(w, w') \leq \tau$, because then $\mathsf{dis}(b, \pi(w')) \leq \tau$, so decoding $\pi(w')$ will result in $b' = b$, which in turn means that $\pi^{-1}(b') = w$.

A bound on the entropy loss of this scheme, which follows simply from "counting" entropies, is $|\text{“}\pi''| - \log K$, where $|\text{“}\pi''|$ is the size, in bits, of a canonical description of π. (We omit the proof, as it is a simple generalization of the proof of Lemma 3.4.) Clearly, this quantity will be small if the family Π of transifitive isometries is small and the code C is dense. (For the scheme to be usable, we also need the operations on the code, as well as π and π^{-1}, to be implementable reasonably efficiently.)

Constructions from Biometric Embeddings We now introduce a general technique that allows one to build good fuzzy extractors in some metric space $\mathcal{M}_1$ from good fuzzy extractors in some other metric space $\mathcal{M}_2$. Below, we let $\mathsf{dis}(\cdot, \cdot)_i$ denote the distance function in $\mathcal{M}_i$. The technique is to *embed* $\mathcal{M}_1$ into $\mathcal{M}_2$ so as to "preserve" relevant parameters for fuzzy extraction.

Definition 3.3. *A function $f : \mathcal{M}_1 \to \mathcal{M}_2$ is called a $(\tau_1, \tau_2, t_1, t_2)$-biometric embedding if the following two conditions hold:*

- *For all $w_1, w_1' \in \mathcal{M}_1$ such that $\mathsf{dis}(w_1, w_1')_1 \leq \tau_1$, we have $\mathsf{dis}(f(w_1), f(w_2))_2 \leq \tau_2$.*
- *For all W_1 on $\mathcal{M}_1$ of min-entropy at least t_1, $f(W_1)$ has min-entropy at least t_2.*

The following lemma is immediate:

Lemma 3.3. *If f is $(\tau_1, \tau_2, t_1, t_2)$-biometric embedding of $\mathcal{M}_1$ into $\mathcal{M}_2$ and $(\mathsf{Gen}_1(\cdot), \mathsf{Rep}_1(\cdot, \cdot))$ is a $(\mathcal{M}_2, t_2, \ell, \tau_2, \epsilon)$-fuzzy extractor, then $(\mathsf{Gen}_1(f(\cdot)), \mathsf{Rep}_1(f(\cdot), \cdot))$ is a $(\mathcal{M}_1, t_1, \ell, \tau_1, \epsilon)$-fuzzy extractor.*

Notice that a similar result does not hold for secure sketches, unless f is injective (and efficiently invertible).

We will see the utility of this particular notion of embedding (as opposed to previously defined notions) in Section 3.6.

3.4 Constructions for Hamming Distance

In this section we consider constructions for the space $\mathcal{M} = \{0,1\}^n$ under the Hamming distance metric.

The Code-Offset Construction Juels and Wattenberg [47] considered a notion of "fuzzy commitment."[1] Given a binary $[n, k, 2\tau + 1]$ error-correcting code C (not necessarily linear), they fuzzy-commit to X by publishing $W \oplus C(X)$. Their construction can be rephrased in our language to give a very simple construction of secure sketches: for random $X \leftarrow \{0,1\}^k$, set

$$\mathsf{SS}(W; X) = W \oplus C(X)\,. \tag{3.1}$$

(Note that if W is uniform, this secure sketch direcly yields a fuzzy extractor with $R = X$).

When the code C is linear, this is equivalent to revealing the syndrome of the input w, and so we do not need the randomness X. Namely, in this case we could have set

$$\mathsf{SS}(w) = \mathsf{syn}_C(w) \tag{3.2}$$

(as mentioned in the introduction, this construction also appears implicitly in the information reconciliation literature, e.g. [7, 8, 24]: when Alice and Bob hold secret values which are very close in Hamming distance, one way to correct the differences with few bits of communication is for Alice to send to Bob the *syndrome* of her word w with respect to a good linear code.)

Since the syndrome of a k-dimensional linear code is $n - k$ bits long, it is clear that $\mathsf{SS}(w)$ leaks only $n - k$ bits about w. In fact, we show the same is true even for nonlinear codes.

[1] In their interpretation, one commits to X by picking a random W and publishing $\mathsf{SS}(W; X)$.

Lemma 3.4. *For any $[n, k, 2\tau+1]$ code C and any t, SS above is a $(\mathcal{M}, t, t+k-n, \tau)$ secure sketch. It is efficient if the code C allows decoding errors in polynomial time.*

Proof. Let D be the decoding procedure of our code C. Since D can correct up to τ errors, if $v = w \oplus C(x)$ and $\mathsf{dis}(w, w') \leq \tau$, then $D(w' \oplus v) = x$. Thus, we can set $\mathsf{Rec}(w', v) = v \oplus C(D(w' \oplus v))$.

Let A be the joint variable (X, W). Together, these have min-entropy $t + k$ when $\mathbf{H}_\infty(W) = t$. Since $\mathsf{SS}(W) \in \{0,1\}^n$, we have $\tilde{\mathbf{H}}_\infty(W, X \mid \mathsf{SS}(W)) \geq t + k - n$. Now given $\mathsf{SS}(W)$, W and X determine each other uniquely, and so $\tilde{\mathbf{H}}_\infty(W \mid \mathsf{SS}(W)) \geq t + k - n$ as well. □

In Chap. 4, we present some generic lower bounds on secure sketches and extractors. Let $A(n, d)$ denote the maximum number of codewords possible in a code of distance d in $\{0,1\}^n$. Then Proposition 4.1 implies that the entropy loss of a secure sketch for the Hamming metric is at least $n - \log A(n, 2\tau + 1)$, when the input is uniform (that is, when $t = n$). This means that the code-offset construction above is optimal for the case of uniform inputs. Of course, we do not know the exact value of $A(n, d)$, never mind any efficiently decodable codes which meet the bound, for most settings of n and d. Nonetheless, the code-offset scheme gets as close to optimality as is possible in coding.

Getting Fuzzy Extractors As a warm-up, consider the case when W is uniform ($t = n$) and look at the code-offset sketch construction: $V = W \oplus C(X)$. Setting $R = X$, $P = V$ and $\mathsf{Rep}(W', V) = D(V \oplus W')$, we clearly get an $(\mathcal{M}, n, k, \tau, 0)$ fuzzy extractor, since V is truly random when W is random, and therefore independent of X. In fact, this is exactly the usage proposed by Juels-Wattenberg, except they viewed the above fuzzy extractor as a way to use W to "fuzzy commit" to X, without revealing information about X.

Unfortunately, the above construction setting $R = X$ only works for uniform W, since otherwise V could leak information about X. However, by using the construction in Lemma 3.1, we get

Lemma 3.5. *Given any $[n, k, 2\tau + 1]$ code C and any t, ϵ, we can get an $(\mathcal{M}, t, \ell, \tau, \epsilon)$ fuzzy extractor, where $\ell = t + k - n - 2\log(1/\epsilon)$. The recovery Rep is efficient if C allows decoding errors in polynomial time.*

3.5 Constructions for Set Difference

Consider the collection of all sets of a particular size s in a universe $\mathcal{U} = [n] = \{1, ..., n\}$. The distance between two sets A, B is the number of points in A that are not in B. Since A and B have the same size, the distance is half of the size of their symmetric difference: $\frac{1}{2}\mathsf{dis}(A, B) = |A \triangle B|$. If A and B are viewed as n-bit characteristic vectors over $[n]$, this metric is the same as the Hamming metric (scaled by 1/2). Thus, the set difference metric can be viewed as a restriction of the binary

Hamming metric to all the strings with exactly s nonzero components. However, one typically assumes that n is much larger than s, so that representing a set by n bits is much less efficient than, say writing down a list of elements, which requires $(s \log n)$ bits.

Large Versus Small Universes Most of this section studies situations where the universe size n is super-polynomial in the set size s. We call this the large universe setting. By contrast, the small universe setting refers to situations in which $n = poly(s)$. We want our various constructions to run in polynomial time and use polynomial storage space. Thus, the large universe setting is exactly the setting in which the n-bit string representation of a set becomes too large to be usable. We consider the small-universe setting first, since it appears simpler (Section 3.5.1). The remaining subsections consider large universes.

3.5.1 Small Universes

When the universe size is polynomial in s, there are a number of natural constructions. Perhaps the most direct one, given previous work, is the construction of Juels and Sudan [48]. Unfortunately, that scheme achieves relatively poor parameters (see Section 3.5.2).

We suggest two possible constructions: first, to represent sets as n-bit strings and use the constructions of the previous section (with the caveat that Hamming distance is off by a factor of 2 from set difference). The second construction, presented below, goes directly through codes for set difference, also called "constant-weight" codes.

In order to be able to compare the constructions, and for consistency with the coding theory literature, we will in fact work with the Hamming metric here. Thus, codes which correct any τ errors in set difference will have minimum distance at least $4\tau + 1$.

Permutation-based Sketch Recall the general construction of Section 3.1 for transitive metric spaces, and observe that the set difference space is transitive. Specifically, the family of permutations Π we use is simply the one induced by the set of all permutations on the universe $[n]$. Let $C \subseteq \{0,1\}^n$ be any $[n, k, d]$ code (nonlinear) in which all words have weight exactly s, and view elements of the code as sets of size s. We obtain the following scheme, which produces a sketch of length $n \log n$:

Algorithm 3.1 (Permutation-based sketch). Input: a set $A \subseteq \mathcal{U} = [n]$ of size s.

1. Choose $B \subseteq [n]$ at random from the code C.
2. Choose a random permutation $\pi : [n] \to [n]$ such that $\pi(A) = B$:
   (That is, choose a random matching between A and B and a random matching between $[n] - A$ and $[n] - B$.)
3. Output $\mathsf{SS}(A) = \pi$ (say, by listing $\pi(1), ..., \pi(n)$).

Lemma 3.6. *Suppose that C is a $[n, k, d]$ constant-weight-s code (for Hamming distance), then:*

1. *If $d \geq 4\tau + 1$, there is an algorithm $\mathsf{Rec}()$ such that $\mathsf{Rec}(A', \mathsf{SS}(A)) = A$ for any sets A, A' such that $\frac{1}{2}|A \triangle A'| \leq \tau$. The algorithm is efficient if C has an efficient decoding algorithm.*

2. *The left-over entropy is $\tilde{\mathbf{H}}_\infty(A \mid \mathsf{SS}(A)) \geq \mathbf{H}_\infty(A) + k - \log \binom{n}{s}$.*

Proof. (1) Given π and A', we can compute $B' = \pi^{-1}(A')$. The intersection of B and B' is the same size as $A \cap A'$, and so the Hamming distance between the characteristic vectors of B and B' is at most 2τ. Since the code has minimum distance $d \geq 4\tau + 1$, it can correct 2τ errors, and so the closest codeword is $B = \pi^{-1}(A)$. All operations are $n \log n$-time except for (possibly) the random choice of B in the algorithm and the decoding.

(2) Let X be the randomness used by the sketching algorithm. There are $s!$ possibilities for the matching from A to B and $(n-s)!$ possibilities for the matching from $[n]-A$ to $[n]-B$. Hence, the min-entropy of the pair (A, X) is $\mathbf{H}_\infty(A)+\log(s!(n-s)!)$. There are $n!$ possibilities for the sketch π, and so the average min-entropy of (A, X) given $\mathsf{SS}(A)$ is at least $\mathbf{H}_\infty(A) + \log(s!(n-s)!) - \log(n!) = \mathbf{H}_\infty(A) - \log \binom{n}{s}$. Given A and $\mathsf{SS}(A)$ we can recover X exactly, and so $\tilde{\mathbf{H}}_\infty(A \mid \mathsf{SS}(A))$ is the same as $\tilde{\mathbf{H}}_\infty(A, X \mid \mathsf{SS}(A))$. □

Comparing the Hamming Scheme with the Permutation Scheme In order to get a feeling for how the random permutation technique compares to simply using Hamming-based schemes directly, we recall some notation from the coding theory literature. Let $A(n, d, s)$ denote the maximum size of a binary code for which all codewords have weight exactly s. Here n is the length of the code and d is the minimum distance. Let $A(n, d)$ denote the maximum size of an (unrestricted) binary code of length n and minimum distance d. In all cases, we're interested in codes with minimum distance $d \geq 4\tau + 1$, since we want to correct τ errors in the set difference metric.

The code-offset construction was shown to have entropy loss $n - \log A(n, d)$ if an optimal code is used; the random permutation scheme can have entropy loss $\log \binom{n}{s} - \log A(n, d, s)$ for an optimal code. The Bassalygo-Elias inequality (see [53]) shows that the bound on the random permutation scheme is always at least as good as the bound on the code offset scheme: $A(n, d) \cdot 2^{-n} \leq A(n, d, s) \cdot \binom{n}{s}^{-1}$. This implies that $n - \log A(n, d) \geq \log \binom{n}{s} - \log A(n, d, s)$. Moreover, standard packing arguments give better constructions of constant-weight codes than they do of ordinary codes. [2] In fact, the random permutations scheme is optimal for this metric, just as the code-offset scheme is optimal for the Hamming metric: Proposition 4.1 shows that the

[2] This comes from the fact that the intersection of a ball of radius d with the set of all words of weight s is much smaller than the ball of radius d itself.

min-entropy loss of a secure sketch must be at least $\log \binom{n}{s} - \log A(n, d, s)$, in the case of a uniform secret set A. Thus in principle, it is better to use the random permutation scheme. Nonetheless, there are caveats. First, we do not know of *explicitly* constructed constant-weight codes that beat the Elias-Bassalygo inequality and would thus lead to better entropy loss for the random permutation scheme than for the Hamming scheme (see [14] for more on constructions of constant-weight codes and [1] for upper bounds). Second, much more is known about efficient implementation of decoding for ordinary codes than for constant-weight codes; for example, one can find off-the-shelf hardware and software for decoding many binary codes. In practice, the Hamming-based scheme is likely to be more useful.

3.5.2 Improving the Construction of Juels and Sudan

We now turn to the large universe setting, where n is super-polynomial in s. Juels and Sudan [48] proposed a secure sketch for the set difference metric (called a "fuzzy vault" in that paper). They assume for simplicity that $n = |\mathcal{U}|$ is a prime power and work over the field $\mathcal{F} = GF(n)$. On input set A, the sketch they produce is a set of r pairs of points (x_i, y_i) in $\mathcal{F}$, with $s < r \leq n$.

Algorithm 3.2 (Juels-Sudan Secure Sketch). Input: a set $A \subseteq \mathcal{U}$.

1. Choose $p()$ at random from the set of polynomials of degree at most $k = s - 2\tau - 1$ over $\mathcal{F}$.
   Write $A = \{x_1, ..., x_s\}$, and let $y_i = p(x_i)$ for $i = 1, ..., s$.

2. Choose $r - s$ distinct points $x_{s+1}, ..., x_r$ at random from $\mathcal{F} - A$.

3. For $i = s + 1, ..., r$, choose $y_i \in \mathcal{F}$ at random such that $y_i \neq p(x_i)$.

4. Output $\mathsf{SS}(A) = \{(x_1, y_1), ..., (x_r, y_r)\}$ (in lexicographic order of x_i).

The parameter r dictates the amount of storage necessary, one on hand, and also the security of the scheme (that is, for $r = s$ the scheme leaks all information and for larger and larger r there is less information about A). Juels and Sudan actually propose two analyses for the scheme. First, they analyze the case where the secret A is distributed uniformly over all subsets of size s. Second, they provide an analysis of a nonuniform password distribution, but only for the case $r = n$ (that is, their analysis only applies in the small universe setting, where $\Omega(n)$ storage is acceptable). Here we give a simpler analysis which handles nonuniformity and any $r \leq n$. We get the same results for a broader set of parameters.

Lemma 3.7. *The entropy loss of the Juels-Sudan scheme* $\mathsf{SS}()$ *above is at most* $2\tau \log n + \log \binom{n}{r} - \log \binom{n-s}{r-s}$.

Proof. As for the code-offset, we can simply count entropies. Let X denote the random bits used by the algorithm to generate $\mathsf{SS}(A)$. Choosing the polynomial p

requires $s - 2\tau$ random choices from $\mathcal{F}$. The choice of the remaining x_i's requires $\log \binom{n-s}{r-s}$ bits, and choosing the $y_i's$ requires $r - s$ random choices from $\mathcal{F}$ (we will ignore the difference between $\mathcal{F}$ and $\mathcal{F} - \{x_i\}$ here since it doesn't affect the result significantly). The min-entropy of the pair A, X is thus $\tilde{\mathbf{H}}_\infty(A, X) = \tilde{\mathbf{H}}_\infty(A) + (r - 2\tau)\log(n) + \log \binom{n-s}{r-s}$. The output can be described in $\log \left(\binom{n}{r} n^r\right)$ bits, and hence we get that $\tilde{\mathbf{H}}_\infty(A, X \mid \mathsf{SS}(A)) = \tilde{\mathbf{H}}_\infty(A) - 2\tau \log n + \log \binom{n-s}{r-s} - \log \binom{n}{r}$. Finally, note that X is entirely determined by A and $\mathsf{SS}(A)$, so the entropy of A, X given $\mathsf{SS}(A)$ is the same as the entropy of A given $\mathsf{SS}(A)$. □

In the large universe setting, we will have $r \ll n$ (since we wish to have storage polynomial in s). In that setting, the bound on the entropy loss of the Juels-Sudan scheme is in fact very large. We can rewrite the entropy loss as $2\tau \log n - \log \binom{r}{s} + \log \binom{n}{s}$, using the identity $\binom{n}{r}\binom{r}{s} = \binom{n}{s}\binom{n-s}{r-s}$. Now the entropy of A is at most $\binom{n}{s}$, and so our lower bound on the remaining entropy is $(\log \binom{r}{s} - 2\tau \log n)$. To make this quantity large requires making r very large.

Improved JS Sketches We suggest a modification of the Juels-Sudan scheme with entropy loss at most $2\tau \log n$ and storage $s \log n$. Our scheme has the advantage of being even simpler to analyze. As before, we assume n is a prime power and work over $\mathcal{F} = GF(n)$. An intuition for the scheme is that the numbers $y_{s+1}, ..., y_r$ from the JS scheme need not be chosen at random. One can instead evaluate them as $y_i = p'(x_i)$ for some polynomial p'. One can then represent the entire list of pairs (x_i, y_i) using only the coefficients of p'.

Algorithm 3.3 (Modified JS Secure Sketch). Input: a set $A \subseteq \mathcal{U}$.

1. Choose $p()$ at random from the set of polynomials of degree at most $k = s - 2\tau - 1$ over $\mathcal{F}$.

2. Let $p'()$ be the unique monic polynomial of degree exactly s such that $p'(x) = p(x)$ for all $x \in A$.
   (Write $p'(x) = x^s + \sum_{i=0}^{s-1} a_i x^i$. Solve for $a_0, ..., a_{s-1}$ using the s linear constraints $p'(x) = p(x)$, $x \in A$.)

3. Output the list of coefficients of $p'()$, that is $\mathsf{SS}(A) = (a_0, ..., a_{s-1})$.

First, observe that solving for $p'()$ in Step 2 is always possible, since the s constraints $\sum_{i=0}^{s-1} a_i x^i = p(x) - x^s$ are in fact linearly independent (this is just polynomial interpolation).

Second, this sketch scheme can tolerate τ set difference errors. Suppose we are given a set $B \subseteq \mathcal{U}$ which agrees with A in at least $s - \tau$ positions. Given $p' = \mathsf{SS}(A)$, one can evaluate p' on all the points in the set B. The resulting vector agrees with p on at least $s - \tau$ positions, and using the decoding algorithm for Reed-Solomon codes, one can thus reconstruct p exactly (since $k = s - 2\tau - 1$). Finally, the set A can be recovered by finding the roots of the polynomial $p' - p$: since $p' - p$ is not identically

zero and has degree exactly s, it can have at most s roots and so $p' - p$ is zero only on A.

We now turn to the entropy loss of the scheme. The sketching scheme invests $(s - 2\tau)\log n$ bits of randomness to choose the polynomial p. The number of possible outputs p' is n^s. If X is the invested randomness, then the (average) min-entropy (A, X) given $\mathsf{SS}(A)$ is at least $\tilde{\mathbf{H}}_\infty(A) - 2\tau \log n$. The randomness X can be recovered from A and $\mathsf{SS}(A)$, and so we have $\tilde{\mathbf{H}}_\infty(A \mid \mathsf{SS}(A)) \geq \tilde{\mathbf{H}}_\infty(A) - 2\tau \log n$. We have proved:

Lemma 3.8 (Analysis of Modified JS). *The entropy loss of the modified JS scheme is at most $2\tau \log n$. The scheme has storage $(s+1)\log n$ for sets of size s in $[n]$, and both the sketch generation $\mathsf{SS}()$ and the recovery procedure $\mathsf{Rec}()$ run in polynomial time.*

The short length of the sketch makes this scheme feasible for essentially any ratio of set size to universe size (we only need $\log n$ to be polynomial in s). Moreover, for large universes the entropy loss $2\tau \log n$ is essentially optimal for the uniform case $t = \log \binom{n}{s}$. Proposition 4.1 shows that for a uniformly distributed input, the best possible entropy loss is $t - t' \geq \log \binom{n}{s} - \log A(n, s, 4\tau + 1)$, where $A(n, s, d)$ is the maximum size of a code of constant weight s and minimum Hamming distance d. Using a bound of Agrell *et al* ([1], Theorem 12), the entropy loss is at least:

$$t - t' \geq \log \binom{n}{s} - \log A(n, s, 4\tau + 1)$$
$$\geq \log \binom{n}{s} - \log \left(\binom{n}{s - 2\tau} \Big/ \binom{s}{s - 2\tau} \right) = \log \binom{n - s + 2\tau}{2\tau}$$

When $n \geq s$, this last quantity is roughly $2\tau \log n$, as desired.

3.5.3 Large Universes via the Hamming Metric: Sublinear-Time Decoding (Revised December '04)

In this section, we show that code-offset construction can in fact be adapted for small sets in large universe, using specific properties of algebraic codes. We will show that BCH codes, which contain Hamming and Reed-Solomon codes as special cases, have these properties.

Syndromes of Linear Codes For a $[n, k, d]$ linear code C over $GF(q)$ with parity check matrix H, recall that the syndrome of a word $w \in GF(q)^n$ is $\mathsf{syn}(w) = Hw$. The syndrome has length $n - k$, and the code is exactly the set of words c such that $\mathsf{syn}(c) = 0^{n-k}$. The syndrome captures all the information necessary for decoding. That is, suppose a codeword c is sent through a channel and the word $w = c + e$ is received. First, the syndrome of w is the syndrome of e: $\mathsf{syn}(w) = \mathsf{syn}(c) + \mathsf{syn}(e) = 0 + \mathsf{syn}(e) = \mathsf{syn}(e)$. Moreover, for any value u, there is at most one word e of weight less than $d/2$ such that $\mathsf{syn}(e) = u$ (the existence of a pair of distinct words e_1, e_2

would mean that $e_1 - e_2$ is a codeword of weight less than d). Thus, knowing syndrome $\mathsf{syn}(w)$ is enough to determine the error pattern e if not too many errors occurred.

As mentioned before, we can reformulate the code-offset construction in terms of syndrome: $\mathsf{SS}(w) = \mathsf{syn}(w)$. The two schemes are equivalent: given $\mathsf{syn}(w)$ one can sample from $w+C(X)$ by choosing a random string v with $\mathsf{syn}(v) = \mathsf{syn}(w)$; conversely, $\mathsf{syn}(w + C(X)) = \mathsf{syn}(w)$. This reformulation gives us no special advantage when the universe is small: storing $w + C(X)$ is not a problem. However, it's a substantial improvement when $n \gg n - k$.

Syndrome Manipulation for Small-Weight Words Suppose now that we have a small set $A \subseteq [n]$ of size s, where $n \gg s$. Let $x_A \in \{0,1\}^n$ denote the characteristic vector of A. If we want to use $\mathsf{syn}(x_A)$ as the sketch of A, then we must choose a code with $n - k \leq \log \binom{n}{s} \approx s \log n$, since the sketch has entropy loss $(n-k)$ and the maximum entropy of A is $\log \binom{n}{s}$.

Binary BCH codes are a family of $[n, k, d]$ linear codes with $d = 4\tau + 1$ and $k = n - 2\tau \log n$ (assuming $n + 1$ is a power of 2) (see, e.g. [53]). These codes are optimal for $\tau \ll n$ by the Hamming bound, which implies that $k \leq n - \log \binom{n}{2\tau}$ [53].[3] Using the code-offset sketch with a BCH code C, we get entropy loss $n - k = 2\tau \log n$, just as we did for the modified Juels-Sudan scheme (recall that $d \geq 4\tau + 1$ allows us to correct τ set difference errors).

The only problem is that the scheme appears to require computation time $\Omega(n)$, since we must compute $\mathsf{syn}(x_A) = Hx_A$ and, later, run a decoding algorithm to recover x_A. For BCH codes, this difficulty can be overcome. A word of small weight x can be described by listing the positions on which it is nonzero. We call this description the *support* of x and write $\mathsf{supp}(x)$ (that is $\mathsf{supp}(x_A) = A$).

Lemma 3.9. *For a $[n, k, d]$ binary BCH code C one can compute:*

1. $\mathsf{syn}(x)$*, given* $\mathsf{supp}(x)$*, and*

2. $\mathsf{supp}(x)$*, given* $\mathsf{syn}(x)$ *(when x has weight at most $(d-1)/2$),*

in time polynomial in $|\mathsf{supp}(x)| = weight(x) \cdot \log(n)$ and $|\mathsf{syn}(x)| = n - k$.

The proof of Lemma 3.9 requires a careful reworking of the standard BCH decoding algorithm. The details are presented in Section 3.5.4. For now, we present the resulting sketching scheme for set difference. The algorithm works in the field $GF(2^m) = GF(n+1)$, and assumes a generator α for $GF(2^m)$ has been chosen ahead of time.

Algorithm 3.4 (BCH-based Secure Sketch). Input: a set $A \subseteq GF(2^m)^*$ of size s (that is, $n = 2^m - 1$).

[3] The Hamming bound is based on the observation that for any code of distance d, the balls of radius $\lfloor (d-1)/2 \rfloor$ centered at various codewords must be disjoint. Each such ball contains $\binom{n}{\lfloor (d-1)/2 \rfloor}$ points, and so $2^k \binom{n}{\lfloor (d-1)/2 \rfloor} \leq 2^n$. In our case $d = 4\tau + 1$ and so the bound yields $k \leq n - \log \binom{n}{2\tau}$.

1. Let $A_i = \sum_{x \in A} x^i$.

2. Output $\mathsf{SS}(A) = (A_1, A_3, A_5, ..., A_{4\tau+1})$ (computations in $GF(2^m)$).

Lemma 3.9 yields the algorithm $\mathsf{Rec}()$ which recovers A from $\mathsf{SS}(A)$ and any set which intersects A in at least $s - \tau$ points. However, the bound on entropy loss is easy to see: the output is $2\tau \log n$ bits long, and hence the entropy loss is at most $2\tau \log n$. We obtain:

Theorem 3.10. *The BCH scheme above is a $[t, t-2\tau \log n, t]$ secure sketch scheme for set difference with storage $2\tau \log n$. The algorithms SS and Rec both run in polynomial time.*

3.5.4 Syndrome Decoding in Sublinear Time (Revised December '04)

We show that the standard decoding algorithm for BCH codes can be modified to run in time polynomial in the length syndrome. This works for BCH codes over any field $GF(q)$, which include Hamming codes in the binary case and Reed-Solomon for the case $n = q - 1$. BCH codes are handled in detail in many textbooks (e.g., [53]); our presentation here is quite terse. For simplicity, we only discuss primitive, narrow-sense BCH codes here; the discussion extends easily to the general case.

The algorithm discussed here has revised due to an error pointed out by Ari Trachtenberg.

We'll use a slightly non-standard formulation of BCH codes, in which the positions of the code are listed in a different order than usual.

Definition 3.4. *Let $n = q^m - 1$. We will work in two finite fields: $GF(q)$ and a larger extension field $\mathcal{F} = GF(q^m)$. The (narrow-sense, primitive) BCH code of designed distance δ over $GF(q)$ (of length n) is given by the set of vectors of the form $\left(c_x\right)_{x \in \mathcal{F}^*}$ such that each c_x is in the smaller field $GF(q)$, and the vector satisfies the constraints $\sum_{x \in \mathcal{F}^*} c_x x^i = 0$, for $i = 1, \ldots, \delta - 1$, with arithmetic done in the larger field $\mathcal{F}$.*

To explain this definition, let us fix a generator α of the multiplicative group of the large field $\mathcal{F}^*$. For any vector of coefficients $\left(c_x\right)_{x \in \mathcal{F}^*}$, we can define a polynomial

$$c(z) = \sum_{x \in GF(q^m)^*} c_x z^{\mathsf{dlog}(x)}$$

where $\mathsf{dlog}(x)$ is the discrete logarithm of x with respect to α. The conditions of the definition above then reduce to the requirement that $c(\alpha^i) = 0$ for $i = 1, \ldots, \delta - 1$.

We can simplify this somewhat. Because the coefficients c_x are in $GF(q)$, they satisfy $c_x^q = c_x$. Using the identity $(x + y)^q = x^q + y^q$, which holds even in the large field $\mathcal{F}$, we have $c(\alpha^i)^q = \sum_{x \neq 0} c_x^q x^{iq} = c(\alpha^{iq})$. Thus, roughly a $1/q$ fraction of the conditions in the definition are redundant: we only need to check that they hold for $i \in \{1, ..., \delta - 1\}$ such that $q \not| i$.

The syndrome of a word (not necessarily a codeword) $(p_x)_{x\in\mathcal{F}^*} \in GF(q)^n$ with respect to the BCH code above is the vector

$$\mathsf{syn}(p) = p(\alpha^1), \ldots, p(\alpha^{\delta-1}), \quad \text{where} \quad p(\alpha^i) = \sum_{x\neq 0} p_x x^i.$$

As mentioned above, we do not in fact have to include the values $p(\alpha^i)$ such that $q|i$.

Representation Issues Because we are interested in algorithms that run in time much less than the size of the field, the exact representation of field elements is important. The elements of a finite field are typically represented as strings of m symbols from an alphabet of size q (for simplicity, think of $q = 2$). This representation allows one to add and multiply field elements relatively efficiently (in time $\tilde{O}(m \log q)$, where the $\tilde{O}$ notation hides polylogarithmic factors). We can define a canonical ordering $\mathsf{lex} : \mathcal{F}^* \to \{1, \ldots, n\}$ corresponding to lexicographic ordering of the strings representing field elements. This allows us to write $\mathcal{F}^* = \{x_1, ..., x_n\}$ where $x_i \stackrel{def}{=} \mathsf{lex}^{-1}(i)$. The important point here is that both $\mathsf{lex}()$ and $\mathsf{lex}^{-1}()$ are efficiently computable (in fact, in the case $q = 2$, the binary representations of x and $\mathsf{lex}(x)$ could simply be taken to be identical).

Note that in the typical formulation of BCH codes, one orders elements according to the discrete logarithm (i.e. $x_i = \alpha^i$). The problem is that computing the discrete logarithm is generally thought to require much more than polynomial time (here the length of the input is $m \log q$). We will need to compute i from x_i and vice versa, and so the lexicographic ordering of the field elements is more appropriate. A wrong of ordering was the source of an error in an earlier version of this work, and so we've taken particular care to make the issue explicit here.

Computing with Low-Weight Words A low-weight word $p \in GF(q)^n$ can be represented either as a long string or, more compactly, as a list of positions where it is nonzero and its values at those points. We call this representation the support list of p and denote it $\mathsf{supp}(p) = \{(x, p_x)\}_{x:p_x\neq 0}$.

Lemma 3.11. *For a q-ary BCH code C of designed distance δ, one can compute:*

1. $\mathsf{syn}(p)$, *given* $\mathsf{supp}(p)$, *and*

2. $\mathsf{supp}(p)$, *given* $\mathsf{syn}(p)$ *(when p has weight at most $(\delta-1)/2$),*

in time polynomial in $|\mathsf{supp}(p)| = weight(p)\cdot\log(n)\cdot\log(q)$ and $|\mathsf{syn}(p)| = (n-k)\log q$. Note that $|\mathsf{syn}(p)| \leq m\,\lceil(\delta-2)(q-1)/q\rceil \approx \delta\cdot m\cdot(1-1/q)\log(q)$. In particular, when $q = 2$ we get $|syn(p)| = m(\delta-1)/2$.

Proof. Recall that $\mathsf{syn}(p) = (p(\alpha), ..., p(\alpha^{\delta-1}))$ where $p(\alpha^i) = \sum_{x\neq 0} p_x x^i$. Part (1) is easy, since to compute the syndrome we only need to compute the powers of x. This requires about $\delta\cdot weight(p)$ multiplications in $\mathcal{F}$. For Part (2), we adapt the standard

BCH decoding algorithm, based on its presentation in [53]. Let $M = \{x \in \mathcal{F}^* | p_x \neq 0\}$, and define

$$\sigma(z) \stackrel{def}{=} \prod_{x \in M} (1 - xz) \quad \text{and} \quad \omega(z) \stackrel{def}{=} \sigma(z) \sum_{y \in M} \frac{p_j yz}{(1 - yz)}$$

Since $(1 - yz)$ divides $\sigma(z)$ for $j \in M$, we see that $\omega(z)$ is in fact a polynomial of degree at most $|M| = weight(p) \leq (\delta - 1)/2$. The polynomials $\sigma(z)$ and $\omega(z)$ are known as the error locator polynomial and evaluator polynomial, respectively.

We will in fact work with our polynomials modulo z^δ. In this arithmetic the inverse of $(1 + xz)$ is $\sum_{\ell=1}^{\delta} (xz)^{\ell-1}$, that is

$$(1 + xz) \sum_{\ell=1}^{\delta} (xz)^{\ell-1} \equiv 1 \mod z^\delta.$$

We are given $p(\alpha^\ell)$ for $\ell = 1, ..., \delta$. Let $S(z) = \sum_{\ell=1}^{\delta-1} p(\alpha^\ell) z^\ell$. Note that $S(z) \equiv \sum_{x \in M} p_x \frac{xz}{(1+xz)} \mod z^\delta$. This implies that

$$S(z)\sigma(z) \equiv \omega(z) \mod z^\delta.$$

The algorithm will consist of finding any non-zero solution $w'(z), \sigma'(z)$ to this congruence. This will be good enough since the solution $\omega(), \sigma()$ is "unique" in the following sense: any other solution $\omega'(z), \sigma'(z)$ satisfies $w'(z)/\sigma'(z) = \omega(z)/\sigma(z)$. To see why this is, multiply the initial congruence by $\sigma'()$ to get $\omega(z)\sigma'(z) \equiv \sigma(z)\omega'(z) \mod z^\delta$. Since the both sides of the congruence have degree at most $\delta - 1$, they are in fact equal as polynomials.

Thus it is sufficient to find any solution $\sigma'(), \omega'()$ to the congruence $S(z)\sigma'(z) = \omega'(z) \mod z^\delta$ and reduce the resulting fraction $\omega'(z)/\sigma'(z)$ to obtain a solution $\omega(), \sigma()$ of minimal degree. Finally, the roots of $\sigma(z)$ are the points x^{-1} for $x \in M$, and the exact value of p_x can be recovered using the equation $\omega(x^{-1}) = p_x \prod_{y \in M, y \neq x} (1 - yx^{-1})$.

Solving the congruence only requires solving a system of $\delta - 1$ linear equations involving δ variables, which is certainly polynomial in $\delta \log(q^m)$. The reduction of the fraction $\omega'(z)/\sigma'(z)$ requires only running Euclid's algorithm for finding the g.c.d. of two polynomials. Finally, finding the roots of $\sigma()$ can be done in time quadratic in the degree of $\sigma()$, which is at most $\delta/2$. The overall running time of the decoding procedure is $\tilde{O}(\delta^3 m \log q)$. This yields an improvement over standard decoding algorithms (which run in time $\tilde{O}(n) = \tilde{O}(q^m)$) roughly whenever $\delta = o(\sqrt[3]{n}/log(n))$. □

A Dual View of the Algorithm Readers may be used to seeing a different formulation of BCH codes, in which a codeword is the vector $c(x_1), \ldots, c(x_n)$ given by polynomials over the large field $\mathcal{F}$ of degree at most $n - \delta$ such that $c(x_i) \in GF(q)$ for all i. The equivalence of the two formulations is standard.

The syndrome and algorithm above have a natural interpretation in this evaluation-based formulation. For any polynomial $P(z)$ over $\mathcal{F}$ of degree at most $n - 1$, the

syndrome of the vector $(P(x_1), \ldots, P(x_n))$ is in fact the top $\delta - 1$ coefficients of P. That is,

$$\sum_{x \in \mathcal{F}^*} P(x) x^i = -p^{(n-i)}, \quad \text{where} \quad P(z) = \sum_{i=0}^{n-1} p^{(i)} z^i.$$

This is an example of a remarkable duality between evaluations of polynomials and their coefficients: the syndrome can be viewed both as the evaluation of a polynomial whose coefficients are given by the vector, or as the the coefficients of the polynomial whose evaluations are given by a vector.

The algorithm from the previous lemma can also be viewed naturally in this dual world: given $\mathsf{syn}(P(x_1), ..., P(x_n))$, the goal is to find a polynomial $\sigma(z)$, of degree at most $(\delta - 1)/2$, such that the product $P(z)\sigma(z)$ is equivalent to a polynomial of degree at most $n - (\delta - 1)/2$. The equivalence here comes from the fact that in a field of size q^m the relation $x^{q^m - 1} = 1$ always holds for $x \neq 0$, and so we may work with polynomials modulo the relation $z^{q^m - 1} - 1$ (the normal relation would be $z^{q^m} - z$, but we are not evaluating the polynomials at the point 0 so we may use a more restricted relation). It is easy to see why such a polynomial σ must exist when $(P(x_1), ..., P(x_n))$ has weight at most $(\delta - 1)/2$, since the error locator polynomial $\sigma(z) = \prod_{x \in M}(z - x)$ will satisfy the requirement. Finding a non-zero solution involves solving the same system of linear equations as bofre, except that now we may view them as setting the top coefficients of $P(z)\sigma(z) \mod (z^{q^m - 1} - 1)$ to zero (instead of the top coefficients of $S(z)\sigma(z) \mod z^\delta$).

As before, the roots of $\sigma(z)$ give a super-set of M (the proof is very similar to the one above). The exact set M and the values of $P(x)$ inside M can then be recovered by solving the linear system of equations implied by the value of the syndrome.

A more detailed version of the standard decoding algorithm in the "evaluation-based" view of BCH codes may be found in Sudan's lecture notes, Chapter 10.

3.6 Constructions for Edit Distance

First we note that simply applying the same approach as we took for the transitive metric spaces before (the Hamming space and the set difference space for small universe sizes) does not work here, because the edit metric does not seem to be transitive. Indeed, it is unclear how to build a permutation π such that for any w' close to w, we also have $\pi(w')$ close to $x = \pi(w)$. For example, setting $\pi(y) = y \oplus (x \oplus w)$ is easily seen not to work with insertions and deletions. Similarly, if I is some sequence of insertions and deletions mapping w to x, it is not true that applying I to w' (which is close to w) will necessarily result in some x' close to x. In fact, then we could even get $\mathsf{dis}(w', x') = 2\mathsf{dis}(w, x) + \mathsf{dis}(w, w')$.

Perhaps one could try to simply embed the edit metric into the Hamming metric using known embeddings, such as conventionally used low-distorion embeddings, which provide that all distances are preserved up to some small "distortion" factor. However, there are no known nontrivial low-distortion embeddings from the edit metric to the Hamming metric. Moreover, it was recently proved by Andoni et al [4] that

no such embedding can have distortion less than 3/2, and it was conjectured that a much stronger lower bound should hold.

Thus, as the previous approaches don't work, we turn to the embeddings we defined specifically for fuzzy extractors: biometric embeddings. Unlike low-distortion embeddings, biometric embeddings do not care about relative distances, as long as points that were "close" (closer than τ_1) do not become "distant" (farther apart than τ_2). The only additional requirement of biometric embeddings is that they preserve some min-entropy: we do not want too many points to collide together, although collisions are allowed, even collisions of distant points. We will build a biometric embedding from the edit distance to the set difference.

A *c-shingle* [13], which is a length-c consecutive substring of a given string w. A *c-shingling* [13] of a string w of length n is the set (ignoring order or repetition) of all $(n - c + 1)$ c-shingles of w. Thus, the range of the c-shingling operation consists of all nonempty subsets of size at most $n - c + 1$ of $\{0,1\}^c$. To simplify our future computations, we will always arbitrarily pad the c-shingling of any string w to contain precisely n distinct shingles (say, by adding the first $n - |c\text{-shingling}|$ elements of $\{0,1\}^c$ not present in the given c-shingling). Thus, we can define a deterministic map $\mathsf{SH}_c(w)$ which maps w into n substrings of $\{0,1\}^c$, where we assume that $c \geq \log_2 n$. Let $\mathsf{Edit}(n)$ stand for the edit metric over $\{0,1\}^n$, and $\mathsf{SDif}(N, s)$ stand for the set difference metric over $[N]$ where the set sizes are s. We now show that c-shingling yields pretty good biometric embeddings for our purposes.

Lemma 3.12. *For any $c > \log_2 n$, SH_c is a $(\tau_1, \tau_2 = c\tau_1, t_1, t_2 = t_1 - \frac{n \log_2 n}{c})$-biometric embedding of $\mathsf{Edit}(n)$ into $\mathsf{SDif}(2^c, n)$.*

Proof. Assume $\mathsf{dis}(w_1, w_1')_{ed} \leq \tau_1$ and that I is the smallest set of $2\tau_1$ insertions and deletions which transforms w into w'. It is easy to see that each character deletion or insertion affects at most c shingles, and thus the symmetric difference between $\mathsf{SH}_c(w_1)$ and $\mathsf{SH}_c(w_2) \leq 2c\tau_1$, which implies that $\mathsf{dis}(\mathsf{SH}_c(w_1), \mathsf{SH}_c(w_2))_{sd} \leq c\tau_1$, as needed.

Now, assume w_1 is any string. Define $g_c(w_1)$ as follows. One computes $\mathsf{SH}_c(w_1)$, and stores n resulting shingles in lexicographic order $h_1 \ldots h_n$. Next, one naturally partitions w_1 into n/c disjoint shingles of length c, call them $k_1 \ldots k_{n/c}$. Next, for $1 \leq j \leq n/c$, one sets $p_c(j)$ to be the index $i \in \{1 \ldots n\}$ such that $k_j = h_i$. Namely, it tells the index of the j-th disjoint shingle of w_1 in the ordered n-set $\mathsf{SH}_c(w_1)$. Finally, one sets $g_c(w_1) = (p_c(1) \ldots p_c(n/c))$. Notice, the length of $g_c(w_1)$ is $\frac{n}{c} \cdot \log_2 n$, and also that w_1 can be completely recovered from $\mathsf{SH}_c(w_1)$ and $g_c(w_1)$.

Now, assume W_1 is any distribution of min-entropy at least t_1 on $\mathsf{Edit}(n)$. Since $g_c(W)$ has length $(n \log_2 n/c)$, its min-entropy is at most this much as well. But since min-entropy of W_1 drops to 0 when given $\mathsf{SH}_c(W_1)$ and $g_c(W_1)$, it means that the min-entropy of $\mathsf{SH}_c(W_1)$ must be at least $t_2 \geq t_1 - (n \log_2 n)/c$, as claimed. □

We can now optimize the value c. By either Lemma 3.8 or Theorem 3.10, for arbitrary universe size (in our case 2^c) and distance threshold $\tau_2 = c\tau_1$, we can construct a

secure sketch for the set difference metric with min-entropy loss $2\tau_2 \log_2(2^c) = 2\tau_1 c^2$, which leaves us total min-entropy $t_2' = t_2 - 2\tau_1 c^2 \geq t_1 - \frac{n \log n}{c} - 2\tau_1 c^2$. Applying further Lemma 3.1, we can convert it into a fuzzy extractor over $\mathsf{SDif}(2^c, n)$ for the min-entropy level t_2 with error ϵ, which can extract at least $\ell = t_2' - 2\log\left(\frac{1}{\epsilon}\right) \geq t_1 - \frac{n \log n}{c} - 2\tau_1 c^2 - 2\log\left(\frac{1}{\epsilon}\right)$ bits, while still correcting $\tau_2 = c\tau_1$ of errors in $\mathsf{SDif}(2^c, n)$. We can now apply Lemma 3.3 to get an $(\mathsf{Edit}(n), t_1, t_1 - \frac{n \log n}{c} - 2\tau_1 c^2 - 2\log\left(\frac{1}{\epsilon}\right), \tau_1, \epsilon)$-fuzzy extractor. Let us now optimize for the value of $c \geq \log_2 n$. We can set $\frac{n \log n}{c} = 2\tau_1 c^2$, which gives $c = (\frac{n \log n}{2\tau_1})^{1/3}$. We get $\ell = t_1 - (2\tau_1 n^2 \log^2 n)^{1/3} - 2\log\left(\frac{1}{\epsilon}\right)$ and therefore

Theorem 3.13. *There is an efficient* $(\mathsf{Edit}(n), t_1, t_1 - (2\tau_1 n^2 \log^2 n)^{1/3} - 2\log\left(\frac{1}{\epsilon}\right), \tau_1, \epsilon)$ *fuzzy extractor. Setting* $\tau_1 = t_1^3/(16n^2 \log^2 n)$*, we get an efficient* $(\mathsf{Edit}(n), t_1, \frac{t_1}{2} - 2\log\left(\frac{1}{\epsilon}\right), \frac{t_1^3}{16n^2 \log^2 n}, \epsilon)$ *fuzzy extractor. In particular, if* $t_1 = \Omega(n)$*, one can extract* $\Omega(n)$ *bits while tolerating* $\Omega(n/\log^2 n)$ *insertions and deletions.*

3.7 Alternate Error Models and List-Decoding

The error model considered so far in this work is very restrictive: we required that secure sketches and fuzzy extractors accept *any* secret w' within distance d of the original input w.

This model is simple to reason about and work with but seems too stringent for many applications. If the data in question is an iris scan, there is no reason to assume that the error would be adversarial: if the adversary could control the reader, she could simply learn the original iris scan completely and be done with it.

In this section, we consider a several more realistic error models and show how the constructions of the previous sections can be tweaked to gain greater error-correcting in each model.

Random Errors There is a simple, *known* distribution on the errors which occur in the data. For the Hamming metric, the most common model is the binary symmetric channel BSC_p: each bit of the input is flipped with probability p and left untouched with probability $1 - p$.

Data-only-dependent Errors The errors are adversarial, bounded to a maximum magnitude of d, but *depend only on* w, and not, for example, on the particular value of a secure sketch $\mathsf{SS}(w)$.

Computationally-bounded Errors The errors are adversarial and may depend on both w and the publicly stored information (e.g. $\mathsf{SS}(w)$). However, there is a probabilistic circuit of polynomial size which computes w' from w. The adversary cannot, for example, forge a digital signature and base the error pattern on the signature. (We'd like to thank Chris Peikert for pointing out this model to us.)

Hamming Metric Each of these three models has been studied in literature on error-correcting codes for the Hamming metric. The random and computationally-bounded error models both make obvious sense in the coding context [74, 59]. The second model doesn't immediately make sense in a coding situation, since there is no data other than the transmitted codeword on which errors could depend. The model can be made logical by allowing the sender and receiver to share either (1) common, secret random coins (see [51] and references therein) or (2) a side channel with which they can communicate a small number of noise-free, secret bits [42].

Existing results on these three models for the Hamming metric can be transported directly to our context using the *code-offset* construction(Eqn. 3.1, page 41):

$$\mathsf{SS}(W; X) = W \oplus C(X)\,.$$

Roughly, any code which corrects errors in the models above will lead to a secure sketch (resp. fuzzy extractor) which corrects errors in the model.

3.7.1 Example: Random Errors in the Hamming Metric

The random error model was famously considered by Shannon [74]. He showed that for any discrete, memoryless channel, the rate at which information can be reliably transmitted is characterized by the maximum mutual information between the inputs and outputs of the channel. For the binary symmetric channel with crossover probability p, this means that there exist codes encoding k bits into n bits, tolerating errror probability p in each bit if and only if

$$\frac{k}{n} < 1 - h(p) - \delta(n)$$

where $h(p) = -p \log p - (1-p)\log(1-p)$ and $\delta(n) = o(1)$. Computationally efficient codes achieving this bound were found later, largely due to pioneering work of Forney [37]. Thus, we can state the following:

Proposition 3.14. *For any error rate $0 < p < 1/2$ and constant $\delta > 0$, for large enough n there exist secure sketches with entropy loss $(h(p)+\delta)n$, which correct error rate of p in the data with high probability.*

The quantity $h(p)$ is less than 1 for any p in the range $(0, 1/2)$. In particular, one can get non-trivial secure sketches even for a very high error rate $p < 1/2$. Note that several other works on biometric cryptosystems consider the model of randomized errors and obtain similar results, though they are only stated for the uniform distribution on inputs [82, 22].

3.7.2 Improved Error-Correction via List Decoding

As mentioned above, the results on codes for other error models in the Hamming space [42, 51, 59] extend easily to secure sketches. Several of those constructions

work by transformations *list decodable* codes, defined below, into uniquely decodable codes for a particular error model.

As discussed below, the transformations of [42, 51, 59] can also be used in the setting of secure sketches, leading to better bounds on the number of tolerated errors in the second and third models above (errors which depend only on the data, or errors which are introduced by a computationally bounded adversary).

A code C in a metric space $\mathcal{M}$ is called *list-decodable* with list size L and distance τ if for every point $x \in \mathcal{M}$, there are at most L codewords within distance τ of $\mathcal{M}$. A list-decoding algorithm takes as input a word x and returns the corresponding list $c_1, c_2, ...$ of codewords. The most interesting setting is when L is a small polynomial (in the description size $\log |\mathcal{M}|$), and there exists an efficient list-decoding algorithm. It is then feasible for an algorithm to go over each word in the list and accept if it has some desirable property. There are many examples of such codes for the Hamming space; for a survey see Guruswami's thesis [43].

Similarly, we can define a *list-decodable secure sketch* with size L and distance τ as follows: for any pair of words $w, w' \in \mathcal{M}$ at distance at most τ, the algorithm $\mathsf{Rec}(w', \mathsf{SS}(w))$ returns a list of at most L points in $\mathcal{M}$; if $\mathsf{dis}(w, w') \leq \tau$, then one of the words in the list must be w itself. The simplest way to obtain a list-decodable secure sketch is to use the code-offset construction of Section 3.4 with a list-decodable code for the Hamming space. One obtains a different example by running the modified Juels-Sudan scheme, replacing ordinary decoding of Reed-Solomon codes with list decoding. This yields a significant improvement in the number of errors tolerated at the price of returning a list of possible candidates for the original secret.

Sieving the List Given a list-decodable secure sketch SS, all that's needed is to store some additional information which allows the receiver to disambiguate w from the list. Let's suggestively name the additional information $Tag(w)$. We describe two transformations inspired by [42, 51, 59].

1. If the errors in the data w' do not depend on the value of $Tag(w)$, then one can store $Tag(w) = I, h_I(w)$ where $\{h_i\}_{i \in \mathcal{I}}$ comes from an XOR-universal hash family from $\mathcal{M}$ to $\{0,1\}^\ell$, where $\ell = \log\left(\frac{1}{\epsilon}\right) + \log L$ and ϵ is the probability of an incorrect decoding.

   The proof is simple: the values $w_1, ..., w_L$ do not depend on I, and so for any value $w_i \neq w$, the probability that $h_I(w_i) = h_I(w)$ is $2^{-\ell}$. There are at most L possible candidates, and so the probability that any one of the elements in the list is accepted is at most $L \cdot 2^{-\ell} = \epsilon$

   The additional entropy loss incurred is at most $\ell = \log\left(\frac{1}{\epsilon}\right) + \log(L)$.

2. If the corrupted word w' depends on *both* w and $\mathsf{SS}(w)$, but w' is computed by a polynomial time circuit, then one can store $Tag = \mathsf{hash}(w)$, where w is drawn from a collision-resistant function family. The adversary succeeds only if he can find a value $w_i \neq w$ such that $\mathsf{hash}(w_i) = \mathsf{hash}(w)$, that is only by finding a collision for the hash function.

The additional entropy loss of this scheme is equal to the output length of the hash function. If we think of ϵ as the probability of a decoding error, then for standard assumptions on hash functions this loss will be polynomial in $\log(1/\epsilon)$.

3.8 Application: Password Authentication

One of the key applications discussed in the Introduction was password authentication—a server would like to store information which allows it to verify the identity of a user, but wants to ensure that if the stored information is ever made public, the password is not leaked.

Specifically, we would like to construct a pair of (possibly randomized) functions $Store()$ and $Verify()$. Given the password w the server will compute and store $s \leftarrow Store(w)$. When a user presents a potential password w', the server accepts the user if and only if $Verify(w', s) = 1$. Informally, we would like to guarantee two conditions:

1. (Completeness) If $s \leftarrow Store(w)$ is computed correctly, then for any w' within distance d of w, running $Verify(w', s)$ returns "accept" with high probability.

2. (Soundness) The adversary succeeds at breaking the scheme if, given only $s = Store(w)$, she can produce a value $\tilde{w}'$ such that $Verify(s, \tilde{w}') = 1$. This event should happen with probability at most ϵ, where the probability is over w, the randomness used in computing s, and the randomness used by the adversary.

   Such a statement can only hold for computationally bounded adversaries, since given enough time one can always try all possible words w until the right one is found.

Suppose we're given a cryptographic hash function $\mathsf{hash}()$ from n bits to ℓ bits. Given the discussion in the introduction, the natural solution is to store a secure sketch of w as well as the hash. Let SS be a secure sketch.

$$Store(w) = \mathsf{SS}(w), \mathsf{hash}(w) \tag{3.3}$$

This solution satisfies the first condition above (completeness): given w' close to w, the verification algorithm uses $\mathsf{SS}(w)$ to recover w and then checks that the hash value yields what it should.

Ensuring soundness is more delicate. If the adversary finds a string $\tilde{w}'$ which passes verification, then the adversary has in fact found string $\tilde{w} = \mathsf{Rec}(\tilde{w}', \mathsf{SS}(w))$ which hashes to the same value as the original w. Depending on the exact assumptions on the function $\mathsf{hash}()$, this may or may not yield a contradiction.

We consider three different "strengths" of the assumption on $\mathsf{hash}()$. Assume that the conditional entropy $\tilde{\mathbf{H}}_\infty(W|\mathsf{SS}(W)) = m'$.

- If we assume that $\mathsf{hash}()$ acts like a completely random function (the "random oracle" model), then the natural solution always works. If the output length

of the random oracle is ℓ, then the adversary's probability of breaking the soundness condition is $O((2^{-m'} + 2^{-\ell}) \cdot T)$, where T is number of queries made to the function $\mathsf{hash}()$ by the adversary.

(In particular, if the adversary is polynomial time, and m and ℓ are large, say $\sqrt{n}$, then the adversary's chance of succeeding is $O(poly(n) \cdot 2^{-\sqrt{n}})$, which is negligible for large values of n.)

- If we assume that $\mathsf{hash}()$ was drawn from a collision-resistant function family, then the solution in Eqn. 3.3 can be proven secure as long as the output of $\mathsf{hash}()$ function is sufficiently short.

  If the output length ℓ is at most $m' - \log\left(\frac{1}{\epsilon}\right) + 1$, and the adversary's probability of finding collisions for $\mathsf{hash}()$ is bounded above by δ,[4] then the probability that the adversary with similar computation time can break the soundness is $O(\epsilon+\delta)$. No restriction on the input length of $\mathsf{hash}()$ is necessary, except that it must be at least n to take w as input.

- Finally, if we only assume that $\mathsf{hash}()$ is a one-way function, then we actually have to modify the scheme for the proof to through, since the definition of a one-way function only gives a guarantee when the input to the function to be uniformly distributed. Let $h_i : \{0,1\}^n \to \{0,1\}^N$ a family of XOR-universal hash functions. Consider the storage function:

$$Store(w) = \mathsf{SS}(w), I, \mathsf{hash}(h_I(w)) \tag{3.4}$$

  If the adversary's chance of inverting the function $\mathsf{hash}()$ on a random input is δ, and the output length ℓ of $\mathsf{hash}()$ is at most $m' - 2\log\left(\frac{1}{\epsilon}\right) + 1$, then an adversary with similar computation time can break the soundness condition of with probability $O(\epsilon + \delta)$. No restriction on the input length of $\mathsf{hash}()$ is necessary, except that it must be at least n to take w as input.

We only explain the last of these three statements in detail, since it is the most complicated. We leave the first two as exercises for the interested reader.

Assume that $\mathsf{hash} : \{0,1\}^n \to \{0,1\}^\ell$ is a (strongly) one-way function. Since we are interested in security against polynomial-time adversaries, we should, strictly speaking, talk about *families* of one-way functions (for infinitely many input lengths) and also about families of authentication storage schemes. However, for simplicity, we will avoid this discussion here and assume that an adversary with some particular running time T (say $T = n^{\log n}$ to capture polynomial-time adversaries) cannot find a pre-image of a randomly chosen image with probability better than δ. That is,

For all probabilistic circuits $\mathcal{A}'$ of size at most T,

$$\Pr[u \leftarrow \{0,1\}^n,\ y = \mathsf{hash}(u),\ u' \leftarrow \mathcal{A}'(y) : \quad \mathsf{hash}(u') = y] \leq \delta \tag{3.5}$$

[4]There is also an implicit lower bound on ℓ. If an adversary's probability of finding a collision is at most δ, then the output length of the function must be at least $\log\left(\frac{1}{\delta}\right)$ since otherwise a simple sampling attack would succeed with probability better than δ.

In order to reduce the soundness of the password authentication scheme to the one-way-ness of $\mathsf{hash}()$, suppose there is an adversary circuit $\mathcal{A}$ which can break the soundness condition. That is, given the tuple $Store(w) = \langle \mathsf{SS}(w), I, \mathsf{hash}(h_I(w)) \rangle$, the circuit $\mathcal{A}$ produces $\tilde{w}'$ which passes verification. It is sufficient to prove that $Store(w)$ is indistinguishable from a tuple $\mathsf{SS}(w), I, \mathsf{hash}(u)$, where u is selected randomly from $\{0,1\}^n$, independently of w.

This doesn't follow immediately from the left-over hash lemma 2.2, since the functions $\mathsf{hash}(h_i(\cdot))$ are not really strong enough (the distribution on the outputs of $\mathsf{hash}()$ may not be uniformly random). However, in the appendix we prove a variant of the leftover hash lemma which does cover such functions (Lemma A.2). We obtain:

Corollary 3.15 (from Lemma A.2). *If* $\ell < \tilde{\mathbf{H}}_\infty(W|\mathsf{SS}(W)) - 2\log\left(\frac{1}{\epsilon}\right) - 1$, *then*

$$Store(W) \approx_\epsilon \langle \mathsf{SS}(W), I, \mathsf{hash}(U_n) \rangle$$

where $U_n \leftarrow \{0,1\}^N$ *is independent of* W.

To complete the proof, we'll make an additiional assumption: sampling from the distribution W should be polynomial time (this assumption isn't strictly necessary, but makes the reduction uniform).

Given an circuit $\mathcal{A}$ which breaks the soundness condition, we build $\mathcal{A}'$ for inverting $\mathsf{hash}()$ as follows: on input y, sample $w \leftarrow W$, sample $i \leftarrow \mathcal{I}$, compute $\mathsf{SS}(w)$ and output

$$\mathcal{A}'(y) = h_i(\mathsf{Rec}(\mathcal{A}(\mathsf{SS}(w), i, y)\ ,\ \mathsf{SS}(w))).$$

By the corollary above, $\mathcal{A}$ behaves almost identically on inputs $Store(w)$ and the tuple $\mathsf{SS}(w), I, \mathsf{hash}(U_n)$. If $\mathcal{A}$ succeeds at breaking soundness with probability β, then with probability $\beta - \epsilon$ (since the inputs are within statistical difference ϵ), the algorithm $\mathcal{A}'$ will invert $\mathsf{hash}()$. This yields a contradiction if $\mathcal{A}$'s success rate is more than $\epsilon + \delta$ and the running time of $\mathcal{A}'$ is less than T. Since the reduction is polynomial time (the algorithm $\mathcal{A}$ has very little computation to do beyond running $\mathcal{A}'$), we get a contradicition when the running time of $\mathcal{A}$ is bounded above by $T - poly(n)$ for some sufficiently large polynomial. If $T = n^{\log n}$ then, for example, we can limit the running time of $\mathcal{A}$ to $T/2$.

Chapter 4

Lower Bounds from Coding

This section specifies and proves the lower bounds on secure sketches and fuzzy extractors which were discussed in the previous sections. We in fact discuss two kinds of lower bounds.

First, we address the min-entropy loss. By Proposition 4.1 (resp. 4.2), if there exists a secure sketch (resp. fuzzy extractor) with error correction radius τ and entropy loss t, then there exists a code with minimum distance $2\tau+1$ and redundancy slightly less than τ. The redundancy of a code C in a metric space $\mathcal{M}$ is $\log(frac|C|\mathcal{M})$. For a block code from k bits to n bits, the redundancy is $n-k$.

This implies that the constructions we describe for the Hamming and set difference metrics are optimal when the input distribution is uniform: both constructions have entropy loss tied to the redundancy of a code. In the code-offset construction, one can use a code with optimal redundancy. In the case of set difference, both constructions have entropy loss equal to the redundancy of BCH codes, which are optimal among codes with redundancy $o(n)$ [53].

The second set of lower bounds addresses loss of *Shannon* entropy. Separate proofs are necessary since, even though the min-entropy of W may drop significantly given $\mathsf{SS}(W)$, this does not directly imply a significant drop in Shannon entropy—the relation between the two notions of entropy is too weak (the only general relationship is $\mathbf{H}_\infty(W) \leq \mathbf{H}_{sh}(W)$. Nonetheless, Propositions 4.3 and 4.5 show that in certain cases, a good deal of Shannon information must be leaked.

Informally, the first set of bounds is "operational": bounds on min-entropy loss show that bounds on how much key material can be derived using the sorts of techniques in this work. Their main use is to understand which constructions are essentially optimal, and which ones leave room for improvement. The bounds on Shannon entropy, on the other hand, are "semantic": they show that *information* about W—in its commonly accepted mathematical formulation— is leaked by $\mathsf{SS}(W)$. We will use the second set of bounds in the next part of the thesis as intuition for the notion of entropically-secure fuzzy fingerprints.

4.1 Bounds on Loss of Min-Entropy

First, some notation: Recall that an $(\mathcal{M}, K, \tau)$ code is a subset of the metric space $\mathcal{M}$ which can *correct* τ errors (this is slightly different from the usual notation of the coding theory literature.

Let $K(\mathcal{M}, \tau)$ be the largest K for which there exists an $(\mathcal{M}, K, \tau)$-code. Given any set S of 2^t points in $\mathcal{M}$, we let $K(\mathcal{M}, \tau, S)$ be the largest K such that there exists an $(\mathcal{M}, K, \tau)$-code all of whose K points belong to S. Finally, we let $L(\mathcal{M}, \tau, t) = \log(\min_{|S|=2^m} K(n, \tau, S))$. Of course, when $t = \log|\mathcal{M}| = \log N$, we get $L(\mathcal{M}, \tau, n) = \log K(\mathcal{M}, \tau)$. The exact determination of quantities $K(\mathcal{M}, \tau)$ and $K(\mathcal{M}, \tau, S)$ form the main problem of coding theory, and is typically very hard. To the best of our knowledge, the quantity $L(\mathcal{M}, \tau, t)$ was not explicitly studied in any of three metrics that we study, and its exact determination seems very hard as well.

The two following propositions give lower bounds (one for secure sketches, the other for fuzzy extractors) which show that the constructions for the Hamming and Set Difference metrics are essentially optimal, at least when the original input distribution is uniform.

The basic idea is the same in both cases: the existence of secure sketches (resp. fuzzy extractors) with small entropy lossimplies the existence of codes with small redundancy. We can then rely on existing lower bounds on the redundancy of codes to get bounds on sketches and extractors. The case of codes over $\{0,1\}^n$ with the Hamming metric is certainly the best-studied, and the known bounds can be found in textbooks, e.g. [53].

Proposition 4.1. *The existence of $(\mathcal{M}, t, t', \tau)$ secure sketch implies that $t' \leq L(\mathcal{M}, \tau, t)$. In particular, when $t = \log N$ (i.e., when the password is truly uniform), $t' \leq \log K(\mathcal{M}, \tau)$.*

Proof. Assume SS is such secure sketch. Let S be any set of size 2^t in $\mathcal{M}$, and let W be uniform over S. Then we must have $\tilde{\mathbf{H}}_\infty(W \mid \mathsf{SS}(W)) \geq t'$. In particular, there must be some particular value v such that $\mathbf{H}_\infty(W \mid \mathsf{SS}(W) = v) \geq t'$. But this means that conditioned on $\mathsf{SS}(W) = v$, there are at least $2^{t'}$ points w in S (call this set T) which could produce $\mathsf{SS}(W) = v$. We claim that these $2^{t'}$ values of w form a code of error-correcting distance τ. Indeed, otherwise there would be a point $w' \in \mathcal{M}$ such that $\mathsf{dis}(w_0, w') \leq \tau$ and $\mathsf{dis}(w_1, w') \leq \tau$ for some $w_0, w_1 \in T$. But then we must have that $\mathsf{Rec}(w', v)$ is equal to both w_0 and w_1, which is impossible. Thus, the set T above must form an $(\mathcal{M}, 2^{t'}, \tau)$-code inside S, which means that $t' \leq \log K(\mathcal{M}, \tau, S)$. Since S was arbitrary, the bound follows. □

Proposition 4.2. *The existence of $(\mathcal{M}, t, \ell, \tau, \epsilon)$-fuzzy extractors implies that $\ell \leq L(\mathcal{M}, \tau, t) - \log(1-\epsilon)$. In particular, when $t = \log N$ (i.e., when the password is truly uniform), $\ell \leq \log K(\mathcal{M}, \tau) - \log(1-\epsilon)$.*

Proof. Assume $(\mathsf{Gen}, \mathsf{Rep})$ is such a fuzzy extractor. Let S be any set of size 2^t in $\mathcal{M}$, and let W be uniform over S. Then we must have $\mathbf{SD}\left(\langle R, P\rangle, \langle U_\ell, P\rangle\right) \leq \epsilon$. In

particular, there must be some particular value p of P such that R is ϵ-close to U_ℓ conditioned on $P = p$. In particular, this means that conditioned on $P = p$, there are at least $(1-\epsilon)2^\ell$ points $r \in \{0,1\}^\ell$ (call this set T) which could be extracted with $P = p$. Now, map every $r \in T$ to some arbitrary $w \in S$ which could have produced r with nonzero probability given $P = p$, and call this map C. We claim that C must define a code with error-correcting distance τ. Indeed, otherwise there would be a point $w' \in \mathcal{M}$ such that $\mathsf{dis}(C(r_1), w') \leq t$ and $\mathsf{dis}(C(r_2), w') \leq \tau$ for some $r_1 \neq r_2$. But then we must have that $\mathsf{Rep}(w', p)$ is equal to both r_1 and r_2, which is impossible. Thus, the map C above must form an $(\mathcal{M}, 2^{\ell+\log(1-\epsilon)}, \tau)$-code inside S, which means that $\ell \leq \log K(\mathcal{M}, \tau, S) - \log(1-\epsilon)$. Since S was arbitrary, the bound follows. $\square$

Observe that, as long as $\epsilon < 1/2$, we have $0 < -\log(1-\epsilon) < 1$, so the lowerbounds on secure sketches and fuzzy extractors differ by less than a bit. In contrast, our construction of fuzzy extractors loses an additional $2\log\left(\frac{1}{\epsilon}\right)$ bits of entropy over the corresponding secure sketch, incurred by smoothing the input distribution with a randomness extractor (see Lemma 3.1).

4.2 Bounds on Loss of Shannon Entropy

In this section, we show that secure sketches and fuzzy extractors must leak information about the input in the sense of Shannon. For concreteness, we focus on the case where data is from $\{0,1\}^n$ equipped with the Hamming (or Set Difference) metric.

The proof techniques are quite different from the previous section. The bound for secure sketches is quite simple, and holds for any metric space where the number of points within distance τ grows exponentially in τ. Both the Hamming metric and edit metric satisfy this, though we only state the proposition for the Hamming metric.

The second result, on fuzzy extractors, is the main result of this section. The proof depends on the particular geometry of $\{0,1\}^n$. Extending the results to the edit metric seems not too difficult, but requires a better understanding of the geometry of that metric space. The result on secure sketches (Proposition 4.3) may be thought of as a warm-up for the bound on fuzzy extractors (Proposition 4.5).

Proposition 4.3. *Assume* SS *is a secure sketch correcting* τ *errors and* E *is a uniform distribution over* $\{v \mid \|v\| \leq \tau\}$. *Then for any distribution* W, *we have* $\mathbf{I}(W; \mathsf{SS}(W)) \geq \mathbf{H}_{sh}(W \mid W \oplus E)$. *In particular, if* W *is uniform over* $\{0,1\}^n$, *then* $\mathbf{I}(W; \mathsf{SS}(W)) \geq \mathbf{H}_{sh}(E) \approx nh_2(\tau/n)$, *where* h_2 *is the binary entropy function.*

Proof. Let $W' = W \oplus E$, so that $\|W - W'\| \leq \tau$. For any random variables A, B, C it is easy to check the following inequality:

$$\begin{aligned} \mathbf{I}(A; B, C) &= \mathbf{I}(A; B) + \mathbf{I}(A; C) + \mathbf{I}(B; C \mid A) - \mathbf{I}(B; C) \\ &\leq \mathbf{I}(A; B) + \mathbf{I}(A; C) + \mathbf{I}(B; C \mid A). \end{aligned}$$

Set $A = W, B = W', C = \mathsf{SS}(W)$. Since $\mathsf{SS}(W)$ is independent of W' conditioned on W, we know that $\mathbf{I}(\mathsf{SS}(W); W' \mid W) = 0$. We obtain:

$$\begin{aligned}\mathbf{I}(W; W', \mathsf{SS}(W)) &\leq \mathbf{I}(W; \mathsf{SS}(W)) + \mathbf{I}(W; W') + \mathbf{I}(W'; \mathsf{SS}(W) \mid W) \\ &= \mathbf{I}(W; \mathsf{SS}(W)) + \mathbf{I}(W; W')\end{aligned}$$

On the other hand, since W' and $\mathsf{SS}(W)$ determine W, we have

$$\mathbf{I}(W; W', \mathsf{SS}(W)) = \mathbf{H}_{sh}(W) - \mathbf{H}_{sh}(W \mid W', \mathsf{SS}(W)) = \mathbf{H}_{sh}(W)$$

Combining, we get $\mathbf{I}(W; \mathsf{SS}(W)) \geq \mathbf{H}_{sh}(W) - \mathbf{I}(W; W') = \mathbf{H}_{sh}(W \mid W')$. □

A Bound for Fuzzy Extractors A more delicate argument than the one above shows that fuzzy extractors must also leak a certain amount of Shannon information about their inputs. Since we are proving a lower bound, we will restrict our attention to the uniform distribution, which is a valid min-entropy t distribution for any t.[1]

The simplest consequence to take away from the result (Proposition 4.5, below) is that as soon as the number of errors τ to be tolerated becomes large (say $\sqrt{n}$), then the public part of the fuzzy extractor leaks $\Omega(n)$ bits of information about the secret input.

The proof uses the isoperimetric inequality on the hypercube $\{0,1\}^n$ (see [11], theorem 16.6), so we first introduce some notation. Given a set $S \in \{0,1\}^n$ and a number τ, we let $Out_\tau(S) = \{y \mid \exists w \in S \text{ s.t. } \|w - y\| \leq \tau\}$ be the τ-th *shadow* of S, i.e. the set of points of distance at most τ from some point in S. Then the isoperimetric inequality states that balls have the smallest outshadows, for every τ. This allows one to lower bound $|Out_\tau(S)|$ in terms of $|S|$. Since we want to find a closed expression bounding $\mathbf{H}_{sh}(W \mid P)$ above, we will only use the following corollary of the isoperimetric inequality. Here h_2 is the binary entropy function, $h_2(p) = p\log(\frac{1}{p}) - (1-p)\log(\frac{1}{1-p})$.

Fact 4.4. *For every set $S \subset \{0,1\}^n$ such that $|Out_\tau(S)| \leq 2^{n-1}$, we have*

$$|S| \leq A_\tau \cdot |Out_\tau(S)|, \quad \textit{where} \quad A_\tau \leq \frac{\sum_{i=0}^{n/2-\tau-1} \binom{n}{i}}{2^{n-1}} \leq 2^{n(h_2(\frac{1}{2}-\frac{\tau}{n})-1)} \tag{4.1}$$

In particular, when $\tau = \Omega(\sqrt{n})$, the ratio is exponentially small, i.e. $A_\tau = 2^{-\Omega(n)}$.

Proposition 4.5. *Assume $(\mathsf{Gen}, \mathsf{Rep})$ is a $(n, t, \ell, \tau, \epsilon)$ fuzzy extractor, and let the output of the generation algorithm $\mathsf{Gen}(W)$ be P, Z, where P is the public part and Z, the extracted key. Then for the uniform distribution $W \leftarrow \{0,1\}^n$, we have*

$$\mathbf{I}(W; P) \geq \log\left(\frac{1}{A_\tau}\right) - 2^{-\ell}n - \epsilon(n+\ell) \approx n\left(1 - h_2\left(\frac{1}{2} - \frac{\tau}{n}\right)\right)$$

where α_τ is as in Fact 4.4. If $\tau = \Omega(\sqrt{n})$, $\ell = \omega(1)$ and $\epsilon = o(1)$, then we can use the bounds on A_τ to conclude P reveals $\Omega(n)$ bits of information about W.

[1] Even though our technique works for more general distributions, the particular bounds we get do not appear to be much stronger, while the exact estimates become intractable.

Since $\tilde{\mathbf{H}}_\infty(W \mid P) \leq \mathbf{H}_{sh}(W \mid P)$, the result also implies that average min-entropy of W is reduced.

Proof. Since W and P determine Z, we have

$$\mathbf{H}_{sh}(W \mid P) = \mathbf{H}_{sh}(W, Z \mid P) = \mathbf{H}_{sh}(Z \mid P) + \mathbf{H}_{sh}(W \mid Z, P).$$

We will bound each of the two last terms separately. We begin with $\mathbf{H}_{sh}(Z|P)$. Let $g(x) = -x \log x$. Recall that the Shannon entropy of a distribution with probabilities $q_1, ..., q_L$ is $\sum_i g(q_i)$. We'll use a simple approximation, which can be derived by computing the derivative of $g()$: for $\delta \geq 0$, $g(2^{-\ell} + \delta) \leq g(2^{-\ell}) + \ell\delta$.

We expect the distribution of the pair Z conditioned on most values p of P to be essentially uniform over $\{0,1\}^\ell$. In order to manipulate the small deviations from uniformity, we let

$$\delta_{p,r} = \max[\Pr(Z = r \mid P = p) - 2^{-\ell},\ 0].$$

Since $\mathbf{SD}\left(\langle Z, P\rangle, \langle U_\ell, P\rangle\right) \leq \epsilon$, we have $\sum_{p,r} \delta_{p,r} \leq \epsilon$. Now, we can upper bound $\mathbf{H}_{sh}(Z \mid P)$ as follows:

$$\begin{aligned}
\mathbf{H}_{sh}(Z \mid P) &= \sum_p \Pr(P = p)\mathbf{H}_{sh}(Z \mid P = p) \\
&= \sum_p \Pr(P = p) \sum_r g(\Pr(Z = r \mid P = p)) \leq \sum_p \Pr(P = p) \sum_r g(2^{-\ell} + \delta_{p,r}) \\
&\leq \sum_p \Pr(P = p) \sum_r \left(g(2^{-\ell}) + \ell\delta_{p,r}\right) \leq \ell + \ell \sum_p \Pr(P = p) \sum_r \delta_{p,r} \\
&\leq \ell(1 + \epsilon)
\end{aligned}$$

Next, given p and r, denote by $S_{p,r}$ the set of w for which $\mathsf{Rep}(w, p) = r$. Note that

$$\mathbf{H}_{sh}(W|P = p, Z = r) \leq \log |S_{p,r}|.$$

We wish to bound the size of the sets $S_{p,r}$. To do so, let $T_{p,r} = Out_\tau(S_{p,r})$ be the τ-th shadow of $S_{p.r}$. For any $r_0 \neq r_1$, their τ-th shadows must be disjoint (Why? If $w' \in T_{p,r_0} \cap T_{p,r_1}$, then error correction property of fuzzy extractors would imply that $\mathsf{Rep}(w', p)$ is equal to both r_0 and r_1, which is impossible.) This allows us to use the following lemma:

Claim 4.6. *For any 2^ℓ subsets $S_1, ..., S_{2^\ell}$ of $\{0,1\}^n$, if the τ-th shadows $Out_\tau(S_i)$ are mutually disjoint, then the product of the sizes is bounded above:*

$$\log(\prod_i |S_i|) \leq n + 2^\ell(\log(A_\tau) + n - \ell) \tag{4.2}$$

We will prove the claim below. For now, we can bound $\mathbf{H}_{sh}(W \mid P, Z)$:

$$\begin{aligned}
&\mathbf{H}_{sh}(W \mid P, Z) \\
&= \sum_p \Pr(P = p) \sum_r \Pr(Z = r \mid P = p) \mathbf{H}_{sh}(W \mid P = p, Z = r) \\
&\leq \sum_p \Pr(P = p) \sum_r (2^{-\ell} + \delta_{p,r}) \log |S_{p,r}| \\
&\leq n \sum_p \Pr(P = p)(\sum_r \delta_{p,r}) + 2^{-\ell} \sum_p \Pr(P = p) \log \left(\prod_r |S_{p,r}| \right)
\end{aligned}$$

The first of the terms in the last equation is at most ϵ, since the probabilities $\Pr[P = p]$ are each bounded by 1, and the sum $\sum_{p,r} \delta_{p,r}$ is at most ϵ. To bound the second term, we can apply the claim, once for each value of p, to the collection $\{S_{p,r}\}_{r \in \{0,1\}^\ell}$:

$$\begin{aligned}
\mathbf{H}_{sh}(W|P, Z) &\leq \epsilon n + 2^{-\ell} \sum_p \Pr[P = p](n + 2^\ell(\log(A_\tau) + n - \ell)) \\
&= n - \ell + \log(A_\tau) + n(2^{-\ell} + \epsilon)
\end{aligned}$$

Combining the bounds for $\mathbf{H}_{sh}(Z \mid P)$ and $\mathbf{H}_{sh}(W \mid P, Z)$, and replacing n with $\mathbf{H}_{sh}(W)$, completes the proof. We get $\mathbf{H}_{sh}(W|P) \leq \mathbf{H}_{sh}(W) + \log(A_\tau) + n2^{-\ell} + \epsilon(n+\ell)$, which implies the main statement. □

Proof of Claim 4.6. By hypothesis, the τ-th shadows $Out_\tau(S_i)$ are all disjoint, and hence at most one of them can have more than 2^{n-1} points. For all the remaining sets, we have $|S_i| \leq A_\tau |Out_\tau(S_i)|$. For the one "exceptional" set $_{i*}$ of large size, we can bound $\log |S_{i*}|$ by n. Thus

$$\log(\prod_{i=1}^{2^\ell} |S_i|) \leq n + \log(A_\tau^{2^\ell} \prod_i |Out_\tau(S_i)|) = n + 2^\ell \log(A_\tau) + \log(\prod_i |Out_\tau(S_i)|).$$

The sets $Out_\tau(S_i)$ are all disjoint, and so their sizes sum to at most 2^n. If one has 2^ℓ numbers a_i whose sum is less than 2^n, their product is maximized by setting all a_i to $2^{n-\ell}$. This gives us $\log(\prod_{i=1}^{2^\ell} |S_i|) \leq n + 2^\ell \log(A_\tau) + 2^\ell \log(2^{n-\ell})$, as desired. □

Part II

Secrecy for High-Entropy Data

Chapter 5

Entropic Security, Prediction and Indistinguishability

This chapter presents the general results on entropic security discussed in the introduction. In later chapters, we'll apply these ideas to get new results on encryption, "perfectly one-way" hash functions, and the fuzzy extractors introduced in Chap. 3.

The main contributions are:

- A new, stronger formulation of entropic security. Previous formulations [17, 18, 70] guaranteed only that no *predicate* of the secret input is leaked. We require that the adversary have no advantage at predicting any function whatsoever of the input.

- The introduction of *indistinguishability* on high-entropy distributions as a notion of security. This notion is technically much easier to work with than entropic security, since it is very close to randomness extraction and can be addressed with similar tools.

- The equivalence of indistinguishability to entropic security. There are actually two main pieces of this result: first, that indistinguishability is equivalent to entropic security for predicates (the definition of [17, 18, 70]) and second, that the adversary's ability to predict some function of the input implies the ability to predict a predicate. These equivalences are inspired by the original equivalence of semantic security and indistinguishability of encryptions [40], though the proof techniques are quite different.

The next section defines entropic security (in two variants), and a notion of indistinguishability. The main result of the section is the equivalence of these notions (Theorem 5.1). Our presentation refers several times to the parallel with a similar equivalence due to Goldwasser and Micali [40]—see Section 1.2.3 for a brief explanation of that result. The remainder of the chapter gives the details of the proof of the equivalence.

5.1 Entropic Security, Prediction of Functions and Indistinguishability

This section formulates the various notions of security we work with, and states the equivalences which are the main technical results of the chapter. To unify the discussion, let $Y(m;R)$ be some randomized map. Here $m \in \{0,1\}^n$ is the input and R is a string of uniformly random bits, independent of m. In the case of encryption, $Y = \mathcal{E}$ is the encryption function, and $R = \langle \kappa, i \rangle$ consists of the key and any extra randomness i used by the encryption. In the setting of hashing, Y will be the hash function and R the randomness used by the hash. When the random string R is implicit, we will simply write $Y(m)$ instead of $Y(m;R)$.

Recall that a predicate is a "yes"/"no" question, that is, a function that outputs a single bit. Entropic security was first formulated in terms of predicates by Canetti, Micciancio and Reingold [17, 18] in the context of hash functions and, subsequently but independently, by Russell and Wang [70] in the context of encryption schemes.

Definition 5.1 ([17, 18, 70]). *The probabilistic map Y hides all **predicates** of X with leakage ϵ if for every (randomized) adversary $\mathcal{A}$, there exists some random variable $G_{\mathcal{A}}$ on $\{0,1\}$, independent of X, such that for all predicates $g : \{0,1\}^n \to \{0,1\}$,*

$$\left| \Pr[\mathcal{A}(Y(X)) = g(X)] - \Pr[G_{\mathcal{A}} = g(X)] \right| \leq \epsilon.$$

*The map $Y()$ is called (t,ϵ)-*entropically secure for **predicates** *if $Y()$ hides all predicates of X, whenever the min-entropy of X is at least t.*

This definition may not seem quite satisfying from several points of view. First, it states only that no predicate of the input is leaked, and provides no explicit guarantees about other functions. In contrast, the original semantic security definition of Goldwasser and Micali held for all functions, not only predicates. Second, there is no guarantee that the distribution for $G_{\mathcal{A}}$ is easy to compute or sample from in the case where, say, $\mathcal{A}$ runs polynomial time and M is samplable in polynomial time. Finally, the definition is somewhat hard to work with.

We introduce two new definitions which we prove are equivalent to entropic security for predicates. First, we show that Definition 5.1 can be extended to hold for *all functions*, not only predicates. This is the definition discussed in the introduction.

Definition 5.2 (Entropic Security, same as Definition 1.1). *The probabilistic map Y hides all functions of X with leakage ϵ if for every adversary $\mathcal{A}$, there exists some adversary $\mathcal{A}'$ with no access to X such that for all functions $f : \{0,1\}^n \to \{0,1\}^*$,*

$$\left| \Pr[\mathcal{A}(Y(X)) = f(X)] - \Pr[A'() = f(X)] \right| \leq \epsilon.$$

The map $Y()$ is called (t,ϵ)-entropically secure if $Y()$ hides all functions of X, whenever the min-entropy of X is at least t.

Second, we show that the entropic security of $Y()$ is equivalent to the indistinguishability of $Y()$'s outputs on certain pairs of distributions on the inputs. This notion is inspired by that of Goldwasser and Micali [40], which required that the output of an encryption scheme on any pair of inputs (as opposed to pair of distributions over inputs) be indistinguishable by polynomial-time adversaries. One nice consequence of this equivalence is that in Definition 5.1 we can take $G_{\mathcal{A}} = \mathcal{A}(E(U))$, where U is the uniform distribution on $\{0,1\}^n$. This definition is also much easier to work with, as we will see in later sections.

Definition 5.3. *A randomized map $Y()$ is* (t,ϵ)-indistinguishable *if there is a random variable G such that for every distribution on messages M over $\{0,1\}^n$ with min-entropy at least t, we have*

$$\mathbf{SD}\left(Y(M),\ G\right) \leq \epsilon.$$

There are two main intuitions behind this part of the thesis. First is that indistinguishability is related to *extraction* of randomness from distributions of high min-entropy. We will return to this in the construction and lower bounds. The second intuition is that entropic security and indistinguishability are related. This leads to the main result of the chapter:

Theorem 5.1. *Let Y be a randomized map with inputs of length n. Then*

1. (t,ϵ)*-entropic security for predicates implies* $(t-1, 4\epsilon)$*-indistinguishability.*

2. $(t-2,\epsilon)$*-indistinguishability implies* $(t, \epsilon/8)$*-entropic security for* ***all functions*** *when* $t \geq 2\log\left(\frac{1}{\epsilon}\right) + 1$.

Entropic security with respect to predicates is trivially implied by entropic security for all functions, and so Theorem 5.1 states that all three notions of security discussed above are equivalent up to small changes in the parameters.

Randomness Extraction and Entropic Security Taking the distribution G in Definition 5.3 to be the uniform distribution, then we recover the definition of randomness extraction—for any input distribution of high-enough entropy, the output is very close to uniform. Thus, Theorem 5.1 implies that an extractor for t-sources hides all partial information about sources of min-entropy at least $t+2$.

5.1.1 Proving Theorem 5.1

The remainder of this section gives an overview of the proof of Theorem 5.1.

First, some notation. Fix a distribution X on $\{0,1\}^n$. For a function $f : \{0,1\}^n \to \{0,1\}^*$, let $\mathsf{pred}_{f,X}$ be the maximum probability of any particular outcome, that is the maximum probability of predicting $f(X)$ without having any information about X:

$$\mathsf{pred}_{f,X} = \max_z \Pr[f(X) = z]$$

(When X is clear from the context, we may simply write pred_f.) We may rephrase entropic security as follows: for every function f and adversary $\mathcal{A}$, the probability of predicting $f(X)$ given $Y(X)$ is at most $\mathsf{pred}_f + \epsilon$:

$$\Pr[\mathcal{A}(Y(X)) = f(X)] \leq \mathsf{pred}_{f,X} + \epsilon$$

From Entropic Security to Indistinguishability The first statement of Theorem 5.1 is the easier of the two to prove, and we give the intuition here: given two distributions X_0, X_1, we can define a predicate $g(x)$ which captures the question "is x more likely to have come from X_0 or X_1?" If X is a equal mixture of X_0 and X_1, then the adversary which makes the maximum likelihood guess at $g(X)$ given $Y(X)$ will have success probability $\frac{1}{2} + \frac{1}{2}\mathbf{SD}\left(Y(X_0), Y(X_1)\right)$. On the other hand, with no access to $Y(X)$, the adversary can succeed with probability at most $\mathsf{pred}_P = \frac{1}{2}$. Entropic security implies that the advantage over random guessing, and hence the statistical distance, must be small. The formal proof is more involved, and is given in Section 5.2.

From Indistinguishability to Entropic Security Proving that indistinguishability implies entropic security is considerably more delicate. Although the statement is a high-entropy version of the equivalence between semantic security and indistinguishability of encryptions due to Goldwasser and Micali [40], the proof techniques are quite different and so we begin with an overview of the main ideas and notation.

The Case of Balanced Predicates We say a function f is *balanced* (w.r.t. X) if it takes on all its possible values with equal probability, i.e. there are $\frac{1}{\mathsf{pred}_f}$ possible values and each occurs with probability pred_f. The reductions we consider are much easier for balanced functions—most of the effort will be in reducing unbalanced functions to balanced ones without losing too much in the prediction probability.

For example, suppose that $g()$ is a balanced *predicate* for distribution X, that is $\Pr[g(X) = 0] = \Pr[g(X) = 1] = \frac{1}{2}$, and that that $\mathcal{A}$ is an adversary contradicting entropic security for min-entropy $t = \mathbf{H}_\infty(X)$, that is $\Pr[\mathcal{A}(Y(X)) = g(X)] = \frac{1}{2} + \epsilon$. For $b \in \{0,1\}$, let X_b be the distribution of X conditioned on $g(X) = b$. The adversary's advantage over random guessing in distinguishing $Y(X_0)$ from $Y(X_1)$ is ϵ. However, that same advantage is also a lower bound for the statistical difference. We get:

$$\begin{aligned}\frac{1}{2} + \epsilon &= \Pr[\mathcal{A}(Y(X)) = g(X)] \\ &= \Pr[b \leftarrow \{0,1\} : \ \mathcal{A}(Y(X_b)) = b] \leq \frac{1}{2} + \frac{1}{2}\mathbf{SD}\left(Y(X_0), Y(X_1)\right),\end{aligned}$$

and so the distance between $Y(X_0)$ and $Y(X_1)$ is at least $\epsilon/2$. To see that this contradicts indistinguishability, note that since $g(X)$ is balanced, we obtain X_0 and X_1 by conditioning on events of probability at least $\frac{1}{2}$. Probabilities are at most doubled, and so the min-entropies of both X_0 and X_1 are at most $\mathbf{H}_\infty(X) - 1$.

Balancing Predicates If the predicate $g()$ is not balanced on X, then the previous strategy yields a poor reduction. For example, $\Pr[g(X) = 0]$ may be very small (potentially as small as ϵ). The probabilities in the distribution X_0 would then be a factor of $1/\epsilon$ bigger than their original values, leadding to a loss of min-entropy of $\log(1/\epsilon)$. This argument therefore proves a weak version of Theorem 5.1: (t, ϵ) indistinguishability implies $(t + \log\left(\frac{1}{\epsilon}\right), 2\epsilon)$ entropic security for *predicates*.

This entropy loss is not necessary. We give a better reduction in Section 5.2. The idea is that to change the predicate $g()$ into a balanced predicate by flipping the value of the predicate on points on which the original adversary $\mathcal{A}$ performed poorly. By greedily choosing a set of points in $g^{-1}(0)$ of the right size, we show that there exists a balanced predicate $g'()$ on which the same adversary as before has advantage at least $\epsilon/2$, if the adversary had advantage ϵ for the original predicate.

From Predicates to Arbitary Functions In order to complete the proof of Theorem 5.1, we need to show that entropic security for predicates implies entropic security for all functions. The reduction is captured by the following lemma, which states that for every function with a good predictor (i.e. a predictor with advantage at least ϵ), there exists a predicate for which nearly the same predictor does equally well. This is the main technical result of this chapter.

The reduction uses the predictor $\mathcal{A}(Y(X))$ as a black box, and so we will simply use the random variable $A = \mathcal{A}(Y(X))$.

Lemma 5.2 (Main Lemma). *Let X be any distribution on $\{0,1\}^n$ such that $t \geq \frac{3}{2}\log\left(\frac{1}{\epsilon}\right)$, and let A be any random variable (possibly correlated to X). Suppose there exists a function $f : \{0,1\}^n \to \{0,1\}^*$ such that $\Pr[A = f(X)] \geq \mathsf{pred}_f + \epsilon$. Then there exists a* predicate *$g : \{0,1\}^n \to \{0,1\}$ and an algorithm $B(\cdot)$ such that*

$$\Pr[B(A) = g(X)] \geq \mathsf{pred}_g + \epsilon/4.$$

There are two main steps to proving the lemma:

- If A is a good predictor for an (arbitrary) function $f(\cdot)$, then there is a (almost) *balanced* function $f'(\cdot)$ and a good predictor A's of the form $g(A)$.
- If $f(\cdot)$ is a balanced function (or almost balanced) and A is a good predictor for $f(X)$, then there is a predicate $g(\cdot)$ of the form $g'(f(\cdot))$ such that $g'(A)$ is a good predictor for $g(X)$.

The proof itself is in Section 5.3.2.

A More Efficient Reduction Lemma 5.2 says nothing about the running time of $B(\cdot)$—in general, the reduction may yield a large circuit. Nonetheless, we may indeed obtain a polynomial-time reduction for certain functions f. If no value of f occurs with probability more than ϵ^2, then inner product with a random vector provides a good predicate.

Proposition 5.3. *Let X be any random variable distributed in $\{0,1\}^n$. Let $f : \{0,1\}^n \to \{0,1\}^N$ be a function such that $\mathsf{pred}_{f,X} \leq \epsilon^2/4$, and let A be a random variable with advantage ϵ at guessing $f(X)$. For $r \in \{0,1\}^N$, let $g_r(x) = r \odot f(x)$. If r is drawn uniformly from $\{0,1\}^N$, then*

$$\mathbb{E}_r \left[\Pr[r \odot A = g_r(X)] - \mathsf{pred}_{g_r} \right] \geq \epsilon/4.$$

In particular, there exists a value r and a $O(N)$-time algorithm B such that

$$\Pr[B(A) = g_r(X)] \geq \mathsf{pred}_{g_r} + \epsilon/4.$$

We prove Proposition 5.3 in Section 5.3.2, and use it as motivation for the proof of Lemma 5.2.

5.2 From Entropic Security to Indistinguishability

Lemma 5.4. *(t,ϵ)-entropic security for predicates implies $(t-1, 4\epsilon)$-indistinguishability.*

Proof. It is sufficient to prove indistinguishability for all distributions which are uniform on some set of 2^{t-1} points. To see why, recall that any distribution of min-entropy at least $t-1$ can be written as a convex combination of such *flat* distributions. If $X_0 = \sum \lambda_{0,i} X_{0,i}$ and $X_1 = \sum_j \lambda_{1,j} X_{1,j}$, where the $X_{0,i}$ and $X_{1,j}$ are all flat distributions, then the statistical distance $\mathbf{SD}\left(Y(X_0), Y(X_1)\right)$ is bounded above by $\sum_{i,j} \lambda_{0,i}\lambda_{1,j} \mathbf{SD}\left(Y(X_{0,i}), Y(X_{1,j})\right)$ (by the triangle inequality). If each of the pairs $Y(X_{0,i}), Y(X_{1,j})$ has distance at most ϵ, then the entire sum will be bounded by ϵ.

Now let X_0, X_1 be any two flat distributions over *disjoint* sets of 2^{t-1} points each (we will deal with non-disjoint sets below), and let X be an equal mixture of the two. That is, to sample from X, flip a fair coin B, and sample from X_B. Take g to be any predicate which is 0 for any sample from X_0 and 1 for any sample from X_1. A good predictor for g will be the adversary $\mathcal{A}$ who, given a string y as input, guesses as follows:

$$\mathcal{A}(y) = \begin{cases} 0 & \text{if } y \text{ is more likely under the distribution } Y(X_0) \text{ than under } Y(X_1) \\ 1 & \text{otherwise} \end{cases}$$

By the definition of statistical difference (Section 2.1), this adversary guesses the predicate with probability exactly:

$$\Pr\left[\mathcal{A}(Y(X)) = B = g(X)\right] = \tfrac{1}{2} + \tfrac{1}{2}\mathbf{SD}\left(Y(X_0), Y(X_1)\right). \tag{5.1}$$

We can now apply the assumption that $Y()$ is (t,ϵ)-entropically secure to bound $\mathbf{SD}\left(Y(X_0), Y(X_1)\right)$. First, for any random variable G over $\{0,1\}$ which is independent of X, the probability that $G = g(X)$ is exactly $\frac{1}{2}$. Now the distribution X has

min-entropy t by construction, and so by entropic security the probability that $\mathcal{A}(y)$ can guess $g(X)$ is bounded:

$$\Pr[\mathcal{A}(Y(X)) = g(X)] \leq \max_G \{\Pr[G = g(X)]\} + \epsilon = \tfrac{1}{2} + \epsilon. \tag{5.2}$$

Combining the last two equations, the statistical difference $\mathbf{SD}\left(Y(X_0), Y(X_1)\right)$ is at most 2ϵ. This takes care of the case where X_0 and X_1 have disjoint supports.

To get the general indistinguishability condition, fix any $\tilde{X}_0$ as above (flat on 2^{t-1} points). For any other flat distribution $\tilde{X}_1$, there is some third flat distribution X' which is disjoint from both $\tilde{X}_0$ and $\tilde{X}_1$. By the previous reasoning, both $\mathbf{SD}\left(Y(\tilde{X}_0), Y(X')\right)$ and $\mathbf{SD}\left(Y(X'), Y(\tilde{X}_1)\right)$ are less than 2ϵ. By the triangle inequality $\mathbf{SD}\left(Y(X_0), Y(X_1)\right) \leq 4\epsilon$ (a more careful proof avoids the triangle inequality and gives distance 2ϵ even when the supports of X_0, X_1 overlap. □

5.3 From Indistinguishability to Entropic Security

5.3.1 Entropic Security for Predicates

Lemma 5.5. *$(t-2, 2\epsilon)$-indistinguishability implies (t, ϵ)-entropic security for **predicates** for $t \geq 2$.*

Proof. Suppose that the scheme is not (t, ϵ)-entropically secure. That is, there is a message distribution X with min-entropy at least t, a predicate g and an adversary $\mathcal{A}$ such that

$$\Pr[\mathcal{A}(Y(X)) = g(X)] > \epsilon + \max_{i=0,1} \{\Pr[g(X) = i]\} \tag{5.3}$$

We wish to choose two distributions of min-entropy $t-2$ and use the adversary to distinguish them, thus contradicting indistinguishability. It's tempting to choose the sets $g^{-1}(0)$ and $g^{-1}(1)$, since we know the adversary can predict g reasonably well. That attempt fails because one of the pre-images $g^{-1}(0), g^{-1}(1)$ might be quite small, leading to distributions of low min-entropy. Instead, we partition the support of X into sets of (almost) equal measure, making sure that the smaller of $g^{-1}(0)$ and $g^{-1}(1)$ is entirely contained in one partition.

Now let:

$$\begin{aligned} p &= \Pr[h(X) = 1] \\ q_0 &= \Pr[\mathcal{A}(Y(X)) = 1 | g(X) = 0] \\ q_1 &= \Pr[\mathcal{A}(Y(X)) = 1 | g(X) = 1] \end{aligned}$$

Suppose without loss of generality that $p \geq 1/2$, i.e. that $g(X) = 1$ is more likely than, or as likely as, $g(X) = 0$ (if $p < 1/2$, we can just reverse the roles of 0 and 1). The violation of entropic security (Eq. 5.3) can be re-written:

$$pq_1 + (1-p)(1-q_0) > p + \epsilon$$

In particular, $p - pq_1 > 0$ so we get:

$$(1-p)(q_1 - q_0) > \epsilon \tag{5.4}$$

Now we wish to choose two distributions A, B, each of min-entropy $t-2$. For now, fix any set $\mathcal{S} \subseteq g^{-1}(1)$, where $g^{-1}(1) = \{m \in \{0,1\}^n | g(m) = 1\}$. We make the choice of § more specific below. Let $A_\mathcal{S}$ be the conditional distribution of X conditioned on $X \in \mathcal{S}$, and let $B_\mathcal{S}$ be distributed as X conditioned on $X \in \{0,1\}^n \setminus \mathcal{S}$. That is, $A_\mathcal{S}$ and $B_\mathcal{S}$ have disjoint supports and the support of $B_\mathcal{S}$ covers $g^{-1}(0)$ entirely.

The first property we will need from $\mathcal{S}$ is that it split the mass of X somewhat evenly. If the probability mass p' of $\mathcal{S}$ under X was exactly $1/2$, then the min-entropies of $A_\mathcal{S}$ and $B_\mathcal{S}$ would both be exactly $t-1$. Depending on the distribution X, it may not be possible to have such an even split. Nonetheless, we can certainly get $\frac{1}{2} \leq p' < \frac{1}{2} + 2^{-t}$, simply by adding points one at a time to § until it gets just below $1/2$. The order in which we add the points is not important. For $t > 2$ (which is a hypothesis of this proof), we get $\frac{1}{2} \geq p' \geq \frac{3}{4}$. Hence, we can choose $\mathcal{S}$ so that the min-entropies of $A_\mathcal{S}$ and $B_\mathcal{S}$ are both at least $t-2$.

We will also need that $\mathcal{S}$ have other properties. For every point x in the support of X, we define $q_x = \Pr[\mathcal{A}(Y(x)) = 1]$. The average over $x \leftarrow X$, restricted to $g^{-1}(1)$, of q_x is exactly q_1, that is

$$\mathbb{E}_{x \leftarrow X}[q_x] = q_1$$

If we now the choose the set $\mathcal{S}$ greedily, always adding points which maximize q_x, we are guaranteed that the average over X, conditioned on $X \in \mathcal{S}$, is at least q_1. That is, there exists a choice of S with mass $p' \in [\frac{1}{2}, \frac{3}{4}]$ such that

$$\Pr[\mathcal{A}(Y(A_\mathcal{S})) = 1] = \mathbb{E}_{x \leftarrow A_\mathcal{S}}[q_x] \geq q_1.$$

We can also now compute the probability that $\mathcal{A}(Y(B_\mathcal{S}))$ is 1:

$$\Pr[\mathcal{A}(Y(B_\mathcal{S})) = 1] = \frac{1-p}{1-p'} q_0 + \frac{p-p'}{1-p'} \Pr[\mathcal{A}(Y(X)) = 1 | X \notin \mathcal{S} \text{ and } g(X) = 0]$$

Now $\Pr[\mathcal{A}(Y(X)) = 1 | X \notin \mathcal{S} \text{ and } g(X) = 0]$ is at most q_1 (since by the greedy construction of $\mathcal{S}$, this is the average over elements in $g^{-1}(1)$ with the lowest values of q_m). Using $\mathcal{A}$ as a distinguisher for the distributions $Y(A_\mathcal{S})$ and $Y(B_\mathcal{S})$, we get:

$$\left| \Pr\left[\mathcal{A}(Y(A_\mathcal{S})) = 1\right] - \Pr\left[\mathcal{A}(Y(B_\mathcal{S})) = 1\right] \right| \geq q_1 - \frac{1-p}{1-p'} q_0 - \frac{p-p'}{1-p'} q_1 = \frac{1-p}{1-p'}(q_1 - q_0)$$

Since entropic security is violated (Eq. 5.4), we have $(1-p)(q_1-q_0)/(1-p') > \epsilon/(1-p')$. By construction, we have $p' > \frac{1}{2}$ so the advantage of the predictor is at least 2ϵ, that is:

$$\mathbf{SD}\left(Y(A_\mathcal{S}), Y(B_\mathcal{S})\right) \geq \left| \Pr\left[\mathcal{A}(Y(A_\mathcal{S})) = 1\right] - \Pr\left[\mathcal{A}(Y(B_\mathcal{S})) = 1\right] \right| \geq 2\epsilon$$

Since A and B each have min-entropy at least $t-2$, this contradicts $(t-2, 2\epsilon)$-indistinguishability, completing the proof. □

5.3.2 From Predicates to General Functions

This section contains the proofs of Lemma 5.2 and Proposition 5.3. We begin with Proposition 5.3, since the proof is straightforward and provides some intuition for the proof of Lemma 5.2.

Proof of Proposition 5.3. We can calculate the expected advantage almost directly. Note that conditioned on the event $A = f(X)$, the predictor $r \odot A$ always agrees with $g_r(X)$. When $A \neq f(X)$, they agree with probability exactly $\frac{1}{2}$. Hence, we have

$$\mathbb{E}_r \left[\Pr[r \odot A = g_r(X)]\right] = \frac{1}{2} + \frac{1}{2}\Pr[A = f(X)] \geq \frac{1}{2}(1 + \mathsf{pred}_f + \epsilon)$$

We must still bound the expected value of pred_{g_r}. Let $r_z = (-1)^{z \odot r}$. For any particular, r, we can compute pred_{g_r} as $\frac{1}{2} + \frac{1}{2}\left|\sum_z p_z r_z\right|$. Using the fact $\mathbb{E}\left[|Z|\right] \leq \sqrt{\mathbb{E}\left[Z^2\right]}$ for any random variable Z, we get:

$$\mathbb{E}_r\left[\mathsf{pred}_{g_r}\right] = \frac{1}{2} + \frac{1}{2}\mathbb{E}_r\left[\left|\sum_z p_z r_z\right|\right] \leq \frac{1}{2} + \frac{1}{2}\sqrt{\mathbb{E}_r\left[\left(\sum_z p_z r_z\right)^2\right]}$$

By pairwise independence of the variables r_z, we have $\mathbb{E}\left[r_z r_a\right]$ is 1 if $z = a$ and 0 otherwise.

$$\mathbb{E}_r\left[\mathsf{pred}_{g_r}\right] \leq \frac{1}{2} + \frac{1}{2}\sqrt{\sum_z p_z^2} \leq \frac{1}{2} + \frac{1}{2}\sqrt{\mathsf{pred}_f}.$$

The last inequality holds since pred_f is the maximum of the values p_z, and the expression $\sum_z p_z^2$ is maximized when $p_z = \mathsf{pred}_f$ for all z (note that this sum is the collision probability of $f(X)$). Combining the two calculations we have

$$\mathbb{E}_r\left[\Pr[r \odot A = g_r(X)] - \mathsf{pred}_{g_r}\right] \geq \frac{1}{2}\left(\mathsf{pred}_f + \epsilon - \sqrt{\mathsf{pred}_f}\right)$$

Using the hypothesis that $\mathsf{pred}_f \leq \epsilon^2/4$, we see that the expected advantage is at least $\epsilon/4$. □

We now turn to the proof of Lemma 5.2. It is tempting, as before, to consider predicates of the form $g(x) = g'(f(x))$ (this is certainly the form of the predicates given by Proposition 5.3). This approach cannot work in general: suppose that $Z = f(X)$ takes on values in $\{0, 1, 2\}$ with equal probability, and suppose that A takes the value of $f(X)$ with probability $1/3 + \epsilon$, and each of the other two values with probability $1/3 - \epsilon/2$. Now any predicate of Z takes on some value with probabilty at least $2/3$. A straightforward calculation shows that no matter what value of A is

observed, the best strategy is to guess the more likely value of the predicate. Hence, to prove Lemma 5.2 we'll have to consider a richer set of predicates.

Nonetheless, in the special case where f is *balanced* over an even number of outputs, we can consider the simpler predicates of the previous proof. We say $f : \{0,1\}^n \to \{1,...,F\}$ is δ-far from balanced with respect to X if for every value $z \in [F] = \{1,...,F\}$ we have $|p_z - 1/F| \leq \delta$. Sub-Lemma 5.6 shows that essentially the same approach as before works for a balanced function; that is, it is sufficient to choose a random *balanced* predicate.

SubLemma 5.6 (Nearly balanced functions). *Suppose F is even and $f : \{0,1\}^n \to [F]$ is δ-almost-balanced. If $\Pr[A = f(X)] \geq \mathsf{pred}_f + \epsilon$, then there is a predicate $g(x) = g'(f(x))$ such that*

$$\Pr[g'(A) = g(X)] \geq \mathsf{pred}_g + \epsilon/2 - \delta\sqrt{F}.$$

In particular, when $F \leq 2/\epsilon$ and $\delta \leq \epsilon^{3/2}/8$ the predictor $g'(A)$ has advantage at least $\epsilon/4$ over random guessing.

Proof. It is sufficient to consider a *random* predicate $G' : [F] \to \{-1,+1\}$ subject to the constraint that $G'(z) = -1$ on exactly half the elements of $[F]$. (The constraint can be satisfied F is even.) As in the proof of Proposition 5.3, we will compute the expected prediction probability of $G'(A)$ and the expectation of $\mathsf{pred}_{G'}$ separately.

We first compute the expected probability that $G'(A) = G'(f(X))$. Conditioned on the event $A = f(X)$, we always have $G'(A) = G'(f(X))$, and conditioned on $A \neq f(X)$, we have $G'(A) = G'(f(X))$ with probability $\frac{1}{2} - \frac{1}{2(F-1)}$ (the difference from $\frac{1}{2}$ comes from the fact that we choose G' only from functions which are balanced on $[F]$).

Let $\hat{p} = \Pr[A = f(X)]$. The expected prediction probability is given by

$$\mathbb{E}_{G'}\left[\Pr[G'(A) = G'(f(X))]\right] = \hat{p} + (1-\hat{p})(\frac{1}{2} - \frac{1}{2(F-1)}) = \frac{1}{2} + \frac{1}{2}(\hat{p} - \frac{1-\hat{p}}{F-1}).$$

By hypothesis $\hat{p} \geq \mathsf{pred}_f + \epsilon \geq 1/F + \epsilon$. Simplifying, we get $\mathbb{E}_{G'}\left[\Pr[G'(A) = G'(f(X))]\right] \geq \frac{1}{2} + \epsilon/2$.

We can also compute the expectation of pred_G (as in the proof of Proposition 5.3). Note that if f is perfectly balanced, $\mathsf{pred}_{G'}$ is always exactly $1/2$. More generally, for each z, let $\delta_z = p_z - \frac{1}{2}$ (recall that $|\delta_z| \leq \delta$ by hypothesis). Since G' is always balanced on $[F]$, for any particular g' we have $\mathsf{pred}_{g'} = \frac{1}{2} + \frac{1}{2}\left|\sum_z \delta_z g'(z)\right|$ (using the convention that g' maps into $\{\pm 1\}$). In expectation, we can apply the inequality $\mathbb{E}\left[|Z|\right] \leq \sqrt{\mathbb{E}\left[Z^2\right]}$ to get:

$$\mathbb{E}_{G'}\left[\mathsf{pred}_{G'}\right] \leq \frac{1}{2} + \frac{1}{2}\sqrt{\mathbb{E}_{G'}\left[\left(\sum_z \delta_z G'(z)\right)^2\right]} = \frac{1}{2} + \frac{1}{2}\sqrt{\sum_{z,z'} \delta_z \delta_{z'} \mathbb{E}_{G'}\left[G'(z)G'(z')\right]}.$$

We know that $\mathbb{E}_{G'}[G'(z)G'(z')]$ is 1 for $z = z'$ and is $\frac{-1}{F-1}$ otherwise. Using $|\delta_z| \leq \delta$ we get:

$$\mathbb{E}_{G'}[\mathsf{pred}_{G'}] \leq \frac{1}{2} + \frac{1}{2}\sqrt{\sum_z \delta^2 + \sum_{z \neq z'} \frac{\delta^2}{F-1}} \leq \frac{1}{2} + \frac{1}{2}\delta\sqrt{2F}.$$

The expectation of $\Pr[G'(A) = G'(f(X))] - \mathsf{pred}_{G'}$ is at least $\epsilon/2 - \delta\sqrt{F}$, as desired. □

SubLemma 5.7 (Balancing Functions). *Let f be any function such that $\Pr[A = f(X)] \geq \mathsf{pred}_f + \epsilon$. If $\mathbf{H}_\infty(X) \geq \log(1/\delta)$, then there is a function f' such that*

1. *f' takes values in $[F]$, for $F \leq \min\left\{\frac{2}{\mathsf{pred}_f}, \frac{4}{\epsilon}\right\} + 2$, and*
2. *f' is δ-almost-balanced.*
3. *$\exists B(\cdot)$ such that $\Pr[B(A) = f'(X)] \geq \mathsf{pred}_{f'} + \epsilon/4$.*

We can prove Sub-Lemma 5.7 using two claims: the first reduces the number of possible outputs simply by "bucketing" certain outputs together. The second claim shows that a function with not too many output can be made almost perfectly balanced, as long as the entropy of X is high enough.

Claim 5.8. *Let f be any function such that $\Pr[A = f(X)] \geq \mathsf{pred}_f + \epsilon$. Then there's a function b such that $f'(x) = b(f(x))$ satisfies $\mathsf{pred}_{f'} \leq \mathsf{pred}_f + \epsilon/2$, and such that f' takes values in $[F]$, for $F \leq \min\left\{\frac{2}{\mathsf{pred}_f}, \frac{4}{\epsilon}\right\} + 2$.*

Proof. We can gradually reduce the number of possible values f can take without decreasing the advantage of the predictor. Let $p_z = \Pr[f(X) = z]$. If there are two values z, z' such that both p_z and p'_z are at most $\mathsf{pred}_f/2 + \epsilon/4$, then we can identify those two outputs. Note that the combined value has probability at most $\mathsf{pred}_f + \epsilon/2$. We can continue to combine pairs of values with combined mass at most $\mathsf{pred}_f + \epsilon/2$ until there is at most one value z with mass less than $\mathsf{pred}_f/2 + \epsilon/4$.

Let $F = \left\lceil \min\left\{\frac{2}{\mathsf{pred}_f}, \frac{4}{\epsilon}\right\} \right\rceil + 1$. At the end of these successive combinations, we will have at most F different outputs remaining. We can thus summarize the previous steps in a single function b with range $[F]$ such that $b(z) = b(z')$ if and only if the outputs z and z' we're identified at some stage. This b satifies the conditions of the theorem: by contruction, we have $\Pr[b(f(X)) = w]$ is at most $\mathsf{pred}_f + \epsilon/4$ for any value w. Moreover, $\Pr[b(A) = b(f(X))] \geq \Pr[A = f(X)] \geq \mathsf{pred}_f + \epsilon$. □

Claim 5.9. *Let $f : \{0,1\}^n \to [F]$ be any function such that $\Pr[A = f(X)] \geq \mathsf{pred}_f + \epsilon$. If $\mathbf{H}_\infty(X) \geq \log(1/\delta)$, then there is a function $f' : \{0,1\}^n \to [F]$ such that*

1. *f' is δ-almost-balanced, and*
2. *$\Pr[A = f'(X)] \geq \frac{1}{F} + \frac{1/F}{\mathsf{pred}_f} \cdot \epsilon$.*

In particular, if $F \leq \frac{2}{\mathsf{pred}_f}$, then the advantage of A at predicting $f'(X)$ is $\epsilon/2$.

Proof. We may imagine the function f as dividing the points $x \in \{0,1\}^n$ into F buckets. Our goal is to move some of these points between buckets so that all the buckets have approximately the same size. Since no point has mass more than δ, we will be able to have all buckets with mass within δ of $1/F$.

The problem is that moving points between buckets will (in general) decrease the chances that A will predict the function (i.e. bucket identifier) accurately. Fortunately, we're interested in the difference between the predicition probability and the maximum bucket size. As long as the two decrease proportionately, then A will remain a good predictor for the modified function.

We now formalize our intuition. Let $p_z = \Pr[f(X) = z]$, and let $S \subseteq [F]$ be the set of z such that $p_z \geq 1/F$. We will change the function f on inputs x such that $f(x) \in S$, and assign instead a value not in S. We keep moving points as long as some bucket remains with mass above $1/F + \delta$. Note that buckets in S will all have mass in $[1/F, 1/F + \delta]$ at the end of this process.

Consider a large bucket given by value z. To decide which points to move out of the bucket, let $w_x = \Pr[A = f(x)|X = x]$, and let $q_z = \Pr[A = f(X)|f(X) = z]$. The value q_z is the average of w_x over all x in the bucket given by z:

$$q_z = \mathbb{E}_{X|f(X)=z}[w_X]$$

By always moving points with the smallest possible values of w_x, we can ensure that the average w_x in the bucket always remains at least q_z. Specifically, let f' be the new function obtained by balancing the buckets, and let $q'_z = \Pr[A = z|f'(X) = z]$. Then by moving x's with small w_x we can ensure that $q'_z \geq q_z$ for all $z \in S$.

At the end of this process we will have

$$\Pr[A = f'(X)] \geq \sum_{z \in S} \frac{1}{F} q'_z + \sum_{z \notin S} p_z q_z \geq \frac{1}{F} \sum_{z \in S} q_z + \sum_{z \notin S} p_z q_z$$

(The contribution of a bucket not in S can be bounded below by $p_z q_z$, since it's contribution to the prediction probability can only increase with re-balancing.)

The original prediction probability was $\sum_z p_z q_z$. Thus the coefficients of the q_z in the new success probability have gone down by a factor of at most $\frac{1/F}{\mathsf{pred}_f}$ (that is, only coefficients in S have changed, and those have decreased from at most pred_f to at least $1/F$). Hence, we have

$$\frac{\Pr[A = f'(X)]}{\Pr[A = f(X)]} \geq \frac{1/F}{\mathsf{pred}_f}$$

Thus the new probability is at least $\frac{1}{F} + \frac{1/F}{\mathsf{pred}_f}\epsilon$, as desired. □

Combining the lemmas above, we can prove Lemma 5.2.

Chapter 6

Encryption of High-Entropy Sources

In this chapter, we discuss the results on entropic security to the encryption of mesages which are guaranteed to come from a high-entropy distribution. Roughly: if the adversary has only a small chance of guessing the message ahead of time, then one can design information-theoretically secure encryption (in the sense of hiding all functions, Definition 5.2) using a much shorter key than is usually possible—making up for the small entropy of the key using the entropy inherent in the message.

Section 6.1 gives some background on symmetric encryption schemes and states the main results of the chapter. The proofs and exact statement are given in Sections 6.2, 6.3 and 6.4.

6.1 Background

The problem which ignited the formal study of cryptography is that of symmetric-key one-time encryption. Alice and Bob share a secret key K and Alice wants to securely send some message M to Bob over a public channel. M is assumed to come from some a-priori distribution on $\{0,1\}^n$ (e.g., uniform), and the goal is to compute a ciphertext E which: (a) allows Bob to extract M from E using K; (b) reveals "no information" about M to the adversary Eve beyond what she already knew. Below, we write $E \leftarrow \mathcal{E}(M,K)$ and $M = \mathcal{D}(E,K)$.

Perfect and Computational Security The first formalization of this problem came in a fundamental work of Shannon [73], who defined "no information" by requiring that M and E be independent as random variables: using information theoretic notation (see Section 2.1.1), $\mathbf{I}(M;E) = 0$, where I is the mutual information. He showed a lower bound on key length for his definition: encrypting messages of length n requires at least n bits of shared key (more formally, the Shannon entropy of the key must be at least that of the message distribution: $\mathbf{H}_{sh}(K) \geq \mathbf{H}_{sh}(M)$). This bound is tight when the message is chosen uniformly from all strings of a fixed length

n, since one can use a one-time pad. This bound was extended to the interactive setting by Maurer [58].

Goldwasser and Micali [40] relaxed the notion of perfect security to the *computational* setting: namely, any *efficient* Eve can extract only negligible "information" about M from E. They had to properly redefine the notion of "information", since mutual information or conditional probabilities do not make much sense in a computationally-bounded world. They suggested two now classical definitions. Consider the following, equivalent version of Shannon's definition: the encryption of any two messages yield the same distribution on ciphertexts, that is $\mathcal{E}(m_0) = \mathcal{E}(m_1)$. The first definition of Goldwasser and Micali, called *computational indistinguishability of encryptions*, generalizes this version of perfect security: they require that no efficient (polynomial-time adversary) can distinguish the encryptions of m_0 and m_1 with advantage more than ϵ over random guessing, where ϵ is some negligible quantity. Their second notion is called *semantic security*: for *any* distribution on messages M and any function $f()$, the adversary can predict $f(M)$ given $\mathcal{E}(M)$ with probability only negligibly better than she could without seeing $\mathcal{E}(M)$. The first definition is easier to work with, but the second definition seems to capture a stronger, more intuitive notion of security: for example, indistinguishability is the special case of semantic security when the message distribution M is restricted to uniform distributions over two points $\{m_0, m_1\}$. In fact, Goldwasser and Micali showed that the two definitions are equivalent. Thus, min-entropy 1 distributions[1] are in some sense the hardest to deal with for semantic security.

Statistical Security? A natural intermediate notion of security between perfect and computational security would be some kind of *statistical security*: Eve is again allowed to be unbounded, as in the perfect setting, but can potentially get some negligible "information" ϵ, as in the computational setting. At the first glance, it seems there there is no gain in this notion, no matter how we interpret "information". For example, following Shannon's approach we could require that $\mathbf{I}(M; E) \leq \epsilon$ instead of being 0. Unfortunately, Shannon's proof still implies that $\mathbf{H}_{sh}(K) \geq \mathbf{H}_{sh}(M) - \epsilon$. Similarly for indistinguishability: since $\mathcal{E}(m)$ should look almost the same for *any* fixed m, one can argue that $\mathbf{I}(E; M) = \mathbf{H}_{sh}(\mathcal{E}(M)) - \mathbb{E}_m \left[\mathbf{H}_{sh}(\mathcal{E}(m))\right]$ still has to be negligible, and so the key must again have entropy almost $\mathbf{H}_{sh}(M)$. The same bound also holds for semantic security.

In his original work Shannon envisioned applications where Eve has a lot of uncertainty about the message. Indeed, to get a pessimistic bound that $\mathbf{H}_{sh}(K) \geq n$, one only has to consider the uniform distribution on M. In the perfect setting the security against the uniform (i.e., min-entropy n) distribution implies the security against *any* distribution. On the other hand, the notions of indistinguishability and semantic security primarily deal with min-entropy 1 distributions, and the straightforward extension of Shannon's bound to the statistical versions of these notions *crucially uses this fact*. Thus, it is natural to ask if we can meaningfully define (statistical) semantic

[1] As defined in Section 2.1.1, the *min-entropy* of a distribution A is $\mathbf{H}_\infty(A) = -\log(\max_a \Pr(A = a))$.

security and/or indistinguishability for high min-entropy distributions (say, uniform), similar in spirit to the original work of Shannon. And if yes,

1. How do these notions relate to Shannon's (statistical) notion, $\mathbf{I}(M;E) \leq \epsilon$? Most importantly, does the pessimistic bound on the key length still extend to these notions?
2. How do these notions relate to each other? In particular, are they still equivalent?

The work of Russell and Wang [70] Russell and Wang [70] introduced the idea of statistical security for encryption of high-entropy message spaces. They considered the first question above, though they focused on weakened version of semantic security. They proposed entropic security *for predicates* (Definition 5.1) as a definition for the high-entropy setting. Remarkably, Russell and Wang showed that Shannon's lower bound does *not* extend to this new notion.

Specifically, they presented two schemes beating Shannon's bound on key length. First, a deterministic scheme of the form $\mathcal{E}(M,K) = M \oplus p(K)$, which is secure only when M is uniformly distributed on $\{0,1\}^n$, where K has length only $k = 2\log n + 3\log\left(\frac{1}{\epsilon}\right) + O(1)$ and $p(\cdot)$ is some carefully designed function from k to n bits.[2] Thus, $p(K)$ could be viewed as a very "sparse one-time pad" which nevertheless hides any a-priori specified predicate $g(M)$. Second, for general min-entropy t, Russell and Wang gave a very different looking *randomized* scheme of the form $(\psi, \psi(M)+K) \leftarrow \mathcal{E}(M,K)$, where ψ is chosen at random from some special class of permutations[3] (and the addition is defined over some appropriate space). To achieve entropic security for distributions of min-entropy t, this second scheme needs key length $n - t + 3\log\left(\frac{1}{\epsilon}\right) + O(1)$. While less than n for nontrivial settings of $n-t$, this key length again becomes $\Omega(n)$ when $n - t = \Omega(n)$. [70] left it open whether such dependence on $n-t$ is necessary.

Our Results Our results on entropically-secure encryption can be divided into four areas:

- A stronger definition of security. As shown in the previous chapter (Theorem 5.1), entropic security for predicates implies entropic security for all functions.
- The equivalence of entropic security and indistinguishability of encryptions for message spaces with high-min-entropy. The results of the previous chapter imply that $\mathcal{E}(m,K)$ is an entropically secure encryption scheme if and only if $\mathcal{E}(m,K)$ satisfies indistinguishability on high-entropy input distributions.
- Lower bounds on the key length k for entropic security and indistinguishability. In particular, we show near tightness of Russell-Wang constructions: $k > n - t$. (In fact, for a large class of schemes $k \geq n - t + \log\left(\frac{1}{\epsilon}\right)$.)

[2] Namely, it samples a random point $p(K)$ from an appropriate δ-biased spaces [64] (where [70] used $\delta = \epsilon^{3/2}$).

[3] Specifically, Russell and Wang required a family of 3-wise indepepndent permutations.

- Two general frameworks for constructing entropically secure encryption schemes, one based on expander graphs and the other on XOR-universal hash functions. These schemes generalize the schemes of Russell and Wang, yielding simpler constructions and proofs as well as improved parameters.

Just like in the computational setting [40], the equivalence of security and indistinguishability allows us to concentrate on a simpler definition of indistinguishability, which immediately gives several benefits.

On one hand, we use it to show that the general construction of Russell and Wang is nearly optimal: *any* entropically secure scheme must have $k > n - t$. In fact, for a special case of *public-coin* schemes, where the ciphertext contains the randomness used for encryption,[4] we get an even stronger bound: $k \geq n - t + \log\left(\frac{1}{\epsilon}\right)$. The latter result is proven by relating the notion of indistinguishability to that of *randomness extractors* [66]: namely, any indistinguishable public-coin scheme almost immediately yields a corresponding extractor. Using the optimal lower bounds on extractors [67], we get our stronger bound as well. We notice that all the schemes in [70] and this work are indeed public-coin.

On the other hand, the indistinguishability view allows us to give a general framework for constructing entropically secure encryption schemes. Specifically, assume we have a d-regular expander G on 2^n vertices V with the property that for any subset T of 2^t vertices, picking a random vertex v of T and taking a random neighbor w, we obtain an almost uniform distribution on V. Then, we almost immediately get an encryption scheme with key length $k = \log d$ which is indistinguishable for message spaces of min-entropy t. Looking at this from another perspective, the above encryption scheme corresponds to a randomness extractor which takes a source M of length n and min-entropy t, invests $\log d$ extra random bits K, and extracts n almost random bits E (with the additional property that the source M is recoverable from E and K). From this description, it is clear that the key length of this paradigm must be at least $n - t$ (which we show is required in any entropically secure encryption scheme). However, using optimal expanders we can (essentially) *achieve* this bound, and in several ways. First, using Ramanujan expanders [55], we get the best known construction with key length $k = n - t + 2\log\left(\frac{1}{\epsilon}\right)$. Second, using δ-biased spaces [64] (for appropriate $\delta = \delta(\epsilon, n, t)$ explained later) and their expansion properties (e.g., see [9]), we get a general construction with slightly larger but still nearly optimal key length $k = n - t + 2\log n + 2\log\left(\frac{1}{\epsilon}\right)$. This last result generalizes (and slightly improves) to any value of t the special case of the uniform message distribution ($n - t = 0$) obtained by Russell and Wang [70]. Our approach also gives clearer insight as to why small-biased spaces are actually useful for entropic security.

While the above deterministic constructions are nearly optimal and quite efficient, we also observe that one can get simpler constructions by allowing the encryption scheme to be *probabilistic*. In our approach, this corresponds to having a *family* of "average case" expanders $\{G_i\}$ with the property that for any set T of size at least 2^t, picking a random graph G_i, a random v in T and taking a random neigbor w of v in G_i,

[4] In particular, this includes all the deterministic schemes.

we get that w is nearly uniform, *even given the graph index* i. By using any family of pairwise independent hash functions h_i (resp. permutations ψ_i) and a new variant of the leftover hash lemma [46], we get a probabilistic scheme of the form $\langle i,\ M \oplus h_i(K)\rangle$ (resp. $\langle i,\ \psi_i(M) \oplus K\rangle$) with a nearly optimal key length $k = n - t + 2\log\left(\frac{1}{\epsilon}\right)$. As a concrete example of this approach, we get the following simple construction: $\mathcal{E}(M, K; i) = (i, M + i \cdot K)$, where the local randomness i is a random element in $GF(2^n)$, $K \in \{0,1\}^k$ is interpreted as belonging to $GF(2^k) \subseteq GF(2^n)$, and addition and multiplication are done in $GF(2^n)$.

Once again, the result above (with permutations ψ_i) improves and simplifies the intuition behind the second scheme of Russell and Wang [70]. Indeed, the latter work had to assume that the ψ_i's come from a family of 3-wise independent permutations — which are more compicated and less efficient than 2-wise independent permutations (or functions) — and presented a significantly more involved analysis of their scheme.

6.2 Using Expander Graphs for Encryption

Formally, a symmetric encryption scheme is a pair of randomized maps $(\mathcal{E}, \mathcal{D})$. The encryption takes three inputs, an n-bit message m, a k-bit key κ and r random bits i, and produces a N-bit ciphertext $y = \mathcal{E}(m, \kappa; i)$. Note that the key and the random bits are expected to be uniform random bits, and when it is not necessary to denote the random bits or key explicitly we use either $\mathcal{E}(m, \kappa)$ or $\mathcal{E}(m)$. The decryption takes a key κ and ciphertext $y \in \{0,1\}^N$, and produces the plaintext $m' = \mathcal{D}(y, \kappa)$. The only condition we impose for $(\mathcal{E}, \mathcal{D})$ to be called an encryption scheme is completeness: for all keys κ, $\mathcal{D}(\mathcal{E}(m, \kappa), \kappa) = m$ with probability 1.

In this section, we discuss graph-based encryption schemes and show that graph expansion corresponds to entropically secure encryption schemes.

Graph-based Encryption Schemes Let $G = (V, E)$ be a d-regular graph, and let $N(v, j)$ denote the j-th neighbor of vertex v under some particular labeling of the edges. We'll say the labeling is *invertible* if there exists a map N' such that $N(v, j) = w$ implies $N'(w, j) = v$.

By Hall's theorem, every d-regular graph has an invertible labeling.[5] However, there is a large class of graphs for which this invertibility is much easier to see. The Cayley graph $G = (V, E)$ associated with a group $\mathcal{G}$ and a set of generators $\{g_1, ..., g_d\}$ consists of vertices labeled by elements of $\mathcal{G}$ which are connected when they differ by a generator: $E = \{(u, u \cdot g_i)\}_{u \in V, i \in [d]}$. When the set of generators contains all its inverses, the graph is undirected. For such a graph, the natural labeling is indeed invertible, since $N(v, j) = v \cdot j$ and $N'(w, j) = w \cdot j^{-1}$. All the graphs we discuss in this paper are in fact Cayley graphs, and hence invertibly labeled.

[5]We thank Noga Alon for pointing out this fact. If $G = (V, E)$ is a d-regular undirected graph, consider the bipartite graph with $|V|$ edges on each side and where each edge in E is replaced by the corresponding pair of edges in the bipartite graph. By Hall's theorem, there exist d disjoint matchings in the bipartite graph. These induce an invertible labelling on the original graph.

Now suppose the vertex set is $V = \{0,1\}^n$ and the degree is $d = 2^k$, so that the neighbor function N takes inputs in $\{0,1\}^n \times \{0,1\}^k$. Consider the encryption scheme:

$$\mathcal{E}(m,\kappa) = N(m,\kappa). \tag{6.1}$$

Notice, $\mathcal{E}$ is a proper encryption scheme if and only if the labeling is invertible. In that case, $\mathcal{D}(y,\kappa) = N'(y,\kappa) = m$. For efficiency, we should be able to compute N and N' in polynomial time. We will show that this encryption scheme is secure when the graph G is a sufficiently good expander. The following definition is standard:

Definition 6.1. *A graph $G = (V,E)$ is a (t,ϵ)-extractor if, for every set S of 2^t vertices, taking a random step in the graph from a random vertex of S leads to a nearly uniform distribution on the whole graph. That is, let U_S be uniform on S, J be uniform on $\{1,...,d\}$ and U_V be uniform on the entire vertex set V. Then for all sets S of size at least 2^t, we require that:*

$$\mathbf{SD}\left(\ N(U_S,J)\ ,\ U_V\ \right) \leq \epsilon.$$

The usual way to obtain extractors as above is to use good expanders. This is captured by the following lemma.

Lemma 6.1 (Expander Smoothing Lemma [41]). *A graph G with second largest (normalized) eigenvalue $\lambda \leq \epsilon 2^{-(n-t)/2}$ is a (t,ϵ)-extractor.*

The equivalence between entropic security and indistinguishability (Theorem 5.1) gives us the following result:

Proposition 6.2. *For a 2^k-regular, invertible graph G as above, the encryption scheme $(\mathcal{E},\mathcal{D})$ given by N, N' is (t,ϵ)-entropically secure if G is a $(t-2,2\epsilon)$-extractor (in particular, if G has second eigenvalue $\lambda \leq \epsilon \cdot 2^{-(n-t-2)/2}$).*

Proof. By Theorem 5.1, it suffices to show that $(t-2,\epsilon)$-indistinguishability. And this immediately follows from the lemma above and the fact that any min-entropy $(t-2)$ distribution is a mixture of flat distributions. □

We apply this in two ways. First, using optimal expanders (Ramanujan graphs) we obtain the best known construction of entropically-secure encryption schemes (Corollary 6.3). Second, we give a simpler and much stronger analysis of the original scheme of Russell and Wang (Corollary 6.4).

Corollary 6.3. *There exists an efficient deterministic (t,ϵ)-entropically secure scheme with $k = n - t + 2\log\left(\frac{1}{\epsilon}\right) + 2$.*

Proof. We apply Proposition 6.2 to *Ramanujan graphs*. These graphs are optimal for this particular construction: they achieve optimal eigenvalue $\lambda = 2\sqrt{d-1}$ for degree d [55]. The bound on k now follows. □

The main drawback of Ramanujan graphs is that explicit constructions are not known for all sizes of graphs and degrees. However, large families exist (e.g. graphs with $q+1$ vertices and degree $p+1$, where p and q are primes congruent to 1 mod 4). Below we also show that the construction from Russell and Wang [70] using small-biased spaces is actually a special case of Proposition 6.2 as well.

Using Small-biased Sets A set S in $\{0,1\}^n$ is *δ-biased* if for all nonzero $\alpha \in \{0,1\}^n$, the binary inner product $\alpha \odot s$ is nearly balanced for s drawn uniformly in S:

$$\Pr_{s \leftarrow S}[\alpha \odot s = 0] \in \left[\frac{1-\delta}{2}, \frac{1+\delta}{2}\right] \text{ or, equivalently, } \left|\mathbb{E}_{s \leftarrow S}\left[(-1)^{\alpha \odot S}\right]\right| \leq \delta. \qquad (6.2)$$

Alon et al. [2] gave explicit constructions of δ-biased sets in $\{0,1\}^n$ with size $O(n^2/\delta^2)$. Now suppose the δ-biased set is indexed $\left\{s_\kappa | \kappa \in \{0,1\}^k\right\}$. Consider the encryption scheme: $\mathcal{E}(m,\kappa) = m \oplus s_\kappa$. Russell and Wang introduced this scheme and showed that it is (n,ϵ)-entropically secure when $\delta = \epsilon^{3/2}$, yielding a key length of $k = 2\log n + 3\log\left(\frac{1}{\epsilon}\right)$. However, their analysis works only when the message is drawn uniformly from $\{0,1\}^n$.

We propose a different analysis: consider the Cayley graph for $\mathbb{Z}_2^n$ with generators S, where S is δ-biased. This graph has second eigenvalue $\lambda \leq \delta$ [64, 3, 9]. Hence, by Proposition 6.2 the scheme above is (t,ϵ)-entropically secure as long as $\delta \leq \epsilon 2^{-(n-t-2)/2}$. This gives a version of the Vernam one-time pad for high-entropy message spaces, with key length $k = n - t + 2\log n + 2\log\left(\frac{1}{\epsilon}\right) + O(1)$. Unlike [70], this works for *all* settings of t, and also improves the parameters in [70] for $n = t$.

Corollary 6.4. *If $\left\{s_\kappa | \kappa \in \{0,1\}^k\right\}$ is a δ-biased set, then the encryption scheme $\mathcal{E}(m,\kappa) = m \oplus s_\kappa$ is (t,ϵ) indistinguishable when $\epsilon = \delta 2^{(n-t-2)/2}$. Using the costruction of [2], this yields a scheme with key length $k = n - t + 2\log\left(\frac{1}{\epsilon}\right) + 2\log(n) + O(1)$ (for any value of t).*

6.3 A Random Hashing Construction

This section presents a simpler construction of entropically secure encryption based on pairwise independent hashing. Our result generalizes the construction of Russell and Wang [70] for nonuniform sources, and introduces a new variant of the leftover-hash/privacy-amplification lemma [7, 46].

The idea behind the construction is that indistinguishability is the same as extraction from a weak source, except that the extractor must in some sense be invertible: given the key, one must be able to recover the message.

Let $\{h_i\}_{i \in I}$ be some family of functions $h_i : \{0,1\}^k \to \{0,1\}^n$, indexed over the set $I = \{0,1\}^r$. We consider encryption schemes of the form

$$\mathcal{E}(m,\kappa;i) \;=\; (\; i,\; m \oplus h_i(\kappa)\;) \quad \text{(for general functions } h_i\text{), or} \tag{6.3}$$
$$\mathcal{E}'(m,\kappa;i) \;=\; (\; i,\; h_i(m) \oplus \kappa\;) \quad \text{(when the functions } h_i \text{ are permutations)} \tag{6.4}$$

Notice that this schemes are special low-entropy, probabilistic one-time pads. Decryption is obviously possible, since the description of the function h_i is public. For the scheme to be (t,ϵ)-secure, we will see that it is enough to have $k = n - t + 2\log\left(\frac{1}{\epsilon}\right) + 2$, and for the function family to be pairwise independent. (This matches the result in Corollary 6.3.) In fact, a slightly weaker condition (XOR-universality) is sufficient. We repeat the definitionn from the introduction here:

Definition 6.2 (XOR-universal function families, also Definition 2.3). *A collection of functions $\{h_i\}_{i\in I}$ from n bits to n bits is* XOR-universal *if:*

$$\forall a,x,y \in \{0,1\}^n, x \neq y: \quad \Pr_{i\leftarrow I}[h_i(x) \oplus h_i(y) = a] \leq \tfrac{1}{2^n-1}.$$

It is easy to construct XOR-universal families. Any (ordinary) pairwise independent hash family will do, or one can save some randomness by avoiding the "offset" part of constructions of the form $h(x) = ax + b$. Specifically, view $\{0,1\}^n$ as $\mathcal{F} = GF(2^n)$, and embed the key set $\{0,1\}^k$ as a subset of $\mathcal{F}$. For any $i \in \mathcal{F}$, let $h_i(\kappa) = i\kappa$, with multiplication in $\mathcal{F}$. This yields a family of linear maps $\{h_i\}$ with 2^n members. Now fix any $a \in \mathcal{F}$, and any $x, y \in \mathcal{F}$ with $x \neq y$. When i is chosen uniformly from $\{0,1\}^n$, we have $h_i(x) \oplus h_i(y) = i(x-y) = a$ with probability exactly 2^{-n}. If we restrict i to be nonzero, then we get a family of *permutations*, and we get $h_i(x) \oplus h_i(y) = a$ with probability at most $\frac{1}{2^n-1}$.

Proposition 6.5. *If the family $\{h_i\}$ is XOR-universal, then the encryption schemes*

$$\mathcal{E}(m,\kappa;i) = (i, m \oplus h_i(\kappa)) \quad \textit{and} \quad \mathcal{E}'(m,\kappa;i) = (i, h_i(m) \oplus \kappa)$$

are (t,ϵ)-entropically secure, for $t = n - k + 2\log\left(\frac{1}{\epsilon}\right) + 2$. (However, $\mathcal{E}'$ is a proper encryption scheme only when $\{h_i\}$ is a family of permutations.)

This proposition proves, as a special case, the security of the Russell-Wang construction, with slightly better parameters (their argument gives a key length of $n - t + 3\log\left(\frac{1}{\epsilon}\right)$ since they used 3-wise independent permutations, which are also harder to construct). It also proves the security of the simple construction $\mathcal{E}(m,\kappa;i) = (i, m + i\kappa)$, with operations in $GF(2^n)$.

Proposition 6.5 follows from the following lemma of independent interest, which is closely related to the to the *leftover hash lemma* [44] (also called *privacy amplification*; see, e.g. [7, 8]), and which conveniently handles both the $\mathcal{E}$ and the $\mathcal{E}'$ variants. The proof of the lemma is in Appendix A.2.

Lemma 6.6. *If A, B are independent random variables such that $\mathbf{H}_\infty(A) + \mathbf{H}_\infty(B) \geq n + 2\log\left(\frac{1}{\epsilon}\right) + 1$, and $\{h_i\}$ is a XOR-universal family, then $\mathbf{SD}\left(\langle i,\; h_i(A) \oplus B\rangle\;,\; \langle\; i,\; U_n\rangle\right) \leq \epsilon$, where U_n and i are uniform on $\{0,1\}^n$ and $\mathcal{I}$.*

The Lemma above gives a special "extractor by XOR" which works for product distributions $A \times B$ with at least n bits on min-entropy between them.

6.4 Lower Bounds on the Key Length

Proposition 6.7. *Any encryption scheme which is (t, ϵ)-entropically secure for inputs of length n requires a key of length at least $n - t$.*

Proof. We can reduce our entropic scheme to Shannon-secure encryption of strings of length $n - t + 1$. Specifically, for every $w \in \{0,1\}^{n-t+1}$, let M_w be the uniform over strings with w as a prefix, that is the set $\{w\} \times \{0,1\}^{t-1}$. Since M_w has min-entropy $t - 1$, any pair of distributions $\mathcal{E}(M_w), \mathcal{E}(M_{w'})$ are indistinguishable, and so we can use $\mathcal{E}()$ to encrypt strings of length $n-t+1$. When $\epsilon < 1/2$, we must have key length at least $(n-t+1)-1 = n-t$ by the usual Shannon-style bound (the loss of 1 comes from a relaxation of Shannon's bounds to statistical security). □

Bounds for Public-Coin Schemes via Extractors In the constructions of Russell and Wang and that of Section 6.2 and Section 6.3, the randomness used by the encryption scheme (apart from the key) is sent *in the clear* as part of the ciphertext. That is, $\mathcal{E}(m, \kappa; i) = (i, \mathcal{E}'(m.\kappa; i))$. For these types of schemes, called *public-coin* schemes, the intuitive connection between entropic security and extraction from weak sources is pretty clear: encryption implies extraction. As a result, lower bounds on extractors [67] apply, and show that our construction is close to optimal.

Proposition 6.8. *Any* public-coin*, (t, ϵ)-entropically secure encryption has key length $k \geq n - t + \log\left(\frac{1}{\epsilon}\right) - O(1)$.*

To prove the result, we first reduce to the existence of extractors:

Lemma 6.9. *Let $(\mathcal{E}, \mathcal{D})$ be a* public-coin*, (t, ϵ)-entropically secure encryption scheme with message length n, key length k and r bits of extra randomness. Then there exists an extractor with seed length $k + r$, input length n and output length $n + r - \log\left(\frac{1}{\epsilon}\right)$, such that for any input distribution of min-entropy $t+1$, the output is within distance 3ϵ of the uniform distribution.*

Proof. We combine three observations. First, when U is uniform over all messages in $\{0,1\}^n$, the entropy of the distribution $\mathcal{E}(U)$ must be high. Specifically: $\mathbf{H}_\infty(\mathcal{E}(U)) = n + r$. To see this, notice that there is a function $(\mathcal{D})$ which can produce R, K, U from $K, \mathcal{E}(U, K; R)$. Since the triple (R, K, U) is uniform on $\{0,1\}^{r+k+n}$, it must be that $(K, \mathcal{E}(U, K))$ also has min-entropy $r + k + n$, i.e. that any pair (κ, c) appears with probability at most $2^{-(n-k-r)}$. Summing over all 2^k values of κ, we see that any ciphertext value c appears with probability at most $\sum_\kappa 2^{-n-r-k} = 2^{-n-r}$, as desired.

The second observation is that there is a deterministic function ϕ which maps ciphertexts into $\{0,1\}^{n+r-\log\left(\frac{1}{\epsilon}\right)}$ such that $\phi(\mathcal{E}(U))$ is within distance ϵ of the uniform distribution. In general, any *fixed* distribution of min-entropy t can be mapped into $\{0,1\}^{t-\log(1/\epsilon)}$ so that the result is almost uniform (Simply assign elements of the original distribution one by one to strings in $\{0,1\}^{t-\log(1/\epsilon)}$, so that at no time do two

strings have difference of probability more than 2^{-t}. The total variation from uniform will be at most $2^{t-\log(1/\epsilon)} \cdot 2^{-t} = \epsilon$.). Note that ϕ need not be efficiently computable, even if both $\mathcal{E}$ and $\mathcal{D}$ are straightforward. This doesn't matter, since we are after a combinatorial contradiction.

Finally, by Theorem 5.1, for all distributions of min-entropy $t-1$, we have $\mathbf{SD}\left(\mathcal{E}(U), \mathcal{E}(M)\right) \leq 2\epsilon$, and so $\mathbf{SD}\left(\phi(\mathcal{E}(U)), \phi((\mathcal{E}(M))\right) \leq 2\epsilon$. By the triangle inequality, $\phi(\mathcal{E}(M))$ is within 3ϵ of the uniform distribution on $n + r - \log\left(\frac{1}{\epsilon}\right)$ bits, proving the lemma. □

We can now apply the lower bound of Radhakrishnan and Ta-Shma [67], who showed that any extractor for distributions of min-entropy t, distance parameter δ and d extra random bits, can extract at most $t + d - 2\log(1/\delta) + O(1)$ nearly random bits. From Lemma 6.9, we get and extractor for min-entropy $t+1$, $\delta = 3\epsilon$, $k+r$ extra random bits, and output length $n + r - \log(1/\epsilon)$. Thus, $n + r - \log(1/\epsilon)$ is at most $t + 1 + k + r - 2\log(1/\epsilon) + O(1)$, which immediately gives us Proposition 6.8.

Remark 6.1. We do not lose $\log(1/\epsilon)$ in the output length in Lemma 6.9 when the encryption scheme in indistinguishable from the uniform distribution (i.e., ciphertexts look truly random). For such public-coin schemes, we get $k \geq n - t + 2\log\left(\frac{1}{\epsilon}\right) - O(1)$. Since all of our constructions are of this form, their parameters cannot be improved at all. In fact, we conjecture that $k \geq n - t + 2\log\left(\frac{1}{\epsilon}\right) - O(1)$ for *all* entropically-secure schemes, public-coin or not.

Chapter 7

Entropically-Secure Sketches and Noise-resilient "Perfect" Hash Functions

In this chapter, we focus on constructing noise-resilient ("fuzzy") perfectly one-way hash functions [17, 18].

First, we focus on strengthening the definition of *secure sketches* and *fuzzy extractors*, introduced in Chapter 3. These primitives allow the secure use of noisy data as cryptographic keys. The current definitions are sufficient for most cryptographic applications: the idea is to leak a small amount of information about the noisy data to allow error-correction, and then extract a new key which is completely secret.

Unfortunately this approach fails to address a major concern for sensitive passwords: because it requires only unpredictability of the password given the stored information, it does not rule out revealing large amounts of the password in the clear. For example, if the password is a pair ⟨voice print, retinal scan⟩, the definition does not rule out a secure sketch which reveals the retinal scan entirely, as long as it leaves much of the voice print hidden.

It is tempting to attempt to remedy this by requiring that a secure sketch (resp. fuzzy extractor) reveal no information at all about its input. This is impossible: we proved in Section 4.2 that both secure sketches and fuzzy extractors must reveal a lot of Shannon information about their input, that is $\mathbf{I}(X; \mathsf{SS}(X))$ (resp. $\mathbf{I}(X; P_{FE}(X))$) will be quite large.

Surprisingly, we show that a strong information-theoretic notion of privacy can still be achieved: we construct secure sketches and fuzzy extractors which leak no function of their input, in the sense of entropic security Definition 5.2. This means we can guarantee that $\mathsf{SS}(w)$ *will be useful for error-correction, and nothing else.* Although we focus on the Hamming metric, the results extend to any metric which can be embedded into it (in the weak sense of "biometric embeddings," Section 3.3).

The latter part of the chapter uses these ideas to construct "fuzzy" (noise-resilient) *perfectly one-way hash functions* [17, 18]. Along the way, we get simpler constructions of the ordinary, non-fuzzy version of the primitive.

7.1 Entropic Secrecy for Secure Sketches and Extractors

This section focuses on the construction of secure sketches which hide all functions of their input. Recall that a *secure sketch* $\mathsf{SS}(w)$ of some secret w can be stored in the clear, revealing little information about w yet allowing one to recover w from any close candidate $w' \approx w$. A typical construction of a secure sketch is

$$\mathsf{SS}(W) = \mathsf{syn}_C(w)$$

where $syn_C(\cdot)$ is the syndrome function with respect to some good linear code C (see Section 3.4). A typical use of the primitive is for password authentication: roughly, a server stores $\mathsf{SS}(w), Hash(w)$. The identity of a legitmate user who presents a corrpupted password w' is verified by first using $\mathsf{SS}(w)$ to recover w and then verifying that this candidate password hashes to the correct value (see Section 3.8).

Unfortunately, this stored data $\langle \mathsf{SS}(w), Hash(w) \rangle$ inevitably leaks information about its input, in two senses: both the min-entropy and the Shannon entropy of the password w will drop when $\mathsf{SS}(w)$ is learned by the adversary. If the server's storage is publicly readable, the leakage could cause several kinds of problems:

1. The information might help the adversary find a string $\tilde{w}$ which the server would accept as a valid password. This possibility can be guarded against by bounding the loss of min-entropy (even a losse bound will do) and choosing the hash function carefully; see Section 3.8 for details.

2. The information which is leaked may itself be sensitive, for example if the password w consists of personal or biometric data. Proposition 4.3 states that the loss of Shannon information will be large, and so the only option is to prove that whatever information is leaked is not *useful*.

This section shows how one can achieve such a guarantee: we construct secure sketches (and fuzzy extractors) which leak no function of W as long as the input W has sufficiently high entropy to begin with. This property is far from trivial to ensure: for example, in the "typical" construction above, the adversary always learns the syndrome of the W with respect to the code C.

Most of the section is devoted to constructing entropically-secure sketches. The existence of entropically-secure fuzzy extractors then follows by combining a secure sketch with pariwise-independent hash functions. (Section 7.1.5).

The main result of the section is summarized here.

Theorem 7.1. . *There exist (families of) (n, t, t', τ) efficient secure sketch schemes which are also (t, ϵ)-entropically secure, such that*

- *the tolerated error τ and the residual entropy t' are linear in n, and*
- *the information leakage ϵ is exponentially small*

whenever the original min-entropy t is linear in n. (That is, whenever $t = \Omega(n)$ then we can find schemes where τ, t' and $\log\left(\frac{1}{\epsilon}\right)$ are $\Omega(n)$).

Before proving the result, a word about parameters: the original entropy t of the input W is given by the context in which W arises. The error tolerance τ will also typically be specified externally—it is the amount of noise to which W will likely be subject. Thus, the goal is to get both the (entropic) security $\log\left(\frac{1}{\epsilon}\right)$ and the residual min-entropy t' as high as possible. The quantity $\log\left(\frac{1}{\epsilon}\right)$ measures the difficulty of learning some function of W, while t' measures the difficulty of guessing W exactly. In particular, t' is bounded below by $\log\left(\frac{1}{\epsilon}\right)$, since by the definition of entropic security the adversary's probability of predicting the identity function $f(W) = W$ is at most ϵ as long as t itself was small to begin with. Thus, it is sufficient to look for sketches will tolerate τ errors and are (t, ϵ)-entropically secure for $\tau, \log\left(\frac{1}{\epsilon}\right) = \Omega(n)$. Theorem 7.1 guarantees that such secure sketches do indeed exist.

7.1.1 A Non-Explicit Solution: Codes With Limited Bias

Our starting point for the construction of secure sketches is (perhaps surprisingly) the encryption scheme based on δ-biased sets at the end of Section 6.2. Recall that the *bias* of a random variable A over $\{0,1\}^n$ is the maximum bias of the parity of any subset of bits of A. Formally, the set is δ-biased if for every non-zero vector α in $\{0,1\}^n$, the dot product of α with A is within δ of being a fair coin, that is

$$\mathsf{bias}_\alpha(A) \stackrel{def}{=} \mathbb{E}\left[(-1)^{\alpha \odot A}\right] = 2\left|\Pr[\alpha \odot A = 1] - \tfrac{1}{2}\right| \leq \delta.$$

The bias of a set S is the bias of the uniform distribution over that set. In Section 6.2, we showed that the map $Y(X) = X \oplus S$ is (t, ϵ)-entropically secure whenever the bias of S is sufficiently small ($\delta \leq \epsilon 2^{-(n-t-1)/2}$).

With that in mind, consider the "code-offset" construction of secure sketches, from Section 3.4: if $C : \{0,1\}^k \to \{0,1\}^n$ is a code correcting τ errors, then set $\mathsf{SS}(w) = w \oplus C(R)$. If the distribution $C(R)$ is itself δ-biased, with δ sufficiently low, then the sketch is entropically secure. We obtain an initial result on entropic security:

Definition 7.1. *An error-correcting code $C : \{0,1\}^k \to \{0,1\}^n$ is δ-biased if the set of codewords forms a δ-biased set, i.e. if R is a random string of length k, then for all n-bit vectors $\alpha \neq 0^n$, we have $2|\Pr[\alpha \odot C(R)] - \frac{1}{2}| \leq \delta$.*

Proposition 7.2. *If $C : \{0,1\}^k \to \{0,1\}^n$ is an error-correcting code with bias $\delta \leq \epsilon 2^{-(n-t-1)/2}$, then $\mathsf{SS}(w) = w \oplus C(R)$ is an $(n, t, t-(n-k))$ secure sketch which is (t, ϵ)-entropically secure.*

To apply the proposition we need, at the least, a code (or rather ensemble of codes, since we deal with asymptotics) with good minimum distance and negligible bias. These may not be easy to construct (in particular, a code with bias less than 1 *cannot be linear*). Nevertheless, such codes do exist. If we choose a subset $C \subseteq \{0,1\}^n$

of size 2^k completely at random from $\{0,1\}^n$, then two events will occur with high probability: (1) the set C will form an error-correcting code with minimum distance d, where $k/n \approx (1 - h_2(d/n))/2$ [56], and (2) the set C will have bias approximately $2^{-(k-\log n)/2}$ [64]. By a union bound, the set C is likely to have both properties. It is not necessary to choose the set completely at random; 3-wise independence suffices.[1] Thus the code can be *described* using only $3n$ bits.

Of course, the random construction does not solve our problem: we would like to the recovery procedure of our secure sketch to be efficient, but it is not known how to decode random codes decoded efficiently. This raises a natural question:

> Does there exist an explicitly-constructible ensemble of good codes with small bias and poly-time encoding and decoding algorithms (ideally, codes with linear rate and minimum distance, and negligible bias)?

To the best of our knowledge, the problem remains open.

7.1.2 Efficient, Explicit Solutions via Randomization

In the remainder of this section, we circumvent this diffculty and construct *efficient* entropically-secure sketches. We show that the code-offset construction can be made indistinguishable (even with linear codes) when the choice of error-correcting code is *randomized* as opposed to always using the same fixed code.

Suppose that we have a family of k-dimensional linear error-correcting codes $\{C_i\}_{i \in I}$ indexed by some set I. We consider sketching schemes of the form (below $\mathsf{syn}_{C_i}(w)$ is the *syndrome* of w in C_i).

$$\begin{aligned} &\mathsf{SS}(w; i, x) = (i, w \oplus C_i(x)) \ , \text{ for } i \leftarrow I, x \leftarrow \{0,1\}^k \\ \text{or, equivalently,} \quad &\mathsf{SS}(w; i) = (i, \mathsf{syn}_{C_i}(w)) \ , \text{ for } i \leftarrow I \end{aligned} \tag{7.1}$$

Below, we establish a necessary condition on the code family for the construction to leak no partial information about the input w.

Bias and Secrecy Recall from Equation (6.2) that a single set S is called δ-biased if it has bias at most δ for all $\alpha \neq 0^n$. We generalize this to a family of sets by requiring that on average, the square of the bias with respect to every α is low (at most δ^2):

Definition 7.2. *A* family *of random variables (or sets) $\{A_i\}_{i \in I}$ is δ-biased if, for all $\alpha \neq 0^n$,*

$$\sqrt{\mathbb{E}_{i \leftarrow I} \left[\mathsf{bias}_\alpha(A_i)^2\right]} \leq \delta.$$

Note that this is *not* equivalent, in general, to requiring that the expected bias be less than δ. There are two important special cases:

1. If S is a δ-biased set, then $\{S\}$ is a δ-biased set family with a single member. As mentioned above, we do not know of explicitly constructed codes with small bias.

[1] The 3-wise independence construction is due to Venkat Guruswami.

2. A family of linear codes $\{C_i\}_{i\in I}$ is *δ-biased* if there is no word which is often in the dual $C_i^\perp$ of a random code C_i from the family. Specifically, the bias of a linear space with respect to a vector α is always either 0 or 1:

$$\mathsf{bias}_\alpha(C_i) = \begin{cases} 0 & \text{if} \quad \alpha \notin C_i^\perp \\ 1 & \text{if} \quad \alpha \in C_i^\perp \end{cases}$$

Hence a family of codes is δ-biased if and only if $\Pr_{i\leftarrow I}[\alpha \in C_i^\perp] \leq \delta^2$, for every $\alpha \neq 0^n$.

Note that for linear codes the expected bias must be at most δ^2, while for a single set the bias need only be δ.

The general theorem below will allow us to prove that the randomized code-offset construction is indistinguishable (and hence entropically-secure). We will mainly use Corollary 7.5, which states that even if the adversary knows the index of the code C_i in a small-bias family of codes, the distribution of $\mathsf{syn}_{C_i}(W)$ will be close to uniform in expectation over i.

Theorem 7.3. *Let $\{A_i\}_{i\in I}$ be a δ-biased family of random variables over $\{0,1\}^n$, with $\delta \leq \epsilon \cdot 2^{-\frac{n-t-1}{2}}$. Let B be an independent random variable with min-entropy at least t. Then* $\mathbf{SD}\left(\ (I, A_I \oplus B)\ ,\ (I, U_n)\ \right) \leq \epsilon$.

Proof. Theorem 7.3 The proof uses elementary Fourier analysis over the hypercube $\mathbb{Z}_2^n$. The intuition comes from the proof that Cayley graphs based on ϵ-biased spaces are good expanders: adding a δ-biased family of random variables to B will cause all the Fourier coefficients of B to be reduced by a factor of δ, which implies that *the collision probability* of B (see below) gets multiplied by δ also.

Let D_i be the distribution $A_i \oplus B$. Recall that for any probability distribution D on a set of size K, if $\mathsf{Col}(D) \leq (1+\epsilon^2)/K$, then D is within statistical distance ϵ of the uniform distribution (see Preliminaries, Section 2.1.1). Hence to prove the theorem it is sufficient to show that the collision probability of the *pair* $D = (i, D_i) = (i, A_i + B)$ is bounded above by $\frac{(1+2\epsilon^2)}{|\mathcal{I}|2^n}$.

Claim: $\mathsf{Col}(D) = \frac{1}{|\mathcal{I}|}\mathbb{E}_{i\leftarrow\mathcal{I}}\left[\mathsf{Col}(D_i)\right]$.

Proof. We can write out the probability of a collision (here prime $'$ denotes an independent copy):

$$\Pr[(I, D_I) = (I', D'_{I'}] = \sum_i \Pr[I = I' = i] \Pr[D_i = D'_i]$$

Factoring out $\frac{1}{|\mathcal{I}|}$, we get $\mathsf{Col}(D) = \frac{1}{|\mathcal{I}|}\sum_i \frac{1}{|\mathcal{I}|}\mathsf{Col}(D_i)$, as desired. □

To bound $\mathsf{Col}(D)$, we need only bound the average collision probability of D_i. To do so, we use a standard fact from Fourier analysis over the hypercube:

Fact 7.4. *For any distribution D_i on $\{0,1\}^n$, the collision probability $\mathsf{Col}(D_i)$ is given by the sum of the squared biases of D_i with respect to all possible vectors:*

$$\mathsf{Col}(D_i) = \frac{1}{2^n} \sum_{\alpha \in \{0,1\}^n} \mathsf{bias}_\alpha(D_i)^2 = \frac{1}{2^n} + \frac{1}{2^n} \sum_{\alpha \neq 0} \mathsf{bias}_\alpha(D_i)^2.$$

Since $D_i = A_i \oplus B$ (that is, the distribution of D_i is the convolution of A_i and B), we can compute the bias of D_i as a product of the biases of A_i and B:

$$\begin{aligned} \mathsf{bias}_\alpha(D_i) &= \mathbb{E}\left[(-1)^{\alpha \odot (A_i \oplus B)}\right] \\ &= \mathbb{E}\left[(-1)^{\alpha \odot (A_i)}\right] \mathbb{E}\left[(-1)^{\alpha \odot B}\right] = \mathsf{bias}_\alpha(A_i)\mathsf{bias}_\alpha(B). \end{aligned}$$

We now want to bound the bias of D_i. We don't know how this bias will behave for particular values of i, but we can use the fact that $\{A_i\}$ is δ-biased family to bound the *average* squared bias:

$$\mathbb{E}_i\left[\mathsf{bias}_\alpha(D_i)^2\right] \leq \mathbb{E}_i\left[\mathsf{bias}_\alpha(A_i)^2\right] \mathsf{bias}_\alpha(B)^2 \leq \delta^2 \mathsf{bias}_\alpha(B)^2.$$

Finally, we can combine these bounds:

$$\mathsf{Col}(D) = \tfrac{1}{|\mathcal{I}|}\mathbb{E}_i\Big[\underbrace{\tfrac{1}{2^n} + \tfrac{1}{2^n}\sum_{\alpha \neq 0} \mathsf{bias}_\alpha(D_i)^2}_{\mathsf{Col}(D_i)}\Big] = \frac{1}{|\mathcal{I}|2^n}(1 + \delta^2 \sum_{\alpha \neq 0} \mathsf{bias}_\alpha(B)^2)$$

By the fact above, the sum of squared biases of B is at most $2^n\mathsf{Col}(B)$. Since the min-entropy of B is at least t, its collision probability is at most 2^{-t}, and we get the bound $\mathsf{Col}(D) \leq \frac{1}{|\mathcal{I}|2^n}(1 + \delta^2 2^{-t+n})$. By hypothesis, $\delta \leq \epsilon 2^{-(n+t)/2}$, which implies the desired bound $\mathsf{Col}(D) \leq \frac{1}{|\mathcal{I}|2^n}(1+\epsilon^2)$. □

Corollary 7.5 (Small-Bias Codes Yield Entropically-Secure Sketches). *The randomized code-offset construction (Equation (7.1)) is a $(n, t, t-(n-k), \tau)$ secure sketch which is (t, ϵ) entropically secure, as long as $\{C_i\}_{i \in I}$ is a family of $[n, k, 2\tau+1]$ linear codes such that*

$$\delta^2 = \max_{\alpha \in \{0,1\}^n, \alpha \neq 0^n} \Pr_{i \leftarrow I}[\alpha \in C_i^\perp] \leq \epsilon^2 2^{-(n-t-1)} \tag{7.2}$$

7.1.3 Constructing Small-Bias Families of Linear Codes

To instantiate the construction of the previous section (Corollary 7.5), we need families of linear codes $\{C_i\}$ in $\{0,1\}^n$ such that:

1. All of the C_i have dimension k, with k as high possible. Recall that the entropy loss of the secure sketch is given by $n-k$, i.e. if the input has initial entropy t then the residual entropy is $t' = t + k - n$.

2. Each of the codes C_i can efficiently decode up to τ errors, with τ as large as possible. This corresponds to the number of errors corrected by the sketch.

3. The squared bias $\delta^2 = \max_{\alpha \neq 0^n} Pr_{i \leftarrow I}[\alpha \in C_i^\perp]$ of the code family is as small as possible. The sketch is then (t, ϵ)-entropically secure where $\log\left(\frac{1}{\epsilon}\right) = \log\left(\frac{1}{\delta}\right) - (n-t)/2$.

The main result of the section is a construction based on algebraic-geomtric codes:

Lemma 7.6. *For any constant rate $R = k/n$, there exists an explicitly constructible ensemble of code families which* efficiently *correct $\tau = \Omega(n)$ errors and have bias $\log\left(\frac{1}{\delta}\right) = \Omega(n)$.*

We begin with an easy lower bound, which gives us a bar by which to measure the construction we find:

Proposition 7.7. *Any family of $[n, k, d]$* linear *codes $\{C_i\}$ has bias $\delta^2 \geq 2^{-k}(1 - o(1))$.*

Proof. Let $p(\alpha)$ be the probability that a particular non-zero vector $\alpha \in \{0,1\}^n$ is in the dual of a random code from the family. If we choose α itself at random, then we can reverse the order of the experiment: first choose i, then choose α. The probability that a random α will lie in the dual $C_i^\perp$ is exactly $\frac{2^{n-k}-1}{2^n-1} \geq 2^k(1 - 2^{-(n-k)})$, since the dual is a space of dimension $n - k$. This probability is in fact the expectation of $p(\alpha)$ (over $\{0,1\}^n \setminus \{0^n\}$), and so there exists a non-zero α for which $p(\alpha)$ exceeds the bound. □

The bound means that $\log\left(\frac{1}{\delta}\right) < k/2$, and for *any* family of linear codes, the entropic security

$$\log\left(\frac{1}{\epsilon}\right) = \log\left(\frac{1}{\delta}\right) - (n-t)/2 \leq (k + t - n)/2 = t'/2.$$

Hence, in general the residual entropy t' bounds the level of entropic security (this is not surprising, since the adversary's chance of guessing the message itself is then $2^{-t'}$ on average).

We first observe that we can match this bound using random linear codes. The construction is explicit, but not useful since it is not known how to decode random linear codes efficiently. We then turn to efficient constructions based on random binary images of codes over slightly higher alphabets. For very large alphabets, a construction based on Reed-Solomon codes yields optimal bias, but poor error-correction in the binary Hamming cube. Applying the same construction to algebraic-geometric codes yields the main construction of this section, which corrects a linear number of errors and has exponentially small bias.

Inefficient Construction: Random Linear Codes An easy observation is that the family of *all* linear codes of a particular dimension k is a very good δ-biased family, albeit not an efficiently decodable one. Since the family is invariant under invertible linear transformations, the probability that any fixed word $\alpha \neq 0$ is contained in the dual of a random linear code is

$$\delta^2 = \frac{\#\left\{x \in C^\perp : x \neq 0\right\}}{\#\left\{x \in \{0,1\}^n : x \neq 0\right\}} = \frac{2^{n-k}-1}{2^n - 1} < 2^k.$$

This matches the lower bound on bias above (Proposition 7.7): for any initial entropy $t > n - k$, the corresponding secure sketch will be t, ϵ)-entropically scure for $\log\left(\frac{1}{\epsilon}\right) = t'/2 = (k + t - n)/2$.

Random linear codes also exhibit the best known tradeoff between rate and distance for binary codes. With overwhelming probability, a random linear code will have minimum distance d where $k/n \approx 1 - h_2(d/n)$ (this is the Gilbert-Varshamov bound [56]). Of course, this solution is not useful since decoding random linear codes is thought to be very hard (it is known to be NP-hard for some parameter settings [33], and little is known about other settings). Nonetheless, it gives us a point of reference with which to measure other constructions.

Efficient Constructions via Randomizing Known Codes A natural approach to construct small-bias codes consists of taking a known binary code C and considering all $n!$ codes resulting from permuting the n coordinates of $\{0,1\}^n$ (that is, apply the same permutation to the bits of every codeword in C). This approach works reasonably well if $C^\perp$ has no words of either very high or very low weight. Unfortunately, such codes are tricky to construct with very good parameters, and so we turn to codes over a larger alphabet.

Let $\mathcal{F} = GF(q)$, where $q = 2^e$. Taking a $[n', k', d]_q$ code C' over $\mathcal{F}$, we can construct a binary code by taking the *binary image* of C', that is by writing down the codewords of C' using some canonical e-bit binary representation for elements of $\mathcal{F}$. Fix a basis of $\mathcal{F}$. For $f \in \mathcal{F}$, let $\mathsf{bin}(f) \in \{0,1\}^e$ be the binary representation of f. By extension, for a vector $\alpha = (a_1, ..., a_{n'}) \in \mathcal{F}^{n'}$, let $\mathsf{bin}(\alpha)$ be the concatenation $(\mathsf{bin}(a_1), ..., \mathsf{bin}(a_{n'}))$. Finally, the binary image of C' is denoted $\mathsf{bin}(C')$.

We can randomize the code C' by

1. permuting the n' coordinates of $\mathcal{F}^{n'}$, and
2. multiplying each coordinate of the code by some random non-zero scalar in $\mathcal{F}$.
3. taking the binary image of the result.

These operations affect neither the dimension nor the decodability of C': they are invertible and preserve Hamming distances in $\mathcal{F}^{n'}$. Describing the particular operations that were applied to the code requires $O(n' \log n' + n' \log(q-1))$ bits (we must describe a permutation of n' positions and n' non-zero scalars).

We show below that using these basic randomization steps and then taking binary images produces good small-bias code families. We then apply the result to

Reed-Solomon codes and to algebraic-geometric codes to get our main construction (Lemma 7.6).

Lemma 7.8 (Random binary images). *Let C' be a linear $[n', k', d]_q$ code over $\mathcal{F} = GF(q)$, with $q = 2^e$. Let $\{C'_i\}$ be the set of $[n', k', d]_q$ codes over $\mathcal{F}$ obtained by permuting the coordinates and multiplying each coordinate by a non-zero scalar in $\mathcal{F}$. Let $C_i = \mathsf{bin}(C'_i)$. Then*

1. *The C_i are $[n, k, d]_2$ codes with $n = n'e$ and $k = k'e$. (In particular, the rate $k/n = k'/n'$ does not change).*
2. *If C' can correct τ errors in $\mathcal{F}^{n'}$ efficiently, than each C_i can efficiently correct τ errors in $\{0,1\}^n$.*
3. *If $(C')^\perp$ has minimum distance $d_\perp$, then the average square bias of $\{C_i\}$ is*

$$\delta^2 = \max_{\alpha \in \{0,1\}^n, \alpha \neq 0^n} \left\{ \Pr_i[\alpha \in C_i^\perp] \right\} \leq 1/(q-1)^{d_\perp - 1}.$$

Note that in the last statement, the dual code $(C')^\perp$ is taken with respect to the dot product in $\mathcal{F}^{n'}$, while the dual code $C_i^\perp$ is taken with respect to the dot product in $\{0,1\}^n$.

Proof. (1),(2): The first two statements are straightforward since the multiplication by non-zero scalars in one component and permutations of positions are easily invertible isometries of $\mathcal{F}^{n'}$.

(3): There are really two separate stages to proving this statement. In the first stage, we have to relate the dual of a q-ary code to the dual of a binary code. Second, we will bound the bias of the q-ary codes $\{C'_i\}$.

To clarify the notion of "dual" code, let $\odot_2$ denote binary inner product on $\{0,1\}^n$, and let $\odot_{\mathcal{F}}$ denote the standard inner product in $\mathcal{F}^{n'}$. The duals of the codes $C_i \subseteq \{0,1\}^n$ are defined with respect to the binary inner product, while the duals of the $C'_i \in \mathcal{F}^{n'}$ are defined w.r.t. the dot product over $\mathcal{F}^{n'}$:

$$\begin{aligned} C_i^\perp &= \{y \in \{0,1\}^n \ : \ y \odot_2 x = 0 \ (\forall x \in C_i)\} \\ (C'_i)^\perp &= \{y' \in \mathcal{F}^{n'} \ : \ y' \odot_{\mathcal{F}} x' = 0_{\mathcal{F}} \ (\forall x' y \in C'_i)\} \end{aligned}$$

For the rest of the proof, fix some $\alpha \in \{0,1\}^n$, and let α' be the corresponding vector in $\mathcal{F}^{n'}$, that is $\alpha = \mathsf{bin}(\alpha')$. The statement to be proved follows from two claims:

Claim 1: For all $\alpha \in \{0,1\}^n$, there exists $\alpha' \in \mathcal{F}^{n'}$ s.t. $\Pr_i[\alpha \in C_i^\perp] = \Pr_i[\alpha' \in (C'_i)^\perp]$.

Claim 2: For all $\alpha' \in \mathcal{F}^{n'}$, we have: $\Pr_i[\alpha' \in (C'_i)^\perp] \leq 1/(q-1)^{d_\perp - 1}$.

Proof of Claim 1. The first claim is mostly a careful unwinding of the definitions. We will use the trace function $\mathrm{Tr} : \mathcal{F} \to \{0,1\}$. The exact definition of the trace is not important here (see, e.g. [56]). All we require is that the trace is linear, i.e. $\mathrm{Tr}(a+b) = \mathrm{Tr}(a) + \mathrm{Tr}(b)$, and not identitically zero. $\mathrm{Tr}(ab)$ is a bilinear map from $\mathcal{F} \times \mathcal{F}$ to $\{0,1\}$, and so there exists an invertible linear transformation $B : \{0,1\}^e \to \{0,1\}^e$ such that for all scalars $a, b \in \mathcal{F}$, we have $B(\mathsf{bin}(a)) \odot_2 \mathsf{bin}(b) = \mathrm{Tr}(ab)$.

Fix $\alpha \in \{0,1\}^n$. We can choose the unique vector α' in $\mathcal{F}^{n'}$ such that α is the concatenation of the e-bit vectors $B(\mathsf{bin}(\alpha'_i))$. Then for any vector $x' \in \mathcal{F}^{n'}$, we have:

$$\alpha \odot_2 \mathsf{bin}(x') = \mathrm{Tr}(\alpha' \odot_{\mathcal{F}} x')$$

Sub-Claim: α is in $C_i^{\perp}$ if and only if α' is in $(C'_i)^{\perp}$.

One direction of the sub-claim is easy: suppose $\alpha' \in (C'_i)^{\perp}$. Then for any vector $x \in C_i$, we have $\alpha \odot_2 x = \mathrm{Tr}(\alpha' \odot_{\mathcal{F}} \mathsf{bin}^{-1}(x))$. Now the image of x in $\mathcal{F}^{n'}$ is in C'_i, and so $\mathrm{Tr}(\alpha' \odot_{\mathcal{F}} \mathsf{bin}^{-1}(x)) = \mathrm{Tr}(0_{\mathcal{F}}) = 0$. In the other direction (of the sub-claim), suppose that $\alpha \in C_i^{\perp}$. Suppose, to get a contradiction, that there is some $x' \in C'_i$ such that $\alpha' \odot_{\mathcal{F}} x' \neq 0_{\mathcal{F}}$. Then there exists some non-zero scalar $b \in \mathcal{F}$, such that $0 \neq \mathrm{Tr}(b(\alpha' \odot_{\mathcal{F}} x')) = \mathrm{Tr}(\alpha' \odot_{\mathcal{F}} (bx')) = \alpha \odot_2 \mathsf{bin}(bx')$. But the vector bx' is in C'_i since C'_i is a linear code, and so the inner product of its binary image with α should be 0. Thus, we get a contradiction and conclude that $\alpha' \in (C'_i)^{\perp}$, completing the proof of the sub-claim.

Based on the sub-claim, we can conclude that $\Pr_i[\alpha \in C_i^{\perp}] = \Pr_i[\alpha' \in (C'_i)^{\perp}]$. $\square$

Proof of Claim 2. The main observation behind this proof is that the randomization operations we use behave nicely in the dual space. Permuting the coordinates of the code C' induces the same permutation on the coordinates of C'. Similarly, if we multiply the n' coordinates by non-zero scalars $b_1, ..., b_{n'} \in \mathcal{F}$, then we multiply the dual code by the inverses $b_1^{-1}, ..., b_n^{-1}$. Thus we get the same family of q-ary codes C'_i by applying the randomization procedure to the dual instead of the primal code.

Now fix some vector $\alpha' \in \mathcal{F}^{n'}$. By symmetry, we can imagine that the randomizing operation is applied to the target word α' instead of to the code itself. *This maps α' to a random word in $\mathcal{F}^{n'}$ of the same weight as α.* The probability that this hits a codeword is exactly the fraction of words of a given weight w which are in the code. We call the set of words in $\mathcal{F}^{n'}$ with weight exactly w the w-slice. To complete the proof, we need only prove the following:

Sub-Claim: For a linear code over $\mathcal{F} = GF(q)$ of minimum distance $d_{\perp}$, the fraction of codewords in any slice of $\mathcal{F}^{n'}$ is bounded above by $(q-1)/(q-1)_{\perp}^{d}$ (except for the singleton slice $\left\{0^{n'}\right\}$).

To prove the sub-claim, fix some weight $0 < w \leq n'$. We can partition the slice of weight w according to which w positions in a word are non-zero. Each of these partitions can further be subidivided into pieces where all but $d_{\perp}$ of the non-zero

values are fixed, i.e. sets of the form

$$(\underbrace{0,...,0}_{n-w \text{ times}}, \underbrace{b_1,...,b_{w-d_\perp}}_{\text{non-zero scalars}}, \underbrace{*,...,*}_{d_\perp \text{ times}}),$$

up to permutation of coordinates, where $*$ may take any non-zero value.

Now within any such piece, there can be at most $q-1$ codewords (since the codewords must differ in $d_\perp$ positions). There are $(q-1)^d$ words in the piece, and so overall the fraction of codewords in any constant-weight slice is at most $(q-1)^{d-1}$. □

This completes the proof of Lemma 7.8. □

Remark 7.1. The key piece of the proof above is a bound on the number of codewords of a given weight, based only on the minimum distance of the code. This bound is tight in some cases, such as for Reed-Solomon codes. However, it is quite loose in cases where the alphabet size is small. It is sufficient for our purpose: we are mainly interested in proving that reasonable families of codes exist (rather than trying to optimize the parameters).

7.1.4 Constructions of Small-Bias Families from Specific Codes

We can now use Lemma 7.8 to construct small-bias families from known code families.

Warm-up: Reed-Solomon-Based Constructions Reed-Solomon (RS) codes are a class of efficiently-decodable $[n', k', d]_q$ linear codes over a large alphabet: $q = 2^e$ must be at least n. They have distance $d = n' - k' + 1$ and, because the dual of a Reed-Solomon code is another Reed-Solomon code, they have dual distance $d_\perp = k' + 1$ (see, e,g., [53]).

Consider the family $\{C_i\}$ of binary images of a fixed RS code C'. By Lemma 7.8, the probability that a non-zero word a lies in the dual is at most $\delta^2 = (q-1)^{-d_\perp+1} = (q-1)^{-k'}$. Since $k < q$ and $(1-1/q)^q > 1/3$, we can in fact write $\delta^2 \leq 3q^{-k'} = 3 \cdot 2^{-k}$. Thus, binary images of RS codes (often called "generalized Reed-Solomon codes") have optimal bias: $\log\left(\frac{1}{\delta}\right) = k/2 - O(1)$, as with random linear codes, matching the lower bound Proposition 7.7.

Unfortunately, the conversion to a binary alphabet increases the code length and dimension without increasing the distance. Thus, these codes are only guaranteed to correct about $\frac{n-k}{2 \log n}$ errors. Nevertheless, for *large alphabets*, these codes do very well. That is, if the metric in which we care about error-correction for the sketch is Hamming distance in $GF(q)^{n'}$, then we get as good a secure sketch as possible, with as small a bias as possible.

Proposition 7.9 (RS-based Families for Large Alphabets). *For all $k < n \leq q = 2^e$, there exists a family $\{C_i\}$ of $[n, k, d]_q$ linear codes for $q \geq n$ with bias $\delta \leq 2^{-k/2+1}$, correcting $\tau \geq \frac{n-k}{2}$ errors efficiently.*

Algebraic-Geometric Constructions We now turn to our main construction. Our starting point is a construction of "algebraic-geometric" (AG). We get binary codes with exponentially small bias and linear minimum distance. We will need the following fact:

> [Algebraic-geometric codes] Let $q \geq 4$ be an even power of a prime, $q \geq 16$. There exists an infinite ensemble of $[n', k', d]_q$ linear codes C' (over $GF(16)$) with minimum distance at least $d = n' - k' - \frac{n'}{\sqrt{q}-1}$ and dual minimum distance $d' \geq k' - \frac{n'}{\sqrt{q}-1}$. Moreover, these codes have efficient algorithms for decoding up to $\lfloor (d-1)/2 \rfloor$ errors.

This follows from well-known bounds on algebraic-geometric codes (see, e.g., [80], section II.2). The main fact we need is that the dual of an AG code is an AG code for the same curve, and the distance of an AG code is bounded below by $n - k + 1 - g$, where g is the genus of the underlying curve. For infinitely many n, there exist curves over $GF(16)$ with n' points and genus at most $n'/3$.

We can now prove Lemma 7.6, which we restate here:

Lemma 7.10 (Good Code Families). *For any constant rate $R = k/n$, there exists an explicitly constructible ensemble of code families which* efficiently *correct $\tau = \Omega(n)$ errors and have bias* $\log\left(\frac{1}{\delta}\right) = \Omega(n)$.

In fact, the codes can be made arbitrarily close to optimal, at some cost in error-correction. That is, for any $\gamma > 0$, we can have $\log\left(\frac{1}{\delta}\right) > k/2(1-\gamma)$ but the fraction of errors corrected τ/n will decrease with γ.

Proof. Suppose that $R > 1/2$ (this is the interesting case, since it corresponds to small entropy loss; the case $R < 1/2$ is similar). Let q be any (constant) even power of a prime. By the facts above on AG codes, there exist $[n', k', d]_q$ codes with rate $k'/n' = R$, minimum distance at least $d = n'(1 - R - \frac{1}{\sqrt{q}-1}) \geq n'(1-R)/2$, and dual distance $d_\perp \geq n'(R - \frac{1}{\sqrt{q}-1}) \geq n'(3R-1)/2$.

We can now apply Corollary 7.5 to get a family of codes which correct τ errors and have bias δ, where:

$$\begin{aligned} \tau &\geq \frac{n}{\log q}\left(1 - R - \frac{1}{\sqrt{q}-1}\right) \\ &\text{and} \\ \log\left(\tfrac{1}{\delta}\right) &\geq \frac{1}{2}(d_\perp - 1)(\log(q-1)) \\ &= \frac{n}{2}\left(R - \frac{1}{\sqrt{q}-1}\right)\left(1 - \frac{\log q}{q-1}\right) \\ &= \frac{k}{2}\left(1 - \frac{1}{R(\sqrt{q}-1)}\right)\left(1 - \frac{\log q}{q-1}\right) \end{aligned}$$

By choosing q to be large enough (but constant), we get codes with constant error-correction rate and exponentially small bias, as desired. In fact, we can get $\log\left(\frac{1}{\delta}\right)$ as close as we want to the bound of Proposition 7.7, at some price in error-correction. □

Combining Lemma 7.10 with Corollary 7.5 proves Theorem 7.1. We conclude the section with an example.

Example: Consider the case $k = n/2$. Using Lemma 7.8, we get a ensemble of efficiently decodable binary codes with linear minimum distance $d = \frac{n}{4}\left(\frac{1}{2} - \frac{1}{3}\right) = \frac{n}{24}$, with squared bias $\delta^2 = \Pr_i[a \in C_i^{\perp}] \leq (1/15)^{d'-1} \leq 2^{-n/7}$. Applying now Corollary 7.5, we get an efficient entropically secure construction tolerating linear number of errors:

> There exists efficient $(n, t, t - \frac{n}{2}, \frac{n}{48})$ secure sketches with entropy loss $n/2$, correcting a linear number of errors $\tau = n/48$ and which are (t, ϵ) entropically secure as long as $t \geq \frac{6}{7}n + 2\log\left(\frac{1}{\epsilon}\right)$. In particular, one can have $n - t = \Omega(n)$ without losing either secrecy or error-tolerance.

Remark 7.2. There are various places in this section where the analysis can be improved substantially. First of all, the bounds on weight enumerators in the proof of Lemma 7.8 can be improved when the alphabet size q is a constant. Second, there are recent, better constructions of AG codes than those used in Fact 7.1.4. However, those codes are not linear (!), and as far as we know good bounds do not yet exist on their bias.

7.1.5 Secrecy for Fuzzy Extractors

Recall that for secure sketches, we required that $Y(W) = \mathsf{SS}(W)$ be entropically secure. For fuzzy extractors, we will in fact require that the *pair* $Y(W) = \langle P, Z\rangle$ satisfy the definition of security. This is somewhat counter-intuitive: we think of P as being published and Z as being used as a secret key in some other application. However, we cannot guarantee that no information about Z will be leaked in the other application (indeed, if Z is used to encrypt a known string it may be leaked completely). Requiring that the pair $\langle P, Z\rangle$ be entropically secure protects against arbitrary information being revealed about Z.

Nevertheless, if we consider fuzzy extractors built from a sketch scheme and a hash family (Lemma 3.1), then the requirement that $\langle Z, P\rangle$ be entropically secure reduces to the requirement that $\mathsf{SS}(W)$ be entropically secure. The following lemma follows from a standard hybrid argument:

Lemma 7.11. *Suppose that* SS *is a secure sketch with entropy loss* $t - t'$*, and* H *is drawn from a 2-universal hash family from* n *bits to* $t' - 2\log\left(\frac{1}{\epsilon}\right)$ *bits. Let* $P = \langle H, \mathsf{SS}(W)\rangle$ *and* $Z = H(W)$ *as in Lemma 3.1.*

If $Y_1(W) = \mathsf{SS}(W)$ *is* (t, ϵ)*-indistinguishable, then* $Y_2(W) = \langle P, Z\rangle$ *is* $(t, 2\epsilon)$*-indistinguishable.*

Hence, it is sufficient to build secure sketch schemes which are entropically secure—the resulting fuzzy extractors will inherit the property.

7.2 Perfectly One-Way Hash Functions

"Perfectly one-way" hash functions (POWFs) were introduced by Canetti [17] to attempt the formalize the common intuition that cryptographic hash functions reveal very little about their input. We will adopt the somewhat simplified version of the definition used in the subsequent paper of Canetti, Micciancio and Reingold [18]; see [17, 18] for further motivation and discussion.

Informally, POWFs are *randomized* hash functions $w \mapsto H(w; R)$ which satisfy two properties:

1. Given w and y, one can verify that $y = H(w; r)$ for some value of the randomness r. This means that a computationally bounded adversary cannot produce a pair $w' \neq w$ which would pass the same test.

2. If R is random, then $H(w; R)$ reveals no information about w.

The intuition that the hash leaks no information about the input was formalized using a definition almost identical to entropic security for predicates. Thus, we can apply the results of this thesis to the problem of designing perfectly one-way hash functions.

Contributions Our results apply in two different ways:

1. We show how to construct "fuzzy"—that is, noise-resilient—perfect hash functions. The hash value for w allows one to verify whether a candidate string w' is close to w, but reveals nothing else about w.

   This is a significant departure from the approach of Canetti *et al.* The motivation behind [17, 18] was to formalize the properties of an ideal "random oracle" which might be achievable by a real computer program. In contrast, even given a random oracle, it is not at all clear how to construct a *proximity* oracle for a particular value w (i.e. an oracle that accepts an input if and only if it is sufficiently close to w).

   In that sense, the result is also about *code obfuscation:* noise-resilient POWFs might best be viewed as weakly obfuscated versions of a proximity oracle (this is all the more interesting since strong obfuscation is not possible, see [6]).

2. We strengthen the results of [18] on information-theoretically-secure POWF's. First, following Chap. 5, we strengthen the definition of perfect one-way-ness to preclude the adversary from improving her ability to predict any *function* whatsover of the input when she sees the hash value.

   Second, we reduce the assumptions necessary for security: Canetti, Micciancio and Reingold [18] assume the existence of a collision-resistant hash function with an extra combinatorial property—*regularity* (a.k.a. balancedness)—in order for their proof of security to g through. We show how to modify the proof so the extra condition is unnecessary.

Finally, we improve the parameters of the [18] construction, roughly halving the requirement on the min-entropy of the input for the same level of security.

7.2.1 Definitions of Perfect One-way-ness

Recall the two informal conditions on PWOF's. Formalizing the first requirement is simple, though we note that the hash function requires a key in order to get full collision resistance. We denote by R_n the space of random coins required by the hash, and by K_n the space of keys (for input lengths n). A family of keyed randomized hash function $H^{(n)}$ with input length n and output length $\ell(n)$ is a family of functions $\left\{H_k : \{0,1\}^n \times R_n \to \{0,1\}^{\ell(n)}\right\}_{k \in K_n}$. An ensemble of such functions $\mathcal{H} = \left\{H^{(n)}\right\}_{n \in \mathbb{N}}$ consists of one such family for every input length n.

Definition 7.3 ([17, 18]). *A ensemble of keyed randomized functions $\mathcal{H} = \{H_k\}_{k \in K_n, n \in \mathbb{N}}$ as above is* publicly verifiable *if there is a polynomial-time verification algorithm* Ver *such that*

- *For all keys $k \in K_n$, inputs $w \in \{0,1\}^n$, and strings $r \in R_n$, $\mathsf{Ver}(k, w, H_k(w; r)) =$* ACC.
- *For any PPT adversary $\mathcal{A}$, the probability over $k \in K_n$ that $\mathcal{A}(k)$ outputs a triple (w, y, c) such that $\mathsf{Ver}(k, w, c) = \mathsf{Ver}(k, y, c) =$* ACC *is negligible in n.*

The intuition that the hash leaks no information about the input was formalized using a definition almost identical to entropic security for predicates. Given the equivalence of entropic security with respect to predicates and functions, we formulate the definition in terms of functions.

The main difference was that in the definitions of [17, 18], the adversary's ability to predict a predicate $g(W)$ given the (randomized) hash value $H(w) = H_k(w; R))$ is compared the adversary's ability to predict $g(W)$ given only polynomially many accesses to an *identity oracle* $Id_w(\cdot)$ which answers outputs "yes" on input w and "no" on any other input.

We'll say an adversary $\mathcal{A}^{\mathcal{O}(\cdot)}$ with access to an oracle $\mathcal{O}(\cdot)$ is poly-limited if there is some polynomial $p(\cdot)$ such that on inputs of length n, the adversary makes at most $p(n)$ queries to the oracle. A ensemble $\{W_n\}_{n \in \mathbb{N}}$ of $t(n)$-sources consists of distributions on $\{0,1\}^n$ with min-entropy at least $t(n)$.

Definition 7.4 (Perfect One-Way-ness, [17, 18]). *A ensemble of keyed randomized functions $\mathcal{H} = \{H_k\}_{k \in K_n, n \in \mathbb{N}}$ is $(t(n), \epsilon(n))$-perfectly one-way if for every adversary $\mathcal{A}$, for every ensemble $\{W_n\}_{n \in \mathbb{N}}$ of $t(n)$-sources, and for every function $f : \{0,1\}^* \to \{0,1\}^*$, there exists a poly-limited oracle adversary $\mathcal{A}^{\cdot}_*$ such that, for every n and $k \in K_n$:*

$$\Pr_{w \leftarrow W_n, r \leftarrow R_n}[\mathcal{A}(H_k(w; r)) = f(w)] - \Pr_{w \leftarrow W_n, r \leftarrow R_n}[\mathcal{A}^{Id_w(\cdot)}(1^n)) = f(w)] \leq \epsilon(n)$$

Note that adding the identity oracle makes no significant difference when the min-entropy of W is very high and hence the chance that the adversary queries the oracle on W is negligible. We encapsulate the equivalence as follows:

> If, for sufficiently large $n \in \mathbb{N}$ and all $k \in K_n$, the map $Y(w) = H_k(w; R)$ is $(t(n), \epsilon(n))$-entropically secure, then the hash function ensemble is $(t(n), \epsilon(n))$-semantically perfectly one-way.
>
> Moreover, the converse is true when $t > \log\left(\frac{1}{\epsilon}\right) + 1$ (since then the adversary has probability at most $\epsilon/2$ of succeeding to use the oracle).

Despite this equivalence, the formulation in terms of the identity oracle makes sense in the context, since the public verifiability makes one able to verify if a particular value is indeed w. We retain the "oracle" flavor in the definition of noise-resilient POWFs.

Noise-resilient POWFs

We can now define the new primitive which we construct in this section. A *proximity* oracle $B_{w,\tau}(\cdot)$ accepts its input w' if and only if the distance between w and w' is less than τ. Implicit here is a measure of distance between strings. We will only discuss constructions for the Hamming distance, but we formulate the definitions in more generality. We will assume that the distance function $\mathsf{dis}(,)$ is in fact a metric (that is, it satisfies the triangle inequality) on the space $\{0,1\}^*$. For simplicity we also assume that the distance between strings of different lengths is $+\infty$.

An ensemble of hash functions is called a *one-time* $(t(n), \epsilon(n), \tau(n))$-noise-resilient POWF (in the space $\mathsf{dis}(\cdot,\cdot)$) if it satisfies the following two conditions:

Definition 7.5 (Proximity Verifiability). *A ensemble of keyed randomized functions $\mathcal{H} = \{H_k\}_{k \in K_n, n \in \mathbb{N}}$ is* $(\mathsf{dis}(,), \tau(n))$-publicly proximity-verifiable *if there is a polynomial-time verification algorithm* Ver *such that*

- *For all pairs of inputs $w, w' \in \{0,1\}^n$ such that $\mathsf{dis}(w, w') \leq \tau(n)$, keys $k \in K_n$, and strings $r \in R_n$, $\mathsf{Ver}(k, w, H_k(w; r)) = \text{ACC}$.*
- *For any PPT adversary $\mathcal{A}$, the probability over $k \in K_n$ that $\mathcal{A}(k)$ outputs a triple $(w, \tilde{w}, c)$ such that $\mathsf{Ver}(k, w, c) = \mathsf{Ver}(k, \tilde{w}, c) = \text{ACC}$ and $\mathsf{dis}(w, \tilde{w}) \geq 2\tau(n)$ is negligible in n.*

Definition 7.6 (Proximity-Semantic-Security). *A ensemble of keyed randomized functions $\mathcal{H} = \{H_k\}_{k \in K_n, n \in \mathbb{N}}$ is $(t(n), \epsilon(n))$-semantically perfectly one-way for $(\mathsf{dis}(,), \tau(n))$ if for every adversary $\mathcal{A}$, for every ensemble $\{W_n\}_{n \in \mathbb{N}}$ of $t(n)$-sources, and for every function $f : \{0,1\}^* \to \{0,1\}^*$, there exists a poly-limited oracle adversary $\mathcal{A}^{\cdot}_*$ such that, for every n and $k \in K_n$:*

$$\Pr_{w \leftarrow W_n, r \leftarrow R_n}[\mathcal{A}(H_k(w; r)) = f(w)] - \Pr_{w \leftarrow W_n, r \leftarrow R_n}[\mathcal{A}^{B_{w,\tau(n)}(\cdot)}(1^n) = f(w)] \leq \epsilon(n)$$

where $B_{w,\tau}(\cdot)$ is the proximity oracle which accepts its input w' iff $\mathsf{dis}(w, w') \leq \tau$.

Unlike in the case of an identity oracle, proving that the proximity orace is not useful to the adverary requires much stronger bounds on the initial value of the min-entropy t. See the proof of security of the main construction, below.

Note: All the constructions discussed in this chapter are *one-time* secure, that is they provide no guarantee of secrecy when many hashes of the same input are given to the adversary. In fact, for any constant t, it is fairly easy to get t-time security (depending on the exact parameters). However, in the case of ordinary (non-noise-resilient) POWFs, [18] showed that it is possible to construct poly-time secure function ensembles (that is, the adversary may see as many hashes of the same input w as she wishes, and then try to predcit some predicate of w). Generalizing the result to noise-resilient POWFs is a fascinating open problem.

7.2.2 Constructing Noise-resilient POWFs

Our main construction is stated below. The basic idea is that entropically-secure secure sketches compose well with any ordinary POWF, as long as we have a guarantee. Note that the proof we give here applies only to the Hamming metric (we will use lower bounds on the volume of balls in the Hamming cube).

Theorem 7.12 (Generic Construction). *Suppose that*

- $\{\mathsf{SS}_n\}_{n\in\mathbb{N}}$ *is an ensemble of* $(n, t-1, t', \tau)$ *sketches which are* (t,ϵ)*-entropically secure,*
- $\{H_k\}_{k\in K_n, n\in\mathbb{N}}$ *is a (ordinary) POWF as defined above which is* $(t' - \log\left(\frac{1}{\epsilon}\right) + 1, \epsilon)$*-perfectly one-way,*

Then the ensemble $\{H'_k\}_{k\in K_n, n\in\mathbb{N}}$ *of randomized hash functions given by*

$$\tilde{H}_k(w; \underbrace{r_1, r_2}_{r}) \stackrel{def}{=} \mathsf{SS}(w; r_1), H_k(w; r_2)$$

is τ*-proximity-verifiable and* $(t+1, 2\epsilon)$*-perfectly one-way. (Here* t, t', τ, ϵ *are functions of* n*.)*

Proof. The fact that the construction in the preceding theorem is *proximity-verifiable* is easy to check. Given a candidate string $\tilde{w}$, and a string (s, c) which is a correctly generated hash of w, then the verification alogrithm $\mathsf{Ver}'(k, \tilde{w}, (s, c))$ does the following (a) Run the recovery procedure for $\mathsf{SS}()$ on the pair $(\tilde{w}, s)$, get back a candidate string w' for w, and (b) check if $\mathsf{dis}(\tilde{w}, w') \leq \tau$ and $\mathsf{Ver}(k, w', c) = \text{ACC}$, where $\mathsf{Ver}()$ is the verification function for the original (nonfuzzy) POWF.

If $\tilde{w}$ is indeed close to w, then this test will always succeed. [2] On the other hand, if a poly-time adversary can produce a values $\tilde{w}, \tilde{z}$ which both pass verification with the

[2] Similarly, if the sketch only corrects errors with high probability, then the test will succeed with high probability, achieving a sligthly relaxed version of the definition of verifiability.

same string c, then there are corresponding values w, z within distance τ of $\tilde{w}$ (resp. $\tilde{z}$) such that $\mathsf{Ver}(k, w, c) = \mathsf{Ver}(k, z, c) = \text{ACC}$. By the verifiability of the original POWF-scheme, it must be that $w = z$, and so $\mathsf{dis}(\tilde{w}, \tilde{z}) \leq \mathsf{dis}(\tilde{w}, w) + \mathsf{dis}(z, \tilde{z}) \leq 2\tau$, as desired.

We now turn to the proof that the scheme is semantically perfectly one-way in the sense of Definition 7.6. We'll use the following general lemma on composing entropically-secure maps:

Lemma 7.13. *If (1): $Y_1()$ is a (t, ϵ)-entropically-secure map, (2): for all distributions W of min-entropy at least $t-1$ we have $\tilde{\mathbf{H}}_\infty(W \mid Y_1(W)) \geq t'$ and (3): $Y_2()$ is a $(t' - \log\left(\frac{1}{\epsilon}\right) + 1, \epsilon)$ secure map, then the map which outputs the pair $Y(w) = Y_1(w), Y_2(w)$ is $(t+1, 2\epsilon)$-entropically-secure.*

The lemma can be proven using a simple hybrid argument (see below). For now, we can use it to complete the proof of security of the noise-resilient POWF. Let $Y_1 = \mathsf{SS}()$ and $Y_2 = H_k()$. By the definition of a secure sketch and the hypotheses of the theorem statement, the conditions of the lemma are satisfied, and we get that the map $H'_k(\cdot; R)$ is $(t+1, 2\epsilon)$-entropically-secure. Entropic security implies semantic perfect one-way-ness with the same parameters. □

We can now prove the composition lemma used above:

Proof of Lemma 7.13. The proof follows a careful hybrid argument. In order to prove $t+1$-entropic security, we will prove $t-1$-indistinguishability and then apply the equivalence (Theorem 5.1). Suppose that W has min-entropy at least $t-1$. With probability $1-\epsilon$ over the values of $\mathsf{SS}(W)$, the min-entropy $\mathbf{H}_\infty(W \mid \mathsf{SS}(W))$ will be at least $te' - \log\left(\frac{1}{\epsilon}\right)$ (recall that $t' = \tilde{\mathbf{H}}_\infty(W \mid Y_1(W))$, so $2^{-t'}$ is the average value of $2^{-\mathbf{H}_\infty(W|Y_1(W))}$). Since $Y_1(W)$ is $t' - \log\left(\frac{1}{\epsilon}\right) + 1$-entropically secure, it is $\left(t' - \log\left(\frac{1}{\epsilon}\right), 4\epsilon\right)$ indistinguishable and so with probability $1-\epsilon$ (over values of $Y_1(W)$), the statistical difference between $Y_1(W), Y_2(W)$ and $Y_1(W), Y_2(U_n)$ is at most 4ϵ. Hence, the overall statistical difference between the two distributions is at most 5ϵ. Finally, the distance between $Y_1(W), Y_2(U_n)$ and $Y_1(U'_n), Y_2(U_n)$ is at most 4ϵ since $Y_1()$ is $(t-1, 4\epsilon)$-indistinguishable. By the triangle inequality, the distance between $Y_1(W), Y_2(W)$ and $Y_1(U'_n), Y_2(U_n)$ is at most 9ϵ, and so the scheme is $(t-1, 9\epsilon)$-indistinguishable. Applying Theorem 5.1 in the other direction completes the proof. □

7.2.3 Improved Construction of Ordinary POWFs

Before we can apply the generic construction of the previous section, we need to constructions of ordinary, non-noise-resilient POWF's.

Canetti et al. [18] gave the following simple construction of perfect one-way hash functions which achieves (information-theoretic) entropic secrecy. Given a family

of "regular" collision-resistant hash functions $\{\mathsf{crhf}_k\}_{k\in K_n}$, and a family of pairwise independent *permutations* $\{\pi_i\}_{i\in\mathcal{I}}$, we can define a probabilistic map

$$H_k(w;i) = i, \mathsf{crhf}_k(\pi_i(w)).$$

[18] proved that the construction is (t,ϵ)-entropically secure as long as the output length $\ell(n)$ of the functions crhf_k satisfies $\ell(n) \leq (t - 2logeps)/2$. Their analysis also required an additional assumption on the crhf, namely that the functions be "regular" (a.k.a. balanced), that is for all k, every point in the image of crhf_k must have the same number of pre-images.

Here we improve on the analysis in several ways. First, we remove the assumption of regularity. This is based on a version of the left-over hash lemma in which a pairwise independent hash function is fed through an arbitrary function before producing output (Lemma A.2). Second, we improve the parameters: we show that their construction only requires $\ell(n) \leq t - 2\log\left(\frac{1}{\epsilon}\right)$ (that is, we may leak twice as many bits about the input without compromising entropic security). Finally, we provide a stronger security guarantee, namely that the adversary may not learn any *function* of the input. We encapsulate these improvements in the following proposition.

Proposition 7.14. *Suppose that*

- $\{\mathsf{crhf}_k(\cdot)\}_{k\in K_n, n\in\mathbb{N}}$ *is a collision-resistant hash family from* n *bits to* $\ell(n)$ *bits,*
- $\ell < t - 2\log\left(\frac{1}{\epsilon}\right)$,
- $\left\{\{\pi_i\}_{i\in\mathcal{I}}\right\}_{n\in\mathbb{N}}$ *is an ensemble of XOR-universal* permutations *of* $\{0,1\}^n$.

Then the ensemble of randomized hash functions given by: $H_k(w;i) = i, \mathsf{crhf}_k(\pi_i(w))$ *is* (t,ϵ)*-entropically secure. (Here* $t, t', \tau, \ell, \epsilon$ *are all functions of* n*.)*

To prove entropic security, it suffices to prove that the scheme is indistinguishable. The statement follows directly from a variant of the left-over hash lemma (Lemma A.2), which basically states that combining XOR-independent permutations with any arbitrary functions yields a "crooked" strong extractor: that is, the output may not be look random, but it will look the same for all input distributions of sufficiently high entropy. Contrary to intuition, this statement does *not* follow directly from the left-over hash lemma.

7.2.4 Putting It All Together

We can now combine the results of this chapter so far. Our initial goal was a non-trivial family of noise-resilient POWF's. As mentioned above, these can be viewed as obfuscated code for proximity queries. We would like to combine Theorem 7.1 with the generic constructions of this section. For this purpose, we will use the fact that if there are length-reducing collision-resistant hash functions, then for for any output length $\ell(n) = \Omega n$, there exists a hash family $\{\mathsf{crhf}_k\}_{k\in K_n, n\in\mathbb{N}}$ with output length $\ell(n)$ for which no PPT adversary can find collisions with non-negligible probability. We obtain:

Theorem 7.15. *If collision-resistant hash functions exist, then for any initial entropy $t = \Omega(n)$, there exists a noise-resilient POWF ensemble which tolerates a linear number of errors $\tau = \Omega(n)$, is (t, ϵ)-entropically-secure for $\epsilon = 2^{-\Omega(n)}$ and is proximity-publicly verifiable with negligible soundness error.*

We conclude with a caveat and a question. The noise-resilient POWF's of Theorem 7.15 are only one-time secure, that is they do not bear up to revealing many hashes of the same secret input.

> Is it possible to construct poly-time entropically-secure *noise-resilient* POWF's (for which an arbitrary polynomial number of hashes of the same input may be revealed)?

Bibliography

[1] E. Agrell, A. Vardy, and K. Zeger. Upper bounds for constant-weight codes. *IEEE Transactions on Information Theory*, **46**(7), pp. 2373–2395, 2000. Cited on p. 33, 45, 47

[2] Noga Alon, Oded Goldreich, Johan Håstad, René Peralta: Simple Constructions of Almost k-Wise Independent Random Variables. FOCS 1990: 544-553. Cited on p. 87

[3] Noga Alon and Yuval Roichman. Random Cayley graphs and expanders. *Random Structures & Algorithms* 5 (1994), 271–284. Cited on p. 87

[4] A. Andoni, M. Deza, A. Gupta, P. Indyk, S. Raskhodnikova. Lower bounds for embedding edit distance into normed spaces. In *Proc. ACM Symp. on Discrete Algorithms, 2003*, pp. 523–526. Cited on p. 18, 52

[5] C. Barral and J.-S. Coron and D. Naccache. Externalized Fingerprint Matching. Cryptology ePrint Archive, Report 2004/021, 2004. Cited on p. 19

[6] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, K. Yang. On the (Im)possibility of Obfuscating Programs. In *Advances in Cryptology — CRYPTO 2001*, pp. 1–18. Cited on p. 104

[7] C. Bennett, G. Brassard, and J. Robert. Privacy Amplification by Public Discussion. *SIAM J. on Computing*, **17**(2), pp. 210–229, 1988. Cited on p. 19, 20, 32, 39, 41, 87, 88

[8] C. Bennett, G. Brassard, C. Crépeau, and U. Maurer. Generalized Privacy Amplification. *IEEE Transactions on Information Theory*, **41**(6), pp. 1915-1923, 1995. Cited on p. 19, 31, 32, 39, 41, 88

[9] Eli Ben-Sasson, Madhu Sudan, Salil P. Vadhan, Avi Wigderson: Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. STOC 2003: 612-621 Cited on p. 84, 87

[10] R.M. Bolle, J.H. Connell, S. Pankanti, N.K. Ratha, A.W. Senior. *Guide to Biometrics*, Springer Professional Computing Series, 2004, 364 p. Cited on p. 13

[11] B. Bollobás. *Combinatorics*. Cambridge University Press, 1986. Cited on p. 64

[12] X. Boyen. Reusable Fuzzy Extractors. In *ACM CCS 2004*. Cited on p. 20

[13] A. Broder. On the resemblence and containment of documents. In *Compression and Complexity of Sequences*, 1997. Cited on p. 53

[14] A. E. Brouwer, J. B. Shearer, N. J. A. Sloane, and W. D. Smith, "A new table of constant weight codes," *IEEE Transactions on Information Theory*, **36**, p. 1334–1380, 1990. Cited on p. 45

[15] BBC News. "Facing a biometric future," 13 January, 2004. "US passport deadline is extended," 27 July 2004. Cited on p. 13

[16] Andrew Burnett and Adam Duffy and Tom Dowling. A Biometric Identity Based Signature Scheme. Unpublished manuscript, 2004. `http://eprint.iacr.org/2004/176` Cited on p. 20

[17] R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *Advances in Cryptology — CRYPTO 1997*. Cited on p. 21, 22, 23, 24, 69, 70, 91, 104, 105

[18] R. Canetti, D. Micciancio, O. Reingold. Perfectly One-Way Probabilistic Hash Functions. In *Proc. 30th ACM Symp. on Theory of Computing*, 1998, pp. 131–140. Cited on p. 21, 22, 24, 69, 70, 91, 104, 105, 107, 108, 109

[19] Ran Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. *Proc. IEEE Symp. on Foundations of Computer Science*, 2001, pp. 136-145. Cited on p. 25

[20] J. L. Carter, M. N. Wegman. Universal Classes of Hash Functions. *Journal of Computer and System Sciences*, 18, 1979, pp. 143–154. Cited on p. 31

[21] T. Clancy, N. Kiyavash, D. Lin. Secure Smartcard-Based Fingerprint Authentication. In *Proc. of the 2003 ACM SIGMM workshop on Biometric Methods and Applications*. `http://www.cs.umd.edu/~clancy/docs/bio-wbma2003.pdf` Cited on p. 19

[22] G. Cohen, G. Zémor. Generalized Coset Schemes for the Wire-Tap Channel: Application to Biometrics. In *International Symp. on Information Theory*, June 2004. Cited on p. 19, 55

[23] T. Cover, J. Thomas. *Elements of Information Theory*. Wiley series in telecommunication, 1991, 542 pp. Cited on p. 30

[24] C. Crépeau. Efficient Cryptographic Protocols Based on Noisy Channels. In *Advances in Cryptology — EUROCRYPT 1997*, pp. 306–317. Cited on p. 19, 41

[25] L. Csirmaz and G.O.H. Katona. Geometrical Cryptography. In *Proc. International Workshop on Coding and Cryptography*, 2003. Cited on p. 14, 15, 19

[26] G. Davida, Y. Frankel, B. Matt. On enabling secure applications through off-line biometric identification. In *Proc. IEEE Symp. on Security and Privacy*, pp. 148–157, 1998. Cited on p. 18

[27] G.I. Davida, Y. Frankel, B.J. Matt and R. Peralta. On the relation of error correction and cryptography to an online biometric based identication scheme. In Proceedings of WCC99, Workshop on Coding and Cryptography, 1999. Cited on p. 18

[28] M. van Dijk, D. Woodruff. Manuscript, 2004. Cited on p. 20

[29] Y.Z. Ding. Manuscript. Cited on p. 20

[30] Y. Dodis, L. Reyzin and A. Smith. Fuzzy Extractors and Cryptography, or How to Use Your Fingerprints. In *Advances in Cryptology — EUROCRYPT 2004*. Originally appeared as IACR Eprint Report 2003/235, November 2003. Cited on p. 6

[31] Y. Dodis and A. Smith. Entropic Security and the Encryption of High-Entrop Messages. Manuscript, 2004. Cited on p. 6

[32] Y. Dodis and A. Smith. Obfuscating Proximity Queries: Fuzzy Extractors Which Hide All Partial Information. Manuscript, 2004. Cited on p. 6

[33] Ilya Dumer, Daniele Micciancio, and Madhu Sudan. Hardness of approximating the minimum distance of a linear code, *IEEE Transactions on Information Theory*, **49**(1), pp. 22–37, 2003. Cited on p. 98

[34] Electronic Privacy and Information Center (EPIC). Web page on Biometric Identifiers. `http://www.epic.org/privacy/biometrics/` Cited on p. 13

[35] Electronic Privacy Information Center (EPIC). Web page on US-VISIT program. `http://www.epic.org/privacy/us-visit/` Cited on p. 13

[36] C. Ellison, C. Hall, R. Milbert, B. Schneier. Protecting Keys with Personal Entropy. *Future Generation Computer Systems*, **16**, pp. 311–318, 2000. Cited on p. 18

[37] D. Forney. *Concatenated Codes*. MIT Press, 1966. Cited on p. 55

[38] N. Frykholm. Passwords: Beyond the Terminal Interaction Model. *Master's Thesis*, Umea University. Cited on p. 14

[39] N. Frykholm, A. Juels. Error-Tolerant Password Recovery. In *Proc. ACM Conf. Computer and Communications Security, 2001*, pp. 1–8. Cited on p. 12, 18

[40] S. Goldwasser and S. Micali. Probabilistic encryption. *JCSS*, **28**(2), pp. 270–299, April 1984. Cited on p. 21, 22, 23, 25, 26, 69, 71, 72, 82, 84

[41] Oded Goldreich, Avi Wigderson: Tiny families of functions with random properties: A quality-size trade-off for hashing. Random Structures and Algorithms 11(4): 315-343 (1997) Cited on p. 86

[42] Venkatesan Guruswami. List Decoding with Side Information. In *IEEE Conference on Computational Complexity* 2003, p.300-309. Cited on p. 55, 56

[43] V. Guruswami, M. Sudan. Improved Decoding of Reed-Solomon and Algebraic-Geometric Codes. In *Proc. 39th IEEE Symp. on Foundations of Computer Science*, 1998, pp. 28–39. Cited on p. 56

[44] J. Håstad, R. Impagliazzo, L. Levin, M. Luby. A Pseudorandom generator from any one-way function. In *Proc. 21st ACM Symp. on Theory of Computing*, 1989. Cited on p. 19, 20, 31, 32, 39, 88

[45] Jonathan Herzog. *Computational Soundness for Standard Assumptions of Formal Cryptography*. Ph.D. Thesis, Massachusetts Institute of Technology, May 2004. Cited on p. 25

[46] R. Impagliazzo and D. Zuckerman. How to Recycle Random Bits. In *Proc. 30th IEEE Symp. on Foundations of Computer Science*, 1989. Cited on p. 29, 32, 85, 87, 122

[47] A. Juels, M. Wattenberg. A Fuzzy Commitment Scheme. In *Proc. ACM Conf. Computer and Communications Security, 1999*, pp. 28–36. Cited on p. 17, 18, 41

[48] A. Juels and M. Sudan. A Fuzzy Vault Scheme. In *IEEE International Symposium on Information Theory*, 2002. Cited on p. 12, 17, 19, 43, 45

[49] J. Kelsey, B. Schneier, C. Hall, D. Wagner. Secure Applications of Low-Entropy Keys. In *Proc. of Information Security Workshop*, pp. 121–134, 1997. Cited on p. 18

[50] H. Krawczyk. LFSR-Based Hashing and Authentication. In *Advances in Cryptology — CRYPTO '94*, p. 129–139, 1994. Cited on p. 31

[51] M. Langberg. Private codes or Succinct random codes that are (almost) perfect. *Proc. IEEE Symp. on Foundations of Computer Science*, 2004. Cited on p. 55, 56

[52] J.-P. M. G. Linnartz, P. Tuyls. New Shielding Functions to Enhance Privacy and Prevent Misuse of Biometric Templates. In *AVBPA 2003*, p. 393–402. Cited on p. 19

[53] J.H. van Lint. *Introduction to Coding Theory.* Springer-Verlag, 1992, 183 pp. Cited on p. 44, 48, 49, 51, 61, 62, 101

[54] C. Lu, O. Reingold, S. Vadhan and A. Wigderson. Extractors: Optimal Up to Constant Factors. In *Proc. ACM Symp. on Theory of Computing*, 2003. Cited on p.

[55] A. Lubotzky, R. Phillips, P. Sarnak: Ramanujan graphs. Combinatorica 8(3): 261-277 (1988). Cited on p. 84, 86

[56] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam, New York, Oxford, 1978. Cited on p. 94, 98, 100

[57] U. Maurer. Conditionally-Perfect Secrecy and a Provably-Secure Randomized Cipher. *J. Cryptology*, **5**(1), pp. 53–66, 1992. Cited on p. 20

[58] U. Maurer. Secret Key Agreement by Public Discussion. *IEEE Trans. on Info. Theory*, 39(3):733–742, 1993. Cited on p. 82

[59] Silvio Micali, Chris Peikert, Madhu Sudan, and David Wilson. Cryptographic Sieving: Optimal Error Correction Against Computationally Bounded Noise. Manuscript, 2004. Cited on p. 55, 56

[60] F. Monrose, M. Reiter, Q. Li, S. Wetzel. Cryptographic key generation from voice. In *Proc. IEEE Symp. on Security and Privacy*, 2001. Cited on p. 18

[61] F. Monrose, M. Reiter, Q. Li, S. Wetzel. Using voice to generate cryptographic keys. In *Proc. of Odyssey 2001, The Speaker Verification Workshop*, 2001. Cited on p. 18

[62] F. Monrose, M. Reiter, S. Wetzel. Password Hardening Based on Keystroke Dynamics. In *Proc. ACM Conf. Computer and Communications Security, 1999*, p. 73–82. Cited on p. 18

[63] R. Morris, K. Thomson. Password Security: a case history. In *Comm. ACM*, **22**(11), pp. 594–597, 1979. Cited on p. 14

[64] J. Naor, M. Naor. Small-Bias Probability Spaces: Efficient Constructions and Applications. In *SIAM J. Comput.* 22(4): 838-856 (1993). Cited on p. 83, 84, 87, 94

[65] New York Times. "Arrest in Bombing Inquiry Was Rushed, Officials Say", May 8, 2004. Cited on p. 13

[66] N. Nisan, D. Zuckerman. Randomness is Linear in Space. In *JCSS*, **52**(1), pp. 43–52, 1996. Cited on p. 16, 19, 84

[67] J. Radhakrishnan and A. Ta-Shma. Tight bounds for depth-two superconcentrators. In *Proc. 38th IEEE Symp. on Foundations of Computer Science*, 1997, pp. 585–594. Cited on p. 31, 84, 89, 90

[68] Enhancing security and privacy in biometrics-based authentication systems N. Ratha, J. Connell, R. Bolle IBM Systems Journal, vol. 40, no. 3, 2001, pp. 614-634. Cited on p. 19

[69] Ron Rivest. Lecture notes from lecture 21 of MIT course 6.857, Fall 2001. `http://web.mit.edu/6.857/OldStuff/Fall01/handouts`. Cited on p. 13

[70] A. Russell and Wang. How to Fool an Unbounded Adversary with a Short Key. In *Advances in Cryptology — EUROCRYPT 2002*. Cited on p. 21, 22, 23, 24, 69, 70, 83, 84, 85, 87

[71] Sagem Morpho Inc, "The History of Fingerprinting," `http://www.dia.unisa.it/professori/ads/corso-security/www/CORSO-9900/biometria/Fingerprinting.htm`. Cited on p. 13

[72] R. Shaltiel. Recent developments in Explicit Constructions of Extractors. *Bulletin of the EATCS*, **77**, pp. 67–95, 2002. Cited on p. 12, 31

[73] C. Shannon. Communication Theory of Secrecy systems. In *Bell Systems Technical J.*, 28:656–715, 1949. Note: The material in this paper appeared originally in a confidential report 'A Mathematical Theory of Cryptography', dated Sept. 1, 1945, which has now been declassified. Cited on p. 11, 24, 28, 81

[74] C. Shannon. A Mathematical Theory of Communication. *Bell System Technical J.*, 27 (July and October 1948), pp. 379-423 and 623-656. Reprinted in D. Slepian, editor, *Key Papers in the Development of Information Theory*, IEEE Press, NY, 1974. Cited on p. 55

[75] Bruce Schneier. Biometrics: Uses and Abuses.Iside Risks 110, *Comm. ACM*, 42(8), Aug. 1999. Cited on p. 13

[76] V. Shoup. A Proposal for an ISO Standard for Public Key Encryption. Available at `http://eprint.iacr.org/2001/112`, 2001. Cited on p. 16

[77] C. Soutar, D. Roberge, S. A. Stojanov, R. Gilroy, and B.V.K. Vijaya Kumar. Biometric Encryption (tm). Chapter 22 of ICSA Guide to Cryptography, ed. Randall Nichols, McGraw-Hill, 1999. `www.bioscrypt.com/assets/Biometric_Encryption.pdf` Cited on p. 18

[78] C. Soutar, D. Roberge, S. A. Stojanov, R. Gilroy, and B.V.K. Vijaya Kumar. Biometric encryption using image processing. Proc. of SPIE, Vol. 3314, 178-188, 1998. Cited on p. 18

[79] C. Soutar, D. Roberge, S. A. Stojanov, R. Gilroy, and B.V.K. Vijaya Kumar. Biometric encryption - Enrollment and Verification Procedures. Proc. of SPIE, Vol. 3386, 24-35, April 1998. Cited on p. 18

[80] H. Stichtenoth. Algebraic Function Fields and Codes. *Springer-Verlag*, Berlin, 1993. Cited on p. 102

[81] George Teomko. Biometrics as a Privacy-Enhancing Technology: Friend or Foe of Privacy? Presented at Privacy Laws & Business 9th Privacy Commissioners' / Data Protection Authorities Workshop, September 15th, 1998. http://www.dss.state.ct.us/digital/tomko.htm Cited on p. 13

[82] Pim Tuyls, Jasper Goseling. Capacity and Examples of Template-Protecting Biometric Authentication Systems. IACR Eprint 2004/106, May 2004. http://eprint.iacr.org/2004/106 Cited on p. 19, 55

[83] U. Uludag, S. Pankanti, S. Prabhakar and A. K. Jain. Biometric Cryptosystems: Issues and Challenges. Proc. of the IEEE, Special Issue on Multimedia Security for Digital Rights Management, vol. 92, no. 6, pp. 948-960, June 2004. http://biometrics.cse.msu.edu/Uludagetal_Cryptosystems_ProcIEEE04.pdf Cited on p. 19

[84] US Department of Homeland Security. US-VISIT (United States Visitor and Immigrant Status Indicator Technology) FAQs. See http://www.dhs.gov/dhspublic/interapp/editorial/editorial_0444.xml Cited on p. 13

[85] Salil Vadhan. Randomness Extractors and their Many Guises. Tutorial from FOCS 2002. http://www.eecs.harvard.edu/~salil/extractors-focs02.ppt Cited on p. 12

[86] Salil Vadhan (and students). Lecture Notes from Harvard course CS225. http://www.courses.fas.harvard.edu/~cs225/Lectures/ Cited on p. 12

[87] E. Verbitskiy, P. Tuyls, D. Denteneer, J.-P. Linnartz. Reliable Biometric Authentication with Privacy Protection. In *Proc. 24th Benelux Symposium on Information theory*, 2003. Cited on p. 19

[88] Mark N. Wegman, Larry Carter. New Hash Functions and Their Use in Authentication and Set Equality. *Journal of Computer and System Sciences*, 22(3), 265-279, 1981. Cited on p. 31

Appendix A

Variants on the Left-over Hash (Privacy Amplification) Lemma

Randomness extractors are key tools in this thesis, and we will use several variants on a classic construction of extractors, referred to as the "left-over hash" or "privacy amplication" lemma. These variants will be useful in our constructions of special-purpose extractors. We gather them here for easy reference, and because their proofs are very similar.

Recall the basic left-over hash lemma:

Lemma A.1 (Also Lemma 2.4). *If $\{h_i\}_{i\in\mathcal{I}}$ is a family of pairwise independent hash functions from n bits to ℓ bits, and X is a random variable in $\{0,1\}^n$ with Renyi entropy $\mathbf{H}_2(X) \geq \ell + 2\log\left(\frac{1}{\epsilon}\right) + 1$, then*

$$\langle I, h_i(X)\rangle \approx_\epsilon \langle I, U_\ell\rangle$$

where $\mathcal{I} \leftarrow \mathcal{I}$, $U_\ell \leftarrow \{0,1\}^\ell$ (both drawn uniformly), and I, X and U_ℓ are independent.

A.1 Composing Hashing with Arbitrary Functions

The first variant will be useful for composing the pairwise-independent extractor with a function whose design is beyond our control (a cryptographic hash function, for example).

The lemma deals with first hashing X into a string that may be very long (using pairwise-independent functions) and then shrinking its length (down to about $\mathbf{H}_2(X)$) using an arbitrary function.

Lemma A.2 (Composing with an arbitrary function). *Let $f : \{0,1\}^N \to \{0,1\}^\ell$ be an arbitrary function. If $\{h_i\}_{i\in\mathcal{I}}$ is a family of pairwise independent hash functions from n bits to N bits and X is a random variable in $\{0,1\}^n$ with Renyi entropy $\mathbf{H}_2(X) \geq \ell + \log\left(\frac{1}{\epsilon}\right) + 1$, then*

$$\langle I, f(h_i(X))\rangle \approx_\epsilon \langle I, f(U_N)\rangle$$

where $I \leftarrow \mathcal{I}$, $U_N \leftarrow \{0,1\}^N$ (both drawn uniformly), and I, X and U_N are independent.

This lemma requires a fresh proof—it does not follow directly from the original left-over hash lemma: because N may be much larger than n and $\mathbf{H}_2(X)$, the distributions $\langle I, h_I(X)\rangle$ and $\langle I, U_N\rangle$ need not be indistinguishable. In fact, when $N > n$ they will have statistical distance almost 1.

The idea behind the proof is to show that for all non-zero strings $\alpha \in \{0,1\}^\ell$, the inner product modulo two $\alpha \odot f(h_I(X))$ is distributed almost identically to $\alpha \odot f(U_N)$. Elementary Fourier analysis then shows that the distributions $f(h_I(X))$ and $f(U_N)$ are close (even given I). Details follow.

Proof. The bias of a distribution A over $\{0,1\}^\ell$ with respect to a string α is defined to be $\mathsf{bias}_\alpha(A) = \left|\ \mathbb{E}_A\left[(-1)^{\alpha\odot A}\right]\ \right| = |2\Pr[\alpha \odot A = 0] - 1|$.

The following fact about the hypercube $\{0,1\}^\ell$ will be useful below: For any random variables (distributions) A and B on $\{0,1\}^\ell$, we have:

$$\mathbf{SD}\left(A, B\right) \leq \sqrt{\sum_{\alpha\in\{0,1\}^\ell} \left(\mathsf{bias}_\alpha(A) - \mathsf{bias}_\alpha(B)\right)^2}. \tag{A.1}$$

Claim A.3. *For every $\alpha \in \{0,1\}^\ell$, the expectation, over $i \leftarrow \mathcal{I}$, of the expression*

$$\left(\mathsf{bias}_\alpha(f(h_i(X))) - \mathsf{bias}_\alpha(f(U_N))\right)^2$$

is at most $\mathsf{Col}(X) = 2^{-\mathbf{H}_2(X)} \leq \epsilon^2 2^{-\ell}$.

We first show that this claim implies the lemma, and then prove the claim further below. For every $i \in \mathcal{I}$, let $D_i = f(h_i(X))$. The first observation is that the distance we are seeking to bound is the average, taken over i, of the distance between D_i and the target distribution $f(h_i(X))$.

$$\mathbf{SD}\left(\langle I, D_I\rangle\ ,\ \langle I, f(U_N)\rangle\right) = \mathbb{E}_I\left[\mathbf{SD}\left(D_I, f(U_N)\right)\right]$$

We can now bound the statistical difference using the biases (Eqn. A.1):

$$\mathbf{SD}\left(\langle I, D_I\rangle\ ,\ \langle I, f(U_N)\rangle\right) \leq \mathbb{E}_I\left[\sqrt{\sum_\alpha \left(\mathsf{bias}_\alpha(D_I) - \mathsf{bias}_\alpha(f(U_N))\right)^2}\right]$$

For any random variable, $\mathbb{E}\left[\sqrt{X}\right] \leq \sqrt{\mathbb{E}\left[X\right]}$ (Jensen's inequality). Hence

$$\mathbf{SD}\left(\langle I, D_I\rangle\ ,\ \langle I, f(U_N)\rangle\right) \leq \sqrt{\sum_\alpha \mathbb{E}_I\left[\left(\mathsf{bias}_\alpha(D_I) - \mathsf{bias}_\alpha(f(U_N))\right)^2\right]}$$

By the main claim above, the term inside the square root sign is at most $\sum_\alpha \epsilon^2 2^\ell = \epsilon^2$, and so the statistical difference which we want to bound is at most ϵ^2. □

To complete the proof above, we just have to prove the claim.

Proof of Claim A.3. For $\alpha = 0^\ell$, the claim is trivial since the difference of biases is always 0. Fix $\alpha \neq 0^\ell$. Let

$$\mu = \mathsf{bias}_\alpha(f(U_N)) = \mathbb{E}_{U_N}\left[(-1)^{\alpha \odot f(U_N)}\right]$$

Let $p_x = \Pr[X = x]$. Then we can write $\mathsf{bias}\alpha(f(h_I(X))) - \mathsf{bias}_\alpha(f(U_N))$ as

$$\mathsf{bias}\alpha(f(h_I(X))) - \mathsf{bias}_\alpha(f(U_N)) = \sum_{x \in \{0,1\}^n} p_x \underbrace{\left((-1)^{\alpha \odot f(h_I(x))} - \mu\right)}_{Z_x}$$

Now let Z_x be the random variable $(-1)^{\alpha \odot f(h_I(x))} - \mu$ (this is a function of I). Since $\{h_i\}$ is a pairwise independent family of hash functions, the expectation of Z_x taken over I is exactly 0 (that is, for any fixed x, $h_I(x)$ is uniformly distributed over $\{0,1\}^N$). Moreover, the variables Z_x and Z_y are independent for every pair of strings $x \neq y$, so that $\mathbb{E}_{Z_x Z_y}[=] 0$. Thus

$$\mathbb{E}_I\left[\left(\mathsf{bias}\alpha(f(h_I(X))) - \mathsf{bias}_\alpha(f(U_N))\right)^2\right] = \sum_{x,y \in \{0,1\}^n} p_x p_y \mathbb{E}_I\left[Z_x Z_y\right] = \sum_x p_x^2 \mathbb{E}\left[Z_x^2\right]$$

The variance $\mathbb{E}_I\left[Z_x^2\right] = \mathsf{Var}\left[Z_x\right]$ is at most half of the range of Z_x, that is 1. Thus the expected square of the difference of biases is at most $\sum_x p_x^2 = \mathsf{Col}(X)$. □

A.2 XOR of Product Distributions

The second variant on the left-over hash lemma has to do with product distributions. Suppose we have two independent random variables A, B, with at least n bits of Renyi entropy (or min-entropy) between them, where $B \in \{0,1\}^n$. Then we can build a strong extractor which maps the product distribution A, B to something nearly uniform, but which allows one to recover B given A and the output of the extractor. Thus, we can think of A as a "key", which allows one to recover the "message" B.

Definition A.1 (XOR-universal function families (also in Definition 2.3). *A collection of functions $\{h_i\}_{i \in I}$, where $h_i : \{0,1\}^n \to \{0,1\}^n$, is* XOR-universal *if:* $\forall a, x, y \in \{0,1\}^n, x \neq y : \quad \Pr_{i \leftarrow I}[h_i(x) \oplus h_i(y) = a] \leq \frac{1}{2^n - 1}$.

Lemma A.4. *If A, B are independent random variables in $\{0,1\}^n$ such that $\mathbf{H}_2(A) + \mathbf{H}_2(B) \geq n + 2\log\left(\frac{1}{\epsilon}\right) + 1$, and $\{h_i\}$ is a XOR-universal family (from $\{0,1\}^n$ to $\{0,1\}^n$), then*

$$\mathbf{SD}\left(\langle I,\ h_I(A) \oplus B\rangle\ ,\ \langle\ I,\ U_n\rangle\right) \leq \epsilon,$$

where U_n and I are independent and uniform on $\{0,1\}^n$ and $\mathcal{I}$.

The Lemma above gives a special "extractor by XOR" which works for product distributions $A \times B$ with at least n bits on min-entropy between them.

Proof of Lemma A.4. Consider the collision probability of $(i,\ h_i(A) \oplus B)$. A collision only occurs if the same function h_i is chosen both times. Conditioned on that, one obtains a collision only if $h_i(A) \oplus h_i(A') = B \oplus B'$, for A', B' i.i.d. copies of A, B. We can use the XOR-universality to bound this last term:

$$\begin{aligned}\Pr[(i,\ h_i(A) \oplus B) = (i,\ h_i(A') \oplus B')] = \\ \Pr[i = i']\Big(\Pr[B = B'] \cdot \Pr[h_i(A) = h_i(A')] \\ + \sum_{a \neq 0} \Pr[B \oplus B' = a] \cdot \Pr[h_i(A) \oplus h_i(A') = a]\Big) \quad \text{(A.2)}\end{aligned}$$

Now let $t_a = \mathbf{H}_2(A)$, $t_b = \mathbf{H}_2(B)$. For $a \neq 0$, we have $\Pr[h_i(A) \oplus h_i(A') = a] \leq 1/(2^n - 1)$, by the conditions on $\{h_i\}$. On the other hand, by a union bound we have

$$\Pr[h_i(A) = h_i(A')] \leq \Pr[A = A'] + \frac{1}{2^n - 1} \leq 2^{-t_a} + \frac{1}{2^n - 1}$$

Hence, Eqn. A.2 reduces to

$$\begin{aligned}\frac{1}{|\mathcal{I}|}\left(2^{-t_b}\left(2^{-t_a} + \frac{1}{2^n - 1}\right) + \frac{1}{2^n - 1}\left(\sum_{a \neq 0} \Pr[B \oplus B' = a]\right)\right) \\ \leq \frac{1}{|\mathcal{I}|2^n}\left(1 + 2^{n - t_a - t_b} + 2^{-t_b} + \frac{2}{2^n - 1}\right)\end{aligned}$$

Now $2^{n - t_a - t_b} \leq \epsilon^2/2$ by assumption, and we also have $2^{-n} \leq 2^{-t_b} \leq \epsilon^2/2$, since $t_a, t_b \leq n$ and $t_a + t_b \geq n + 2\log\left(\frac{1}{\epsilon}\right)$ (similarly, $n \geq 2\log\left(\frac{1}{\epsilon}\right)$). Hence Eqn. A.2 reduces to $(1 + 2\epsilon^2)/|\mathcal{I}|2^n$. As we mentioned, any distribution on a finite set S with collision probability $(1 + 2\epsilon^2)/|S|$ is at statistical distance at most ϵ from the uniform distribution [46]. Thus, $(i, h_i(A) \oplus B)$ is ϵ-far from uniform. □

A.3 Conditional Min-Entropy

The parameters of extractors are usually given as in Definition 2.1: for a particular bound t on the min-entropy of the input, one is guaranteed a bound ϵ on the distance from uniform of the output. However, if we are only given a guarantee on the conditional min-entropy of the input, then, in general, we get lose an additive factor of $\log\left(\frac{1}{\epsilon}\right)$ in the entropy. Recall that if the conditional min-entropy $\tilde{\mathbf{H}}_\infty(X|Y)$ is t, then with probability $1 - \epsilon/2$ over $y \leftarrow Y$, the min-entropy of X given $Y = y$ is at least $m - \log\left(\frac{1}{\epsilon}\right) - 1$. This gives us the following bound:

Fact A.5. *If* Ext *is a* $(n, t, \ell, \epsilon/2)$ *extractor, and if* $\tilde{\mathbf{H}}_\infty(X|Y) \geq t + \log\left(\frac{1}{\epsilon}\right) + 1$, *then*

$$\mathbf{SD}\left(\langle Y, \mathsf{Ext}(X; S)\rangle\ ,\ \langle Y, U_{\ell+k}\rangle\right) \leq \epsilon.$$

Typically, in the extractors literature $\log\left(\frac{1}{\epsilon}\right)$ is very small, and so losing that much in the parameters is not a problem. However, in our settings $\log\left(\frac{1}{\epsilon}\right)$ will be the security parameter (say 100, roughly), while the entropies we deal with as inputs are only on the order of a few hundred. Ensuring tight tradeoffs between parameters is therefore important.

For a particular function Ext, let $\epsilon(p)$ denote the maximum distance from uniform of $\mathsf{Ext}(X;S)$ when the min-entropy of X is at least $log(1/p)$ (that is, $p = 2^{-\mathbf{H}_\infty(X)}$). Note that $\epsilon()$ will increase with p, and will be 1 when p is large. If the function $\epsilon(\cdot)$ is convex-$\cap$, then we lose nothing by switching to conditional min-entropy:

Lemma A.6. *If* Ext *is an extractor with error function* $\epsilon(p)$ *which is convex-*$\cap$*, then if* $\tilde{\mathbf{H}}_\infty(X|Y) \geq t'$*, we have*

$$\mathbf{SD}\left(\langle Y, \mathsf{Ext}(X;S)\rangle \ , \ \langle Y, U_{\ell+k}\rangle\right) \leq \epsilon(2^{-t'}).$$

In particular, we get "conditional" versions of the left-over hash lemma, as well Lemmas A.2 and A.4, with no loss of parameters (that is, to ensure statistical difference ϵ one only needs $\tilde{\mathbf{H}}_\infty(X|Y) \geq \ell + 2\log\left(\frac{1}{\epsilon}\right)$, as opposed to $\ell + 3\log\left(\frac{1}{\epsilon}\right)$).

Proof. We can write the statistical difference as an average:

$$\mathbf{SD}\left(\langle Y, \mathsf{Ext}(X;S)\rangle, \langle Y, U_{\ell+k}\rangle\right) = \mathbb{E}_{y \leftarrow Y}\left[\mathbf{SD}\left(\mathsf{Ext}(X|_{Y=y};S), U_{\ell+k}\right)\right]$$

Let $p_y = 2^{-\mathbf{H}_\infty(X|Y=y)}$. We can simplify the expression. By the definition of $\epsilon()$:

$$\mathbf{SD}\left(\langle Y, \mathsf{Ext}(X;S)\rangle, \langle Y, U_{\ell+k}\rangle\right) \leq \mathbb{E}_{y \leftarrow Y}\left[\epsilon(p_y)\right]$$

By Jensen's inequality, we have $\mathbb{E}_y\left[\epsilon(p_y)\right] \leq \epsilon(\mathbb{E}_y\left[p_y\right])$ since $\epsilon(\cdot)$ is convex-$\cap$. The expectation of p_y over y is exactly $2^{-t'}$. □