2010

Adaptive learning and cryptography

David Goldenberg
College of William & Mary - Arts & Sciences

ADAPTIVE LEARNING AND CRYPTOGRAPHY

David Goldenberg

Fairfax, Virginia

B.A. College of William and Mary, 2005
M.S. College of William and Mary, 2007

A Dissertation presented to the Graduate Faculty
of the College of William and Mary in Candidacy for the Degree of
Doctor of Philosophy

Dept. of Computer Science

The College of William and Mary
May 2010

APPROVAL PAGE

This Dissertation is submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

David Goldenberg

Approved by the Committee, May, 2010

Committee Chair
Moses Liskov

Mitre Corporation

Associate Professor Qun Li

College of William and Mary, Computer Science Dept

Associate Professor Weizhen Mao

College of William and Mary, Computer Science Dept

Associate Professor Haining Wang

College of William and Mary, Computer Science Dept

Ferguson Professor Chi-Kwong Li

College of William and Mary, Mathematics Dept.

ABSTRACT PAGE

Significant links exist between cryptography and computational learning theory. Cryptographic functions are the usual method of demonstrating significant intractability results in computational learning theory as they can demonstrate that certain problems are hard in a representation independent sense. On the other hand, hard learning problems have been used to create efficient cryptographic protocols such as authentication schemes, pseudo-random permutations and functions, and even public key encryption schemes.

Learning theory / coding theory also impacts cryptography in that it enables cryptographic primitives to deal with the issues of noise or bias in their inputs. Several different constructions of "fuzzy" primitives exist, a fuzzy primitive being a primitive which functions correctly even in the presence of "noisy", or non-uniform inputs. Some examples of these primitives include error-correcting blockciphers, fuzzy identity based cryptosystems, fuzzy extractors and fuzzy sketches. Error correcting blockciphers combine both encryption and error correction in a single function which results in increased efficiency. Fuzzy identity based encryption allows the decryption of any ciphertext that was encrypted under a "close enough" identity. Fuzzy extractors and sketches are methods of reliably (re)-producing a uniformly random secret key given an imperfectly reproducible string from a biased source, through a public string that is called the "sketch".

While hard learning problems have many qualities which make them useful in constructing cryptographic protocols, such as their inherent error tolerance and simple algebraic structure, it is often difficult to utilize them to construct very secure protocols due to assumptions they make on the learning algorithm. Due to these assumptions, the resulting protocols often do not have security against various types of "adaptive" adversaries. To help deal with this issue, we further examine the inter-relationships between cryptography and learning theory by introducing the concept of "adaptive learning". Adaptive learning is a rather weak form of learning in which the learner is not expected to closely approximate the concept function in its entirety, rather it is only expected to answer a query of the learner's choice about the target. Adaptive learning allows for a much weaker learner than in the standard model, while maintaining the the positive properties of many learning problems in the standard model, a fact which we feel makes problems that are hard to adaptively learn more useful than standard model learning problems in the design of cryptographic protocols. We argue that learning parity with noise is hard to do adaptively and use that assumption to construct a related key secure, efficient MAC as well as an efficient authentication scheme. In addition we examine the security properties of fuzzy sketches and extractors and demonstrate how these properties can be combined by using our related key secure MAC. We go on to demonstrate that our extractor can allow a form of related-key "hardening" for protocols in that, by affecting how the key for a primitive is stored it renders that protocol immune to related key attacks.

Dedicated to my family, my friends, and all those who helped me survive and thrive in these past 5 years.

Table of Contents

ACKNOWLEDGMENTS

I acknowledge all the people who helped me complete my thesis. I would first like to thank my advisor Professor Liskov without whom I would have not been able to understand cryptography at all, much less complete my thesis in this time. I would like to thank the College of William and Mary for their academic and financial support. I would like to thank my parents, without whose help and encouragement I would not have been able to finish the PH.D program. Finally, I would like thank the ballroom and swing dancing clubs on campus for preserving my sanity while I worked on finishing the PH.D program and my doctoral thesis.

ADAPTIVE LEARNING AND CRYPTOGRAPHY

Chapter 1

Introduction

1.1 Introduction

Cryptography and computational learning theory may seem to be opposite disciplines, but many linkages exist between them. Two of these links are the existence/ utility of hard problems, and the use of learning theoretic ideas to help cryptographic protocols tolerate noisy, biased data.

Hard Cryptographic Problems The standard method of showing that a cryptographic protocol is secure in a computational sense is building a reduction between a poly-time machine that breaks the security properties of the protocol, to one that solves some mathematical problem. As long as the problem is thought to be hard (require exponential computation) to solve, the cryptographic protocol is then considered to be secure. Many mathematical problems such as DDH, CDH, factoring, discrete logs and others have had their hardness extensively analyzed and have also found large use in cryptographic protocols.

Problems that have found use in cryptography have several properties in common. First, cryptographic problems are not known to be NP-Complete or even NP-Hard. The hardness of a given cryptographic problem is assumed due to various arguments, or simply because a polynomial time algorithm solving the problem has yet to be discovered after years of effort. As such, it is important to find multiple different hard problems for use in cryptography

as it may be the case that a problem that is thought to be hard now, will be shown to be solvable later on. As a practical example of this consider earlier constructions of cryptographic protocols based off of the hardness of various knapsack problems. In addition, most commonly used "cryptographic" problems such as discrete log, factoring and others will be rendered insecure given the existence of quantum computing.

Second, cryptographic problems are assumed hard under average case assumptions, as opposed to being hard only in the worst case, and these problems are thought to be hard under strong adaptive adversaries as well. A hard problem that is useful in cryptographic protocols must be thought to be hard on average, otherwise protocols based off of the problem may be insecure most of the time even while being "secure" for special cases. With regards to the adaptivity of the algorithm solving the problem, the abilities of the adversary which attempts to solve these hard problems reflects the strength of a real life adversary which tries to break the resulting protocol. As cryptographic protocols attempt to be secure for all imaginable attacks, this usually requires that the underlying cryptographic problem be secure against the strongest possible adversaries.

Hard Problems in Computational Learning Theory Now compare a hard "cryptographic" problem to a hard problem in computational learning theory. A "standard" construction of a learning problem is as follows: Begin with a class of *concept* functions $\mathcal{C}$ and a set of *hypothesis* functions $\mathcal{H}$ where all functions in each class are polynomial time. Let an algorithm receive "samples" of a concept function $x, c(x)+e$ where x is sampled from some input distribution on the domain of c and e is sampled from some noise distribution. Can the algorithm output a function $h \leftarrow \mathcal{H}$ that is a "close enough" approximation to c?

A learning problem is considered hard if no algorithm is likely to produce an approximation in $\mathcal{H}$ for a selected function in $\mathcal{C}$. A hardness result is *representation independent*, or a concept class $\mathcal{C}$ is hard to learn in a representation independent sense if there is no algorithm which can output a hypothesis function h of a selected function c for *any* class of functions $\mathcal{H}$.

Hard learning theory problems have found use in cryptographic protocols. A learning problem that is hard to solve, even in a representation independent sense is a problem that is hard for any polynomial time algorithm to solve. As such, if a reduction exists between an adversary breaking the security of a protocol and an algorithm solving the learning problem, this functions as a proof of the security of the protocol. Moreover, learning problems often involve sets of functions $\mathcal{C}$ that are computationally simple to sample from / store and are algebraically simple to evaluate. As we show in our thesis, these properties of a learning problem have resulted in very efficient protocols which have error correcting as well as related key security properties. In addition, representation independent learning / coding theory problems are often known to be NP-Hard or NP-Complete in the worst case, and possess strong evidence of being hard even given the existence of quantum computation. Also while significant strides have been made in solving many "traditional" cryptographic problems such as factoring or DDH, research has been less successful in improving our ability to solve most hard learning problems.

Though learning theoretic problems often have many qualities which make them attractive in protocol design, the learning theory / coding theory model makes several assumptions about the nature of the learning algorithm which renders a hard learning problem difficult to use in the design of very secure cryptographic protocols. One such assumption is that, due to the fact that learning problems are often known to be NP-Hard or even NP-Complete they are not necessarily hard on average. A class of functions $\mathcal{C}$ is considered hard to learn even if the learning algorithm can closely approximate most of the functions in $\mathcal{C}$ as long as one function is hard to approximate. This differs from commonly used cryptographic problems, as those problems are thought to be hard on average, not just hard in the worst case. A much greater issue is that a hard learning problem usually tasks a "weak" adversary to perform a "large" task especially when compared to a cryptographic problem. Consider the difference between a hard learning problem, and a standard type of hard "cryptographic" problem, namely that of distinguishing a certain distribution from random. The learning problem is hard if an algorithm cannot closely approximate the dis-

tribution given only random samples from that distribution, while the adversary tasked to solve the cryptographic problem is asked only to distinguish a distribution from random. It is not necessarily the case that an algorithm which distinguishes a distribution from random, can closely approximate that distribution. [1] Moreover, algorithms tasked to solve hard cryptographic problems are often given large amounts of adaptivity when compare to hard learning problems. Consider the difference between a hard learning problem that gives the algorithm samples $c(x) + e$ where x is randomly selected, and an algorithm tasked to break the security of a MAC, which is given the ability to adaptively requests MAC's of selected messages m.

As a practical example of all these issues, consider the problem of learning parity functions under noise. This is a hard learning problem that has had some success being utilized in cryptographic protocols. The problem is computationally easy to compute, instances of the problem are easy to sample, and it is algebraically simple (the whole problem involves nothing but linear functions). In order to do utilize the learning parity with noise problem in building cryptographic protocols however, the problem is argued to be hard under average case assumptions (uniform selection of the parity function, inputs and correctly weighted error vector) as well as being hard even in a representation independent sense, that is it is hard to come up with *any* algorithm that can closely approximate the parity function. Another learning problem that has been used to develop cryptographic protocols, the *polynomial reconstruction* problem has been treated in a similar way. While these problems have been used in cryptographic protocols such as McEliece, HB$^{\#}$, HB++ and others, those protocols are not secure against most types of "adaptive" adversarial attack.

Biased, Noisy Data and Cryptography Most cryptographic protocols assume the existence of uniform, perfectly reproducible randomness. Practically though uniform randomness is often hard to produce and can be especially hard to reproduce accurately. Due to this issue a great deal of research has been done in allowing cryptographic protocols

[1] Consider the insecure pseudorandom permutation $\mathsf{E}_K(m) = 1||\mathsf{E}'_K(m)$ where E'_K is a secure blockcipher. $\mathsf{E}_K(m)$ is not pseudorandom, but because E'_K is secure, E_K will be hard to predict.

to function correctly in the presence of quasi-random and/or noisy data. Many different cryptographic protocols exist which allow for error tolerance in the various inputs / outputs. Error Correcting Blockciphers combine decryption and error correction into a single function, rather than requiring two separate functions to fulfill these needs. Identity Based Cryptosystems are cryptosystems in which any string can be considered a valid public key thus allowing everyone's "identity" (where this identity can be considered some piece of publicly known information) to serve as a public key. A *Fuzzy Identity Based Cryptosystem* is a cryptosystem where a message encrypted using public key ω can be decrypted using the private key associated with ω' as long as ω and ω' are "close enough" identities.

While a fuzzy IBE scheme deals with errors in a public key, a fuzzy sketch / extractor deals with errors in the private key. A fuzzy sketch is a method of storing and reliably reproducing a sample from a noisy, imperfectly random distribution by publishing a public string known as a *sketch.* This sketch can be said to store the value by functioning as a way of correcting errors in future inputs. The stored value can be recovered given the sketch and any value considered "close" to the original. A fuzzy extractor is a similar protocol that produces a random string based off of a imperfectly random sample, and can reconstruct this string based off of a published sketch and any sample that is close to the original. After the original work of [13], additional security properties of fuzzy extractors and sketches have been proposed:

Reusability: A sketch is considered reusable if seeing multiple different sketches of the same value reveals little additional information about that value.

Robustness: A sketch is considered robust if an adversary cannot produce a different valid sketch of a secret w after seeing *one* sketch of w.

Insider Secure: A fuzzy extractor is considered insider secure if an adversary capable of viewing multiple sketches of an adaptively perturbed secret, manipulating sketches the challenger receives while observing the resulting extracted key, cannot learn any information about the key extracted from an unmodified sketch unknown sketch.

Our Work We propose the idea of *adaptive* learning and examine its impact on cryptography. Adaptive learning is a weaker form of learning than the standard learning model in that it imposes a much smaller requirement on the adversary. The adversary which attempts to solve an adaptive learning problem need not be able to closely approximate the target function, it need only answer an adaptively chosen query about the target function. Problems which are hard to adaptively learn are closer to the traditional idea of a hard cryptographic problem due to this increased adaptivity and weaker information requirement. We feel this makes hard adaptive learning problems well suited for use in the design of cryptographic protocols. Problems which are hard to learn adaptively give us all the efficiency and simple structure of a normal learning problem, yet eliminate many of the difficulties inherent in turning a traditional learning problem into a cryptographic protocol secure against adaptive adversaries.

We examine the learning parity with noise problem (LPN) and from it develop the strong bit finding (SBF) and strong hidden codeword finding (SHCF) problems as the adaptive versions of LPN. The strong bit finding problem tasks an adversary to find $\langle \mathbf{c}, \mathbf{x} \rangle$ for an adaptively chosen vector $\mathbf{c}$ and a randomly selected vector $\mathbf{x}$ while the strong hidden codeword finding problem, the pseudo-repetition of SBF, tasks an adversary to find a vector that is "close" to $\mathbf{C}\mathbf{b}^*$ for an adaptively chosen vector $\mathbf{b}^*$ and a random matrix $\mathbf{C}$. From these problems we construct a highly efficient fully secure authentication protocol as well as an efficient, error correcting related key secure MAC. Going on, we examine the security properties of fuzzy sketches and extractors and utilize our related key secure MAC to construct a *strongly robust fuzzy extractor*, a fuzzy extractor which is both reusable, and (computationally) robust given multiple queries. We also show that this strongly robust fuzzy extractor implies a related key secure MAC, as well as many other primitives through a phenomenon that we call "related-key hardening", a technique for creating protocols which are related key secure out of a great many protocols which are not originally related key secure by changing how the keys for the protocols are stored.

1.2 Prior Work

Learning Theory and Cryptography Cryptography and computational learning theory are two separate areas of computer science which have had significant interactions in the past. Various intractability results in computational learning theory utilize cryptographic constructions in their proofs [47, 37, 3]. In addition, several learning problems which are thought to be hard have been used to create cryptographic constructions [1, 37, 39, 30, 31, 33]. The two most prominent learning problems used to create cryptographic protocols are the "learning parity with noise" problem and the "polynomial reconstruction problem". These problems are quite similar in nature. Both concern themselves with learning some target function f, given many samples of the form $f(x_i)$, where each sample is perturbed with some probability. For the LPN problem, the target function is linear, while in the polynomial reconstruction problem the target function is a polynomial of fixed degree. The learning parity with noise problem has been the basis of several authentication protocols, [31, 33, 30] as well as a public key cryptosystem [45]. Similarly, the polynomial reconstruction problem (PR) has been the basis of a public key encryption scheme as well as a commitment scheme and a blockcipher [39, 4]. Attacks on the various constructions and the underlying learning problems have been proposed. [40, 25, 16, 28, 12, 42].

The HB family of protocols, HB, HB^+, and $HB^\#$, all based off of the LPN problem, do not require the use of cryptographic primitives and as such are very efficient, however they have not been shown to be secure against the same class of adversaries (namely, fully adaptive man in the middle adversaries). The original protocol, HB [31], is easily seen to be insecure against an adversary who can act as a reader as well as passively observing instances of the authentication protocol. HB^+ [33] deals with this difficulty, however HB^+ has been shown to be insecure against an adversary who can modify messages sent by a valid reader during an instance of the protocol, and who can see if that instance is accepted by the reader. $HB^\#$ is secure in that model however it has recently been shown insecure against a fully adaptive man in the middle adversary [30, 10].

Fuzzy Sketches Many methods to deal with error tolerance in cryptography have been proposed. The most predominant amongst these are methods dealing with biased or noisy keys whether public [50, 5, 24, 23] or private [8, 7, 32, 26, 48, 21].

Predominantly, work in tolerating errors in public keys has concerned itself with fuzzy identity based encryption schemes. Identity based encryption is an asymmetric encryption scheme where the public key is allowed to be any arbitrary string and as such each individual using such a scheme can have his email address, phone number, license plate number of some other piece of personal public information represent his public key. Fuzzy Identity Based Encryption allows a person with identity ω and corresponding private key K to decrypt a message encrypted under identity ω' as long as ω and ω' are close enough. Various constructions of Fuzzy IBE schemes have been created.

Work in tolerating errors in private keys has predominantly concerned itself with the construction of *secure sketches* and *fuzzy extractors.* These protocols were first proposed in 2004 [22] by Dodis, Reyzin and Smith as a methodology for allowing a secret key to be derived and transmitted over an insecure channel, given only imperfectly reconstructible, biased data. The secure sketch is used as a method for storing a biased value in such a way that it can be recovered by any value close to the original. A fuzzy extractor is built from a secure sketch and an extractor, by using the extractor on the stored value to produce a random, consistent key. Several different constructions of secure sketches and extractors have been given [22, 18, 17], for varying choices of the underlying distance metric. Boyen shows that prior definitions are not adequate to cases in which the fuzzy secret is used multiple times, and defines the notion of a reusable sketch which addresses this problem [13].

When a fuzzy extractor is used for the purposes of authentication, there remains the possibility of an adversary modifying the sketch as it is sent across the communications channel, which could lead to a form of man-in-the-middle attack. To avoid this, Boyen et al. defined the notion of a *robust* sketch: a sketch for which no adversary can produce a valid

sketch after seeing one *single* valid sketch [14]. Boyen et al. made a keyless[2], statistically robust sketch in the random oracle model. Several subsequent improvements have been described in the literature using the same basic "one-time robustness" definition. Dodis et al. constructed a keyless, statistically robust sketch in the plain model [20]. Cramer et al. [19] and Kanukruthi and Reyzin [35] give robust sketches that lead to fuzzy extractors that produce relatively longer outputs for similar parameters; the former in the common random string model, the latter in the plain model.

In addition to tolerating errors in keys, public and private, work has been done on the construction of *error correcting blockciphers.* Normal blockciphers are very sensitive to errors during decryption. If a single bit is flipped this can often result in a complete decryption failure. Thus it is often necessary to encode encrypted data through some error correcting code before tranmission. While these two steps can be handled independently several constructions of primitives have been given that combine these two properties into one function [44, 43].

Related Key Security Related-key attacks are attacks against constructions using a secret key (such as a blockcipher) in which an attacker attempts to exploit known or chosen relationships among keys to circumvent security properties. Several related-key attacks on primitives have been developed [38, 49, 46], including attacks on AES [29, 9, 52, 10]. While the realism of an adversary's ability to directly influence a secret key is questionable, the issue of related-key security has implications beyond such a setting. For instance, weakness in a blockcipher's key scheduling algorithm may result in known likely relationships amongst round keys, which could lead to an attack against the cipher [6]. As another example, blockcipher based hash functions are only proven secure in the ideal cipher model [11]; in this strong model, related-key security is implied [6]. Thus, the use of a real blockcipher for hashing that is not related-key secure is theoretically questionable: in many such constructions, the adversary's ability to choose the message to be hashed implies an ability to

[2]Note that if the sender and recipient of a secure sketch share a key, this would imply an authenticated channel. So, only keyless constructions, or constructions with very short keys, are of any interest.

launch related-key attacks on the underlying cipher. Indeed, a recent paper by Biryukov et al has made substantial progress on attacking AES-256 in Davies-Meyer mode via a strong related-key attack on AES [10]. Finally, there are settings in which related-key security has been put to good use: several papers make use of schemes with one-time related-key security properties in order to make fuzzy extractors robust against adversarial modification [20, 19, 35].

Positive results concerning related-key security are few. Bellare and Kohno [6] develop a theoretical framework for defining related-key security, show that some notions of related-key security are inherently impossible, and prove that an ideal cipher is related-key secure for a general class of relations. Lucks [41] shows how to achieve "partial" related-key security (meaning, that only part of the key can be varied), and also gave two proposed constructions of related-key secure pseudorandomness from novel, very strong number theoretic assumptions.

Chapter 2

Preliminaries

2.1 Notation

In this section, we list some general notations/definitions that will be used throughout the rest of the thesis.

Matrices And Vectors In general, we will denote a vector in boldface, (i.e. $\mathbf{x}$), and we will denote then i'th element of the vector $\mathbf{x}$ as x_i. We denote the set of all binary k-column by n-row matrices as $\mathbb{M}_n^k$. We denote individual matrices using bold capital letters. For a matrix $\mathbf{X}$, we denote the i'th row of $\mathbf{X}$ as $[\mathbf{x}]_i$, the j'th colum of $\mathbf{X}$ as $[\mathbf{x}]^j$, and the (i,j)'th entry of $\mathbf{X}$ as $[x]_i^j$. We denote the all 0's and all 1's vectors as $\mathbf{0}$ and $\mathbf{1}$ respectively. The inner product of $\mathbf{x}$ and $\mathbf{y}$ is denoted as $\langle \mathbf{x}, \mathbf{y} \rangle$. We denote the set of all binary vectors of length n and Hamming weight t as $\mathcal{H}_t^n$.

Sets and Random Variables We denote the power set of a set $\mathcal{X}$ as $\mathcal{P}(\mathcal{X})$. If an element x is uniformly selected from a set $\mathcal{X}$ we denote this as $x \leftarrow \mathcal{X}$. We use a similar notation if x is sampled from a random variable X. If a family of random variables X is parameterized by parameters x_1, x_2, x_3 we denote the family of variables as $\mathcal{X}_{x_1,x_2,x_3}$ and a member of that family as X_{x_1,x_2,x_3}.

Metric Space A metric space $\mathcal{M}$ is a set along with a *distance function* $||x - y||$ which has the following properties:

1. **Symmetric**: $||x - y|| = ||y - x||$.
2. **Non-Negative**: $||x - y|| \geq 0$ with equality iff $x = y$.
3. **Triangle Inequality**: $\forall z$, $||x - y|| \leq ||x - z|| + ||z - y||$

Codes A code $\mathcal{C}$ is a subset of metric space $\mathcal{M}$ along with a tuple of algorithms $(\mathsf{C}, \mathsf{C}^{-1}, \mathsf{D})$. The minimum distance of a code $\mathcal{C}$ is $d = min_{\forall x,y \in \mathcal{C}}||x - y||$. For an *efficient* (n, k, t) code, C is the encoding function which takes elements of a domain of size 2^k to $\mathcal{C}$, C^{-1} is the decoding function which reverses this process, and D has the property that for all $m \in \mathcal{C}$, $m' \in \mathcal{M}$, if $||m - m'|| \leq t$ then $\mathsf{D}(m') = m$.

A $[n, k, t]$ linear code is a code where $\mathcal{C}$ is a k dimensional linear subspace of $\mathcal{M}$. As such, $\mathsf{C}(\mathbf{x}) = \mathbf{Cx}$ where $\mathbf{C}$ is an n by k matrix, and $\mathbf{x}$ is a k-bit vector. In addition to $\mathsf{C}, \mathsf{C}^{-1}$ and D a linear code $\mathcal{C}$ has a parity check matrix $\mathbf{H}$ of rank $n - k$, where $\forall \mathbf{c} \in \mathcal{C}$, $\mathbf{Hc} = \mathbf{0}$. For a vector $\mathbf{m} \in \mathcal{M}$ we refer to $\mathbf{Hm}$ as the syndrome of $\mathbf{m}$, or $syn(\mathbf{m})$.

Algorithms, Adversaries and Oracles When referring to an algorithm Alg run on input x, we denote this as $\mathsf{Alg}(x)$. We denote $\mathsf{F}(w; r)$ as the algorithm F running on w and utilizing randomness r. When the randomness is not important or clear from context we simply write $\mathsf{F}(w)$. We will denote adversaries by A, and the family of adversaries that use q queries and t time as $\mathsf{A}_{q,t}$. If an adversary A uses an oracle $\mathcal{O}$ which takes input x, we denote this as $\mathsf{A}^{\mathcal{O}(x)}$. We denote A^L for random variable L, as the adversary which can freely sample the random variable L.

Experiments If p is a predicate, then the notation $Pr[x \leftarrow S; y \leftarrow T; \ldots : p(x, y, \ldots)]$ denotes the probability that the predicate p will be true after the ordered sampling of elements $x, y, \ldots$ where S and T can be random variables, sets, or algorithms running on specific inputs.

Negligible Functions If $f : \mathbb{N} \to \mathbb{R}$ is a function, we say that f is *negligible* if for all c, there is an n_0 such that for all $n > n_0, f(n) < \frac{1}{n^c}$.

Statistical Difference and Entropy

Definition 2.1.1 (Statistical Difference) *The statistical difference of two random variables W, W' over a common domain $\mathcal{D}$ is defined as $SD(W, W')$ where*

$$SD(W, W') = \frac{1}{2} \sum_{\forall d \in \mathcal{D}} |Pr[W = d] - Pr[W' = d]|$$

.

Definition 2.1.2 (Entropy) *The entropy of a random variable W with pdf p is defined as:*

$$H(W) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

Definition 2.1.3 (Min-Entropy) *The min-entropy of a random variable W is defined as:*

$$H_\infty(W) = -\log_2(max_a Pr[W = a]).$$

Definition 2.1.4 (Average Conditional Min-Entropy) *The average conditional min-entropy W given W' is*

$$H_\infty(W|W') = -\log(\mathbb{E}_{b \leftarrow W'}[2^{-H_\infty(W|W'=b)}])$$

where $\mathbb{E}$ denotes expectation.

Definition 2.1.5 ((d,m)-pair) *Two distributions W, W' over $\mathcal{M}$ are called a (d,m)-pair if they have the property that the distance between any two points $w \in W$, $w' \in W'$ is $\leq d$ and $H_\infty(W) \geq m$ and $H_\infty(W') \geq m$.*

Chapter 3

Hard Learning Problems and Cryptography

3.1 Introduction

Computational learning theory concerns itself with the ability to learn "concepts", where a concept is typically represented as a function from a specific domain to a specific range. A typical computational learning theory problem can be posed as follows:

Typical Learning Theory Problem Given a set of *concept* functions $\mathcal{C}$ and a set of *representation* functions $\mathcal{H}$ and some information about a selected concept $c \in \mathcal{C}$, output a representation $h \in \mathcal{H}$ such that h agrees with c on a large enough number of inputs.

The most common source of information about a given $c \in \mathcal{C}$ are *samples*, where each sample is a tuple $x, c(x) + \epsilon$ where x is sampled from some distribution on the domain of c and ϵ is some random variable representing noise. An algorithm which learns a concept from such information can be said to be learning in the *probably approximately correct* or PAC model. A learner which learns a function c in the PAC model will with high probability over the distribution of samples output a representation in h from a class of hypotheses $\mathcal{H}$ that is a good approximation for the target $c \in \mathcal{C}$. How well h approximates c and the probability of success for the learner are both considered parameters of the specific learning

problem in question. As for the representation class $\mathcal{H}$, it is usually assumed that $\mathcal{H} = \mathcal{C}$. In many cases though, it is often beneficial to allow $\mathcal{C} \subset \mathcal{H}$ as a class $\mathcal{C}$ may be hard to learn given only representations in $\mathcal{C}$ but may be easy to learn if we allow additional possible representations. If we allow $\mathcal{H}$ to be all possible poly-time algorithms, then the learner is said to be *representation-independent.*

Some hard learning problems have found uses in cryptographic protocols. The two most prominent examples of this are the *learning parity with noise* (LPN) and *polynomial reconstruction* (PR) problem. These problems task an algorithm to learn a selected linear (fixed degree polynomial) function after receiving several noisy samples from the function. Both problems are thought to be hard to learn, even in a representation-independent sense. The LPN problem has been used in the HB series of lightweight authenticated protocols as well as a circular secure encryption system, and the PR problem has been used to implement a public key encryption system, as well as a commitment scheme and stateful encryption scheme.

Several difficulties exist in utilizing these, and other learning problems in the creation of cryptographic protocols. One is that these learning problems are not necessarily hard under average case assumptions over the concept class. A learning problem is hard if there is no algorithm that can approximate *every* concept. Thus, a concept class may be considered hard to learn, even if an algorithm exists to approximate a great many concepts in that class. Another difficulty is that a hard learning problem tasks an algorithm to approximate the function on all points, not just one, and by using a specific representation of that function. Yet, a cryptographic protocol may be broken if a function is approximated on one point, or even if it is possible to distinguish one function from another, no matter how this approximation / distinguishing is done.

As a practical example of these issues, consider a MAC family. A MAC using a secret key K can be considered as a family of functions $\mathcal{MAC}_K$, one for each key. Learning the family of MAC functions in a learning theoretic sense would entail being able to output a representation of a function MAC_K, for a random K that approximates MAC_K well over

the entire domain, in a sense implicitly recovering the key K for the MAC. On the other hand, a MAC is broken in a cryptographic sense if one can learn the correct output on any one given input for a randomly selected MAC function, regardless of how one learns this output. The use of the PR and LPN problems in cryptographic protocols have had to overcome these problems, usually by simply assuming average case hardness, and giving a reduction from the ability to approximate the function on one point, to being able to approximate the entire function (in a representation independent sense).

Our Results We extend the ideas previously used to utilize hard learning problems in cryptography by introducing the idea of *adaptive* learning. Adaptive learning extends previous work on utilizing learning theory problems in cryptography by tasking an adversary to approximate the correct output of an *adaptively* chosen *new* input, given random samples of the concept function. It is immediately clear that some sets of functions are hard to adaptively learn. Consider a secure blockcipher E_K as a family of functions indexed by the key. It is clear that this is a family of functions that are hard to adaptively learn on the average by the definition of a secure blockcipher, as for a randomly selected function E_{K_i}, even given many *adaptively* chosen, noise free samples $\mathsf{E}_{K_i}(m)$, *all* other inputs $\mathsf{E}_{K_i}(m^*)$ are pseudorandom. We seek to demonstrate that more natural families of functions are hard to adaptively learn on average. In that direction, we argue that the learning parity with noise problem is a hard adaptive learning problem. We extend that assumption to define the "strong hidden codeword finding problem". We use the SHCF problem as the basis for constructions of a highly efficient, fully secure authentication protocol, and an error-correcting, efficient, related key secure MAC.

3.2 LPN, RLD and BF Problems

In this section we introduce some hard learning / coding theory problems.

Learning parity with noise Given a random matrix $\mathbf{M}$ and a vector $\mathbf{z} : \mathbf{z} = \mathbf{Mx}$, it is easy to recover $\mathbf{x}$ using standard linear algebra. However, if $\mathbf{z} = \mathbf{Mx} \oplus \mathbf{e}$ where each bit of $\mathbf{e}$ is either 0 or 1, recovering $\mathbf{x}$ from $\mathbf{z}$ and $\mathbf{M}$ may be much harder.

Let p be a fixed probability such that $0 < p < 1/2$. Let B_p be the Bernoulli distribution that outputs 1 with probability p and 0 with probability $(1-p)$. Let $\mathsf{L}_{\mathbf{x},p}$ be the oracle that when queried returns $\langle \mathbf{a}_i, \mathbf{x} \rangle \oplus e_i$ where $\mathbf{a}_i$ is randomly selected from $\{0,1\}^k$ and $e_i \leftarrow B_p$.

Definition 3.2.1 (Learning parity with noise problem) *Define* $\mathsf{ADV}_{\mathsf{LPN}}(\mathsf{A}, k, p)$ *to be*

$$Pr[\mathbf{x} \leftarrow \{0,1\}^k; \mathbf{x}' \leftarrow \mathsf{A}^{\mathsf{L}_{\mathbf{x},p}} : \mathbf{x}' = \mathbf{x}]$$

We say that the (probability p) learning parity with noise problem is hard if the maximum advantage $\mathsf{ADV}_{\mathsf{LPN}}(\mathsf{A}, k, p)$ *over all* A *is negligible in* k.

Viewed as a learning problem, the learning parity with noise problem is the problem of learning a parity function, where $\mathcal{C}$ is the set of linear parity functions, and $\mathcal{H} = \mathcal{C}$. This is because of the fact that if A outputs a vector $\mathbf{x}' \neq \mathbf{x}$, for a random vector $\mathbf{b}$, $\langle \mathbf{x}', \mathbf{b} \rangle \neq \langle \mathbf{x}, \mathbf{b} \rangle$ with probability $\frac{1}{2}$. As such, there is only one vector that will be able to closely approximate x over random inputs, namely x itself.

The LPN problem allows the adversary to adaptively ask for more samples of the form $\langle \mathbf{a}_i, \mathbf{x} \rangle \oplus e_i$. If we only allow the adversary to non-adaptively specify the number of samples from $\mathsf{L}_{\mathbf{x},p}$ it receives, the LPN problem becomes the random linear decoding problem for a code whose size is adaptively selected by the adversary. Let $B_{p,n}$ be the distribution that outputs a n-bit vector, each bit being sampled independently from B_p. Let $C_{\mathbf{x},p,n}$ be the distribution which, when sampled, outputs $(\mathbf{C}, \mathbf{z})$ where $\mathbf{C} \leftarrow \mathbb{M}_n^k$ and $\mathbf{z} = \mathbf{Cx} \oplus \mathbf{e}$, where $\mathbf{e} \leftarrow B_{p,n}$. One can consider the distribution $C_{\mathbf{x},p,n}$ as the distribution which selects a random $[n, k, t]$ linear error correcting code, and gives a perturbed codeword $\mathbf{Cx} \oplus \mathbf{e}$ where the expected weight of $\mathbf{e}$ is pn.

Definition 3.2.2 (Random linear decoding problem) *Define* $\mathsf{ADV}_{\mathsf{RLD}}(\mathsf{A}, k, p)$ *to be*

$$Pr[\mathbf{x} \leftarrow \{0,1\}^k; q \leftarrow \mathsf{A}(1^k); (\mathbf{C}, \mathbf{z}) \leftarrow C_{\mathbf{x},p,q}; \mathbf{x}' \leftarrow \mathsf{A}(\mathbf{C}, \mathbf{z}) : \mathbf{x}' = \mathbf{x}]$$

We say that the (probability p) random linear decoding problem is hard if the maximum advantage $\mathsf{ADV}_{\mathsf{RLD}}(\mathsf{A}, k, p)$ *over all probabilistic polynomial-time* A *is negligible in k where* $q \leftarrow \mathsf{A}(1^k)$ *is polynomial in k.*

It is worth noting that this problem is easy to solve when $\mathbf{C}$ and $\mathbf{z}$ are from the real numbers [15]. We will always assume all vectors and matrices are binary in this thesis.

Definition 3.2.3 (Instance of the RLD problem) *We define an* instance *of the RLD problem as one sample from* $(\mathbf{C}, \mathbf{z})$ *from* $C_{\mathbf{x},p,q}$.

Note this is an extension of the traditional problem of decoding random linear codes, in which q is not adversarially chosen. Its is known that the LPN/RLD problems are NP-Complete. It is thought that the LPN/ RLD problems are hard on average for all p non-negligibly less than $\frac{1}{2}$. For p negligibly close to $\frac{1}{2}$ the problem is trivial, in that almost any vector $\mathbf{x}$ satisfies the given equations.

The RLD and LPN problems ask the adversary to output the unknown concept vector $\mathbf{x}$ itself. The "bit-finding" problem allows the adversary attempting to learn the parity function to be representation independent.

Bit-finding problem The *bit-finding* problem (or BF problem) is a variant of the problem of decoding random linear codes, in which the adversary is not asked to reconstruct $\mathbf{x}$, but is rather asked to find $\langle \mathbf{x}, \mathbf{b} \rangle$ for a *randomly* chosen $\mathbf{b}$.

Definition 3.2.4 (BF problem) *Define* $\mathsf{ADV}_{\mathsf{BF}}(\mathsf{A}, k, p)$ *to be*

$$Pr[\mathbf{x} \leftarrow \{0,1\}^k; q \leftarrow \mathsf{A}(1^k); (\mathbf{C}, \mathbf{z}) \leftarrow C_{\mathbf{x},p,q}; \mathbf{b} \leftarrow \{0,1\}^k; z \leftarrow \mathsf{A}(\mathbf{C}, \mathbf{z}, \mathbf{b}) : z = \langle \mathbf{x}, \mathbf{b} \rangle] - \frac{1}{2}$$

We say that the (probability p) bit-finding problem is hard if the maximum advantage $\mathsf{ADV}_{\mathsf{BF}}(\mathsf{A}, k, p)$ *over all probabilistic polynomial-time (in k) adversaries* A *where q is polynomial in k, is negligible in k.*

If there is an adversary A capable of solving the bit-finding problem, consider the learning algorithm which on input $\mathbf{C}, \mathbf{z}$ outputs a circuit C such that $\mathsf{C}(\mathbf{b}) = \mathsf{A}(\mathbf{C}, \mathbf{z}, \mathbf{b}; r)$ for some random coins r. This is an algorithm which solves the representation independent version of the learning parity with noise problem.

The bit-finding, or HB problem is also known to be equivalent to the learning parity with noise problem. [31] It is important to note that the fact that the adversary specifies how many samples he wishes to see in a non-adaptive fashion is not a limiting factor. If there is a non-adaptive adversary A and an adaptive adversary A' who does not get a sample from $C_{\mathbf{x},p,q}$, but rather is allowed to query $\mathsf{L}_{\mathbf{x},p}$, A on 1^k can output q such that the maximum running time of A' on problems of size k is less than q. This ensures that A can simulate $L_{\mathbf{x},p}$ to A' and as such the advantages are the same.

3.3 SBF Problem

For the bit finding problem an adversary A is tasked to find the output of a randomly selected target function on a randomly selected point, given some noisy information about the target function. We can consider adversaries asked to solve an easier problem, namely given some noisy information about a randomly selected target function find the output of a *selected* input.

This idea leads us to describe the SBF problem as follows:

Definition 3.3.1 (Strong BF problem) *For an adversary* A *and non-negligible* α *define* $\mathsf{ADV}_{\mathsf{SBF}}(\mathsf{A}, \alpha, k, p)$ *to be* $|Pr[\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A}, k, p)] - (1 - p - \alpha(k))|$, *where* $\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A}, k, p)$ *is defined to be*

$$Pr[\mathbf{x} \leftarrow \{0,1\}^k; q \leftarrow \mathsf{A}(1^k); (\mathbf{C}, \mathbf{z}) \leftarrow C_{\mathbf{x},p,q}; (\mathbf{b}^* \neq \mathbf{0}, z^*) \leftarrow \mathsf{A}(\mathbf{C}, \mathbf{z}) : z^* = \langle \mathbf{x}, \mathbf{b}^* \rangle \wedge \forall i \ \mathbf{b}^* \neq [\mathbf{c}]_i]$$

We say that the (probability p) strong bit-finding problem is hard if there exists a non-negligible α such that the maximum advantage $\mathsf{ADV}_{\mathsf{SBF}}(\mathsf{A}, \alpha, k, p)$ *over all probabilistic polynomial-time (in k) adversaries* A *is negligible in k.*

Such an adversary is not solving the underlying learning theory problem in a representation-independent sense as there is no guarantee that A can compute $\langle \mathbf{x}, \mathbf{b} \rangle$ for *random* $\mathbf{b}$. This definition is a little bit unusual as we would hope to bound the success probability of the adversary to be negligibly close to $\frac{1}{2}$. As we will see in Section 3.5 this is impossible, so we bound the success probability to be non-negligibly less than $1 - p$ which is sufficient for our purposes.

3.4 SHCF problem

With regards to the SBF and BF problems, we bound the adversary's success probability to be around $\frac{1}{2}$ due to the fact that an adversary which picks a bit at random will solve both the SBF and BF problems with probability equal to $\frac{1}{2}$. Since we require that the security properties of most cryptographic protocols hold with all but negligible probability, we now deal with the repetition of the BF and SBF protocols, problems where we bound the adversary's success probability to be negligible.

Let $M_{\mathbf{C},p,q,n}$ for a n-row by k-column matrix $\mathbf{C}$ be the distribution where each sample is a random k-row by q-column matrix $\mathbf{A}$ as well as a matrix Z where $\mathbf{Z} = \mathbf{CA} \oplus \mathbf{E}$ and where each column of $\mathbf{E}$ is an independent sample from $B_{p,n}$. We now define the hidden codeword finding problem as the problem of finding a vector that is "close enough" to the codeword of a randomly selected word.

Definition 3.4.1 (Hidden codeword finding problem) *Define* $\mathsf{ADV}_{\mathsf{HCF}}(\mathsf{A}, k, n, p, u)$ *to be*

$$Pr[\mathbf{C} \leftarrow \mathbb{M}_n^k; q \leftarrow \mathsf{A}(1^k); (\mathbf{A}, \mathbf{Z}) \leftarrow M_{\mathbf{C},p,q,n}; \mathbf{b} \leftarrow \{0,1\}^k; \mathbf{z} \leftarrow \mathsf{A}(\mathbf{A}, \mathbf{Z}, \mathbf{b}) : ||\mathbf{Cb} - \mathbf{z}|| \leq u]$$

We say that the hidden codeword finding problem is hard, if the maximum advantage $\mathsf{ADV}_{\mathsf{HCF}}$ *for all probabilistic polynomial time adversaries is negligible in* k.

This problem can be seen as solving the "pseudo-repetition" of the BF problem, it being the "pseudo"-repetition because of the fact that the vectors $[\mathbf{a}]^i$ are repeated for each vector $[\mathbf{c}]_j$ This problem is known to be equivalent to the LPN problem [30]. In a similar fashion to the BF problem, we can consider the pseudo-repetition of the SBF problem which we call the strong hidden codeword finding problem.

Definition 3.4.2 (Strong hidden codeword finding problem) *For an adversary* A *set* $\mathsf{ADV}_{\mathsf{SHCF}}(\mathsf{A}, k, n, p, u)$ *to be*

$$Pr[\mathbf{C} \leftarrow \mathbb{M}_n^k; q \leftarrow \mathsf{A}(1^k); (\mathbf{A}, \mathbf{Z}) \leftarrow M_{\mathbf{C},p,q,n}; (\mathbf{b}^*\mathbf{z}^*) \leftarrow \mathsf{A}(\mathbf{A}, \mathbf{Z}) : \forall i\ \mathbf{b}^* \neq [\mathbf{a}]^i \textit{ and } ||\mathbf{C}\mathbf{b}^* - \mathbf{z}^*|| \leq u]$$

We say that the strong hidden codeword finding problem is hard, if the maximum advantage for all probabilistic polynomial time adversaries is negligible in k.

Variants of the error distribution A variant of these problems is when $M_{\mathbf{C},p,q,n}$ produces $\mathbf{E}$, where the columns of $\mathbf{E}$ do not come from from $B_{p,q}$ but rather are randomly selected from $\mathcal{H}_t^n$ for some t. Due to the Chernoff bound on the binomial distribution and the fact that that these problems are considered hard for all $0 < p \leq \frac{1}{2}$, we may simply pick p to be small enough such that with overwhelming probability the resulting vector is in $\mathcal{H}_t^n$, and equivalently we can consider any error pattern vector from $\mathcal{H}_t^n$ to be a sample from $B_{p,n}$ for sufficiently small p. Denoting the advantage of an adversary solving the HCF and SHCF problems where the columns of $\mathbf{E}$ are selected randomly from $\mathcal{H}_t^n$ as $\mathsf{ADV}_{\mathsf{HCF}_\mathsf{H}}(\mathsf{A}, k, n, t, u)$ and $\mathsf{ADV}_{\mathsf{SHCF}_\mathsf{H}}(\mathsf{A}, k, n, t, u)$ respectively this allows us to show that $\mathsf{ADV}_{\mathsf{HCF}_\mathsf{H}}(\mathsf{A}, k, n, t, u) \leq \mathsf{ADV}_{\mathsf{HCF}}(\mathsf{A}, k, n, p, u)$ and $\mathsf{ADV}_{\mathsf{SHCF}_\mathsf{H}}(\mathsf{A}, k, n, t, u) \leq \mathsf{ADV}_{\mathsf{SHCF}}(\mathsf{A}, k, n, p, u)$ for some p. Due to this, in the protocols we construct we will usually draw our error vectors randomly from the set $\mathcal{H}_t^n$ and not from the distribution $B_{p,n}$. This will allow us to utilize error correcting

codes in our protocol constructions which will eliminate false negatives and positives. Note however that the generator matrix for an error correcting code is usually non-random, while the matrix $\mathbf{C}$ is uniformly random in the SHCF and HCF problem description. This leads to the definition of a "randomized" linear code:

Definition 3.4.3 *A $[n, k, t]$ randomized linear code is generated by a matrix $\mathbf{G} = \mathbf{PCX}$ where $\mathbf{P}$ is a random monomial matrix, $\mathbf{X}$ is a random invertible matrix and $\mathbf{C}$ is a generator matrix for a $[n, k, d]$ linear code that can efficiently correct $t < \frac{d}{2}$ errors.*

We assume that the SHCF problem is hard, even when $\mathbf{C}$ is not uniformly random, but is rather the generator matrix of a randomized linear code, for some good choice of a matrices $\mathbf{P}, \mathbf{C}, \mathbf{X}$. We feel that this is a safe assumption as it is weaker than the assumption made by McEliece and most other cryptosystems based off of coding theory. They assume that $\mathbf{PCX}$ is random even when $\mathbf{G}$ is public, while we only assume that $\mathbf{G}$ is indistinguishable from random when the adversary is only given samples $\mathbf{Gb} \oplus \mathbf{e}$.

3.5 Hardness of the SHCF Problem

In this section we discuss the hardness of the SHCF problem. While the hardness of the HCF problem is known from [30], the assumption that the SHCF problem is hard is a novel assumption.

On an informal level, the reduction proving the hardness of the HCF problem in [30] works for any vector $\mathbf{b}$, not just a randomly selected one. Due to the fact that the adversary cannot select $\mathbf{b}$ however, we cannot say that this suffices to show the hardness of the SHCF problem.

We begin by giving a reduction in one direction from the SHCF to the SBF problem; specifically we show that if SBF is hard to solve then so is SHCF. This does not result in a complete proof for the hardness of SHCF, as the hardness of SBF is also an open question. We spend the rest of the section arguing for the hardness of SBF. Specifically, we begin by showing that SBF is hard to solve extremely well (that is, with probability near 1) by giving

a reduction between a strong adversary solving SBF and the LPN problem. Furthermore, we examine known methodologies for solving the LPN, RLD and bit-finding problems and show that these cannot be leveraged to attack the strong bit finding problem. Finally, we show that if the strong bit finding problem is in fact hard, it is hard on average.

We begin by proving a reduction between the SHCF and SBF problems.

Theorem 3.5.1 *Let* A *be an adversary such that* $\mathsf{ADV}_{\mathsf{SHCF}}(\mathsf{A}, k, n, pn)$ *is non-negligible in* k. *Then there exists an adversary* A' *such that* $\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A}', k, p) \geq 1 - 2p + 2p^2 + \epsilon'$ *for some non-negligible* ϵ'.

Proof. Let A be an adversary and let $\epsilon(k) = \mathsf{ADV}_{\mathsf{SHCF}}(\mathsf{A}, k, n, pn)$ Let A' be an adversary that behaves as follows:

On input 1^n:

1. Run $\mathsf{A}(1^n)$ to obtain q. Return $q' = Nq$.

On input $\mathbf{C}', \mathbf{z}'$:

1. Let $\alpha = 1$.
2. Pick a random invertible matrix $\mathbf{R} \in \mathbb{M}_n^n$, let $\mathbf{C}'_\alpha$ be rows $(\alpha - 1)q$ through $\alpha q - 1$ of $\mathbf{C}'$.

Let $\mathbf{C}'' = \mathbf{C}'_\alpha \mathbf{R}^{-1}$.

3. Pick a random $j \leftarrow \{1, \ldots, n\}$.
4. For each $i \in \{1, \ldots, n\}, i \neq j$, pick a random $\mathbf{b}_i$.
5. Construct $\mathbf{Z}$ where $[\mathbf{z}]_i^l$ for $i \neq j$ is $\langle \mathbf{b}_i, [\mathbf{c}'']^l \rangle \oplus e_i^l$ where e_i^l is 1 with probability p, and where $[\mathbf{z}]_j = \mathbf{z}'_\alpha$, where $\mathbf{z}'_\alpha$ is bits $(\alpha - 1)q$ through $\alpha q - 1$ of $\mathbf{z}'$.
6. Compute $(\mathbf{b}^*, \mathbf{z}^*) \leftarrow \mathsf{A}(\mathbf{Z})$.
7. For $\lfloor \sqrt{(n)} \rfloor$ trials, pick a random $i \neq j$ and count how many times $\langle R^T \mathbf{b}^*, \mathbf{b}_i \rangle \neq \mathbf{z}_i^*$.
8. If the count is greater than or equal to $\sqrt{n}(5p/4 - p^2/2)$, increment α.

If $\alpha = N$, return $\perp$, otherwise, go back to step 2.

9. If the count is less than $\sqrt{n}(5p/4 - p^2/2)$, return $(\mathbf{b}^*, z_j^*)$.

Effectively, A' creates up to N samples from $M_{\mathbf{C}'',p,q,n}$, each time for a new random matrix $\mathbf{C}''$. In each of these samples is an "unknown row" j: for all other rows, A' can

check the answer A gives. We sample the other rows to determine whether the probability of a random row is roughly good or bad. If it is good, we use the answer of A in the unknown row as our answer, otherwise, we try again.

In a given trial, the answer A gives will have some fixed number of errors, all of which except the one in row j can be checked by A$'$. With probability ϵ (a non-negligible probability), this number of errors is at most pn. Note that the simulated $\mathbf{Z}$ has the exact same distribution as A expects. Thus, we can imagine that the row j is chosen only after the adversary gives its answer $(\mathbf{b}^*, \mathbf{z}^*)$.

Let ϵ be the probability that A$'$ gives a good answer (with at most pn errors) for a random $\mathbf{C}$ and random $\mathbf{x}$. There are two subcases: say E_{GR} is the event that there are at most pn errors and the jth row is correct, and E_{GW} is the event that there are at most pn errors and the jth row is incorrect. $Pr[E_{GR}] \geq (1-p)\epsilon$ and $Pr[E_{GW}] \leq p\epsilon$.

In either case, the probability that the count will be greater than $\sqrt{n}(5p/4 - p^2/2)$ is negligible. We view this count as counting the number of correct rows; we are sampling $\sqrt{(n)}$ times and looking to see if we get at most $(1 - 5p/4 + p^2/2)\sqrt{n}$ good rows. The probability that each row we sample is correct is $1 - \frac{pn-1}{n-1}$ in the case of E_{GW} and $1 - \frac{pn}{n-1}$ in the case of E_{GR}.

If X is a random variable determined by counting successes on N biased coins, each of with is a success with probability P, then

$$Pr(X \leq P'N) \leq e^{-\frac{N(P-P')^2}{2P}}.$$

Using this bound, we see that the probability of having a passing count given E_{GW} is at least

$$1 - e^{-\frac{\sqrt{n}(\frac{p}{4} - \frac{p^2}{2} + \frac{1-p}{n-1})^2}{2(1-p+\frac{1-p}{n-1})}},$$

and the probability of having a passing count given E_{GR} is at least

$$1 - e^{-\frac{\sqrt{n}(\frac{p}{4} - \frac{p^2}{2} - \frac{p}{n-1})}{2(1-p-\frac{p}{n-1})}}.$$

In both cases, the probability of not passing the test is negligible (the exponent is $O(\sqrt{n})$ with a negative coefficient for sufficiently large n and all $p < 1/2$). Thus, with $N = \frac{1}{\epsilon^2}$ attempts, our probability of aborting is negligible.

So, we can expect that in some trial, we pass the sampling test and output the jth bit as our answer. But passing the test does not guarantee that the number of errors is $\leq pn$. We can prove, however, that the probability of the test passing when there are more than $3pn/2 - p^2$ errors is negligible. In such a case, we use our bound on the cumulative distribution function of the binomial distribution, with $\sqrt{n}$ samples, probability $3p/2 - p^2$ of a counted failure, and looking for $P' = \frac{5p}{4} - \frac{p^2}{2} - \frac{1}{\sqrt{n}}$[1], we get that the probability is

$$\begin{aligned} Pr[\text{Test passed}| \geq 3pn/2 \text{ errors}] &\leq e^{-\frac{\sqrt{n}(3p/2-p^2-(\frac{5p}{4}-\frac{p^2}{2}-\frac{1}{\sqrt{n}}))^2}{3p-2p^2}} \\ &= e^{-\frac{\sqrt{n}(p/4-p^2/2-1/\sqrt{n})^2}{3p-2p^2}} \end{aligned}$$

This probability is negligible in n for all $p < 1/2$. This accounts for all cases where the probability of a row other than the unknown row being wrong is at least $3p/2$. In particular, this includes all cases where there are more than $3pn/2$ errors altogether. So, we have established:

1. The probability of eventually passing the sampling test is near 1.
2. The probability of passing the sampling test with more than $n(3p/2 - p^2)$ errors is negligible.

Thus, with all but negligible probability, we will output the unknown row in a round in which $(\mathbf{b}^*, \mathbf{z}^*)$ has at most $n(3p/2 - p^2)$ errors. So the probability that the output of A'

[1]Not $P' = \frac{5p}{4} - \frac{p^2}{2}$, since having exactly $\sqrt{n}(5p/4 - p^2/2)$ errors does not pass the test

is correct is $(1-3p/2)(1-\nu) = 1 - 3p/2 + p^2 - \nu$ which is greater than $1 - 7p/4 + 3p^2/2$ for all $p < 1/2$ and sufficiently large n. Thus, A' has a non-negligible advantage, at least $\epsilon' = p/4 - p^2/2$, over $1 - 2p + 2p^2$.

□

Theorem 3.5.1 demonstrates that if the success probability of any adversary who tries to solve the SBF problem can be bounded non-negligibly above $1-2p+p^2$, then the SHCF problem should be considered hard. We now give some arguments towards demonstrating the correctness of that upper bound for the SBF problem. We begin by giving a reduction proving that the SBF problem cannot be extremely easy to solve.

Theorem 3.5.2 *Let* A *be a probabilistic polynomial-time adversary such that* $\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A}, k, p) = 1 - \nu(k)$ *where* ν *is negligible. Then there exists* A' *such that* $\mathsf{ADV}_{\mathsf{RLD}}(\mathsf{A}', k, p)$ *is non-negligible.*

Proof. We define the operation of A'. A' begins by sending 1^k to A and receiving q. A then outputs $2kq$. Upon receiving one sample from $C_{\mathbf{x},p,2kq}$, $(\mathbf{C}_{total}, \mathbf{z}_{total})$, A' takes a random q rows of $\mathbf{C}_{total}$, denotes that as $\mathbf{C}$ and denotes the corresponding rows of $\mathbf{z}_{total}$ as $\mathbf{z}$. A' then randomly selects a $k \times k$ matrix $\mathbf{R}$, a matrix $\mathbf{C}' : \mathbf{C}'\mathbf{R} = \mathbf{C}$ and returns $\mathbf{C}'$, $\mathbf{z}$ to A. This is a valid sample from $C_{\mathbf{Rx},p,q}$ because $\mathbf{Cx} \oplus \mathbf{e} = \mathbf{z} = (\mathbf{C}'\mathbf{R})\mathbf{x} \oplus \mathbf{e} = \mathbf{C}'(\mathbf{Rx}) \oplus \mathbf{e}$. A will then return a pair $\mathbf{b}^*, z^*$ such that $\langle \mathbf{b}^*, \mathbf{Rx} \rangle = z^*$ with all but negligible probability. Since $\langle \mathbf{b}^*, \mathbf{Rx} \rangle = (\mathbf{Rx})^T\mathbf{b}^* = \mathbf{x}^T\mathbf{R}^T\mathbf{b}^* = (\mathbf{b}^*)^T\mathbf{Rx} = \langle \mathbf{x}, \mathbf{R}^T\mathbf{b}^* \rangle$ we now have the inner product of a randomly selected vector, (as $\mathbf{R}$ was randomly selected), and $\mathbf{x}$. A' then repeats this procedure with a new set of rows from $\mathbf{C}_{total}$ and re-runs A a total of $2k$ times. Note that Note that the probability that A returns even one answer that is incorrect is at most $2k\nu(k)$, which is negligible.

The $2k$ vectors returned by A, $\mathbf{b}'_i = \mathbf{R}_i^T\mathbf{b}_i^*$ are k distinct random vectors such that with overwhelming probability $\mathbf{Bx} = \mathbf{z}$ where $[\mathbf{b}]_i = \mathbf{b}'_i$, and where the i'th bit of $\mathbf{z}$ is z_i. With overwhelming probability, some k of the $[\mathbf{b}]_i$ will be linearly independent and A' can solve for $\mathbf{x}$ using Gaussian elimination.

Thus, $\mathsf{ADV}_{\mathsf{RLD}}(\mathsf{A}', k, p)$ is near 1. □

The previous theorem allows us to conclude that for all A, $\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A}, k, p) \leq 1 - \epsilon$ for some non-negligible ϵ. We would like to conclude that $\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A}, k, p) \leq \frac{1}{2} + \nu$ for a negligible $\nu(k)$ however we run into a difficulty based on the follow theorem of Blum et al.:

Theorem 3.5.3 *Let* $(\mathbf{C}, \mathbf{z})$ *be a sample from* $C_{\mathbf{x},p,q}$. *For all sets of indices* i_1 *through* i_s, $\langle [\mathbf{c}]_{i_1} \oplus ... \oplus [\mathbf{c}]_{i_s}, \mathbf{x} \rangle = z_{i_1} \oplus, ..., \oplus z_{i_s}$ *with probability* $\frac{1}{2} + \frac{1}{2}(1-2p)^s$.

Proof. See [12]. □

Consider an adversary which takes two samples $[\mathbf{c}]_i, z_i$, $[\mathbf{c}]_j, z_j$ and computes $[\mathbf{c}]_i \oplus [\mathbf{c}]_j, z_i \oplus z_j$ as its answer. Based on Theorem 3.5.3 this will be right with probability $1 - 2p + 2p^2$ which is non-negligibly greater than 1/2. This attack however, does not mean that the SBF problem should be considered "easy", and this bound is sufficient for the reduction in Theorem 3.5.1 and our later protocols.

We continue by arguing that there is no attack on SBF that does better than the above attack by examining a large class of known attacks on the learning parity with noise, random linear decoding, or bit finding problems. We find that the techniques utilized in these attacks is also the technique utilized in the BKW algorithm of Blum, Kalai and Wasserman, as well as many other algorithms for solving the LPN problem or decoding random linear codes. [25, 16, 28, 12, 42, 27]

These algorithms attempt to find a small number vectors from the sample vectors $\mathbf{c}_i$ such that a linear equation $\mathbf{c}_1 \oplus \mathbf{c}_2 \oplus ... \oplus \mathbf{c}_s = \mathbf{c}^*$ exists, where $\mathbf{c}^*$ is equal to a vector specified by the algorithm. With many independent (or pairwise-independent) equations being equal to the same $\mathbf{c}^*$, this can give us the correct value of a single inner product with high probability. The algorithms then use these vectors $\mathbf{c}_i^*$ in some way to find $\mathbf{x}$. For instance, the attack of [12] on the LPN problem attempts to create the canonical basis vectors using this process. To solve the SBF problem, we do not need to find a set of specific solutions $\mathbf{c}_i^*, \langle \mathbf{c}_i^*, \mathbf{x}, \rangle$ we merely require the label for any one vector $\mathbf{c}_i^*$. We now show that

this methodology is incapable of providing a poly-time attack on the strong bit-finding problem.

Theorem 3.5.4 *Let* A *receive q samples of the form $\mathbf{a}_i, \mathbf{z}_i = \langle \mathbf{a}_i, \mathbf{x} \rangle \oplus e_i$. The probability that $\exists \mathbf{c}^*$ such that two linear equations in the $\mathbf{a}_i$ vectors of size $s = O(polylog(k))$ exist where $\mathbf{a}_{i_1} \oplus \mathbf{a}_{i_2} \oplus, ..., \mathbf{a}_{i_s} = \mathbf{c}^*$ and $\mathbf{a}_{i'_1} \oplus \ldots \oplus \mathbf{a}_{i'_{s'}} = \mathbf{c}^*$ is negligible.*

Proof. There are $\sum_{i=1}^{s} \binom{q}{i}$ different equations of size up to s given q samples. As an upper bound to the probability, we assume that each equation produces a different value $\mathbf{c}_j$. From [51] we can conclude that:

$$\sum_{i=1}^{s} \binom{q}{i} \leq s \binom{q}{s} \leq sq^s \leq s2^{ls}$$

where $l = O(polylog(k))$ as if l is not poly-logarithmic then q is exponential in k which makes A exponential in the security parameter. We note that each value produced by the different equations are pair-wise independent from other values by the uniform selection of $\mathbf{a}_i$. The probability that a pair of equations output the same value $\mathbf{c}^*$ is 2^{-k} as the vectors $\mathbf{a}_i$ are randomly selected and there are $s2^{2ls}$ such equations. Thus the probability that two equations exist that both equal a vector $\mathbf{c}^*$ is less $2^{-k+O(2polylog(k)s)}$. If s is logarithmic in k then this probability is negligible and so we are done. □

What this shows is that if the adversary's plan of attack against the bit-finding problem is to gather "votes" for the value of $\langle \mathbf{x}, \mathbf{c} \rangle$, by finding equations of values that xor to $\mathbf{c}$, and if the adversary has only polynomially many samples, then either $\mathbf{c}$ will have a short equation but very likely only *one*, or $\mathbf{c}$ will be the result of multiple equations, but all such equations will have more than polylogarithmic $\mathbf{b}_i$ values involved. In the former case, Theorem 3.5.3 shows that the adversary, with a single equation, has probability $\frac{1}{2} + \frac{1}{2}(1-2p)^s \leq \frac{1}{2} + \frac{1}{2}(1-2p)^2$ of success. In the latter, each equation gives a negligible advantage over $\frac{1}{2}$, so the adversary would need to examine exponentially many such colliding equations, an act

which is impossible for a polynomial-time adversary. This demonstrates that the known methodologies of gaining information about the unknown parity function are not insufficient to solve the SBF problem.

Attacking SHCF directly We can extend this idea of finding linear equations to the SHCF problem, however given two samples $\mathbf{a}_i, \mathbf{z}_i$ and $\mathbf{a}_j, \mathbf{z}_j$ the probability that $||\mathbf{C}(\mathbf{a}_i \oplus \mathbf{a}_j) - (\mathbf{z}_i \oplus \mathbf{z}_j)|| \leq u$, given that $||\mathbf{C}\mathbf{a}_i - \mathbf{z}_i|| \leq u$ and $||\mathbf{C}\mathbf{a}_i - \mathbf{z}_i|| \leq u$ is negligible for large enough n and t. As such, with regards to the SHCF problem we cannot gain any advantage through this methodology at all, much less amplify the advantage through finding multiple equations.

Random self-reducibility In order to support the claim that the SBF problem is useful for cryptography, we need to justify that it is hard on average. Our arguments so far do not establish this, but we now show that the SBF problem has some self-reducibility properties which lend credence to the idea that the SBF problem is hard on average. These arguments are similar to those in [31].

Lemma 3.5.5 (Random self-reducibility) *An instance* $\mathbf{C}, \mathbf{z}$ *of the SBF problem can be transformed into a different random instance of the SBF problem, such that a correct solution to the resulting problem can translated back to a correct solution of the original.*

Proof. Given an instance $\mathbf{C}, \mathbf{z}$ of the SBF problem we select a random invertible $k \times k$ matrix $\mathbf{R}$ and a matrix $\mathbf{C}'$ such that $\mathbf{C}'\mathbf{R} = \mathbf{C}$. We then select a random vector $\mathbf{x}'$ and cast $\mathbf{C}', \mathbf{z} \oplus \mathbf{C}'\mathbf{x}'$ as the new problem instance. It is clear that this is a sample from $C_{\mathbf{R}\mathbf{x}\oplus\mathbf{x}',p,q}$ as $\mathbf{z} \oplus \mathbf{C}'\mathbf{x}' = (\mathbf{C}'\mathbf{R})\mathbf{x} \oplus \mathbf{e} \oplus \mathbf{C}'\mathbf{x}' = \mathbf{C}'(\mathbf{R}x) \oplus \mathbf{C}'\mathbf{x}' \oplus \mathbf{e} = \mathbf{C}'(\mathbf{R}\mathbf{x} \oplus x') \oplus \mathbf{e}$. Any adversary solving SBF will return a pair $\mathbf{b}^*, z^*$ such that $\langle \mathbf{b}^*, \mathbf{R}\mathbf{x} \oplus \mathbf{x}' \rangle = \langle \mathbf{b}^*, \mathbf{R}\mathbf{x} \rangle \oplus \langle \mathbf{b}^*, \mathbf{x}' \rangle$. As we know $\mathbf{x}'$, and $\langle \mathbf{b}^*, \mathbf{R} \rangle = \langle \mathbf{x}, \mathbf{R}^T\mathbf{b}^* \rangle$ we can compute a correct solution to the original problem instance as long as the solution given to us is correct.

□

It should be noted that this is not a random self reduction. The matrix $\mathbf{R}$ is invertible, which means $\mathbf{C}'$ is not perfectly random given $\mathbf{C}$.

Theorem 3.5.6 (Uniform hardness) *Suppose* A *is a probabilistic polynomial-time adversary such that* $Pr[\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A}, k, p)|\mathbf{C}, \mathbf{z}] > p_0$ *for some* p_0 *and a non-negligible fraction of possible* $\mathbf{C}, \mathbf{z}$ *tuples. Then there exists a* A' *such that for* every *valid* $\mathbf{C}, \mathbf{z}$ $Pr[\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A}, k, p)|\mathbf{C}, \mathbf{z}] > p_0$.

Proof. Let A' be an adversary which receives an instance of the SBF problem, $\mathbf{C}, \mathbf{z}$ where $\mathbf{z} = \mathbf{C}\mathbf{x} \oplus \mathbf{e}$ for some $\mathbf{x}$ and $\mathbf{e}$. A' takes n other rows at random and sums them together, producing a new row of the matrix $\mathbf{C}'$. The corresponding entry z_i' is computed by adding together the corresponding n bits of $\mathbf{z}$ together. The noise rate is now set to be $p' = \frac{1}{2} - \frac{1}{2}(1-2p)^{n+1}$. With non-negligible probability, we can assume that $Pr[\mathsf{WIN}_{\mathsf{SBF}}(\mathsf{A})|\mathbf{C}'\mathbf{z}] > p_0$ Since the "random" instance of the problem utilizes the same secret $\mathbf{x}$ vector as the "real" instance, the solution provided by A is a solution for the instance given to A'. □

This does not show that the problem is hard on average, just that if it is hard for some small (but still non-negligible) fraction, it should be hard for almost all instances, an important fact for the use of this problem in cryptographic protocols. If SBF was hard only for some non-negligible fraction of instances and not for almost all instances, it may also be easy for some non-negligible fraction of instances, which could have an impact on its utility in a cryptographic construction.

An open question is whether or not the SBF problem can be reduced to another known problem, particularly, whether or not SBF is equivalent to BF or LPN.

3.5.1 Explicit Parameter Selection

We now give some possible bounds on the security gained from selection of the parameters k, n, u and p of the SHCF problem. By Theorem 3.5.1 we know that solving SHCF must at least as hard as solving SBF once for a similar parameter selection so we first give some possible security bounds on the SHCF problem by relating SHCF to solving the

SBF problem one or more times. Though this a strong assumption we feel it is not an unreasonable one as the only known methodology we know of to attack the SHCF problem is through solving SBF, SHCF can be seen as an almost independent repetition of n versions of the SBF problem, and finally as we've said previously, Theorem 3.5.1 does show that SHCF is at least as hard as SBF.

We begin with a "worst case" scenario in which the space and time complexity necessary to solve SHCF is the same as the complexities necessary to solve SBF for similar parameter choices. In this case to bound the complexity necessary to solve the SHCF problem in the "worst" case we need to bound the query complexity necessary to solve the SBF problem. We only know one method of attacking the SBF problem, namely the methodology discussed in Theorem 3.5.4 which takes many equations of small size in the $\mathbf{c}$ vectors that xor to the same value then takes a majority vote amongst the corresponding bits z_i^*. We can estimate the number of queries necessary to get enough of such equations in the sample vectors with high probability.

We begin by noting that an adversary obtaining only 2 such "good" equations will cannot get a greater advantage than an adversary only finding one equation. Given only two equations that xor to a fixed vector $\mathbf{b}^*$ and their corresponding bits b_1 and b_2, we maximize our success probability if we output b_1 if $b_1 = b_2$, and output a random bit b otherwise. This algorithm's probability of success, when each bit b is correct with probability α, is $2(\alpha)(1-\alpha)\frac{1}{2} + \alpha^2 = \alpha$ which is the probability of success if we just output b_1 and ignore b_2 entirely. As such we need to find at least 3 equations in the $\mathbf{c}$ vectors that xor to the same value if this methodology even has a chance to work in solving the SBF problem. Let q, the number of $\mathbf{c}_i$ vectors be a power of 2 so $q = 2^l$ for some l. For the i'th equation let z_i be the corresponding bit.

In general, we can consider the algorithm which searches for a equations of size s or smaller that xor to the same value, then conducts a majority vote on the corresponding xor'ed bits. According to Theorem 3.5.4 there are approximately 2^{ls} equations of size s or less given 2^l queries. For any given set of a equations where each equation contains at

least one sample not in any other equation, the probability that they all xor to the same value is $2^{-(a-1)k}$. The probability of finding a equations of size s or less that all xor to the same value is approximately $2^{lsa-(a-1)k}$. So for l approximately $\frac{a-1}{sa}k$ we can expect to find the needed equations. Note that as a increase $\frac{a-1}{a}$ grows closer to 1. Due to this fact we can consider an adversary who searches for equations of size $O(\log(k))$ to only require approximately $2^{\frac{k}{\log(k)}}$ equations. While such an adversary reduces the number of queries it needs, the adversary pays for it in in the increased computation time necessary to find all the equations of size $\log(k)$ or less given 2^l queries. There are approximately $2^{l\log(k)}$ different equations of size $\log(k)$ or less given 2^l queries, so while the space complexity of this algorithm is sub-exponential, the time complexity is rather large and in fact is larger than a brute force attack if $l\log(k) \geq k$. Note that there is no point in an adversary searching for equations of size polynomial in k as those equations are indistinguishable from random values in a statistical sense and thus cannot offer any additional information to the adversary and in addition the time complexity of such an algorithm would be greater than a brute force search over values $\mathbf{x}$.

Minimizing the time complexity at the expense of space we can consider the adversary that searches for equations of size 2 or less. For 2^l samples there are 2^{2l} equations of size 2 and approximately 2^{2ls} different sets of s equations. The probability that 3 equations of size 2 all xor to the same value is $\leq 2^{-2k}$ as 3 equations of size 2 can all xor to the same value only when they do not have any vectors in common and as such the probability each equation xors to a given value is independent from the others. As such, for l approximately $\frac{1}{3}k$ we can expect a set of 3 equations of size 2 to exist where each equation xors to the same value. The adversary must perform around 2^{2l} computations to find these equations, far fewer than $2^{\log(k)}$. Note that more than 3 equations of size 2 will be needed to have the success probability of this adversary be close to $1-p$ so this is only a lower bound on the space complexity of this algorithm.

In our worst case scenario, n has no impact on the security of SHCF. A more reasonable assumption is that solving SHCF problem through solving SBF requires us to solve SBF

multiple times. While we have no algorithm that solves SHCF through solving SBF once and once only, we can give a hypothetical SBF solver that can be used n times to solve SHCF. Consider an adversary A that can solve SBF given q queries for some fixed vector $\mathbf{b}^*$. Such an adversary is much stronger than what is necessary to solve the SBF problem, but has not been completely ruled out by our proofs. One can then build an adversary A' to solve SHCF by splitting up the matrix $\mathbf{A}$ and the vectors $[\mathbf{z}]^i$ into separate, independent instances of the SBF problem. By giving each instance to A A' receives n bits z_i^* where each bit is the inner product $\langle \mathbf{a}_i, \mathbf{c}_j \rangle$ with probability close to $(1-p)$. We can then expect $(1-p)n$ bits to be "correct" and as such $\mathbf{z}^* = (z_1^*, z_2^*, \cdots, z_n^*)$ will be a valid answer for the SHCF problem for $u \leq pn\ pn$. This increases the time complexity of solving SHCF by a multiplicative factor of n. It may increase the space complexity by a multiplicative factor of n as well, depending on whether or not the SBF problem can reuse the same sample vectors $\mathbf{a}_i$ for reach invocation of the SBF solver.

We gain much larger improvements in our security guarantees if the adversary solving SBF returns different vectors $\mathbf{b}^*$ over different problem instances $\mathbf{C}, \mathbf{z}$. If that is the case then we must solve SBF enough times to get solutions using the *same* vector $\mathbf{b}^*$. In an extreme case, A solves SBF and outputs a random vector $\mathbf{b}^*$ over each separate problem instance. This adds a very large order of magnitude to the complexity of solving SHCF as compared to solving SBF. Overall, while our lower bound on query complexity is not that high there are good reasons to believe that the actual security given by practical attacks will be higher.

While informal, this analysis demonstrates that solving SHCF is at least as hard as solving the LPN problem for similar parameters, and the possibility exists that it is much harder.

Chapter 4

Applications of the SHCF Problem

In this section we demonstrate the utility of adaptive learning and of the SHCF problem by giving an efficient RFID authentication protocol as well as an efficient related key secure MAC. The security of both primitives is proven by reductions to the SHCF problem.

4.1 hCAP protocol

In this section we give a construction of an RFID authentication protocol which is fully man-in-the-middle secure based off of the SHCF and LPN problems, efficient, and which has no false acceptance rate. Our security model is very similar to [34] and is stronger than the security model of the $HB^{\#}$ protocol[30].

We define an RFID authentication protocol is a triple of algorithms Tag, Reader, Output. A tag's private information consists of a key K from a keyspace $\mathcal{K}$ and a state S from a state space $\mathcal{S}$; the reader maintains a list of triples of tag identifier, key and current state. An RFID authentication protocol runs as follows. We assume there is a fixed number of rounds n, and construct a transcript $T_\sigma = a_1, b_1, \ldots, a_n, b_n$, where σ is a unique session ID, a_i denotes the ith message from the reader and b_i denotes the ith message from the tag. We insist that the reader's message is the first message since RFID chips do not typically contain their own power sources and thus cannot send a message without receiving one from the reader first. Tag on input a_i outputs b_i, and Reader initially outputs a_1, and on input b_i

outputs a_{i+1}. Output is run by the reader once the protocol is complete, and outputs either $\perp$, indicating rejection, or a tag identifier from its list. Note that Reader, Tag and Output have access to the current partial transcript of a_i, b_i values at all times.

Definition 4.1.1 (Accurate) *We say an RFID protocol is* accurate *if when the reader interacts with a tag $\mathcal{T}_i$ with state S and key K and where the reader has (K, S, i) in its list, the probability that* Output *outputs i is 1.*

Definition 4.1.2 (Un-Forgeability) *An adversary* A *is considered to have broken the unforgeability property of an RFID protocol P if $\mathsf{ADV}_{\mathsf{UNFORG}}(\mathsf{A}, q, t)$ is non-negligible in q and where q is non-negligible in the key length, and where $\mathsf{ADV}_{\mathsf{UNFORG}}(\mathsf{A}, q, t)$ is the probability that an adversary takes time t, interacts q times, and succeeds in the following game:*

1. *A tag $\mathcal{T}$ is set up with a key $K \leftarrow \mathcal{K}$ and state $S \leftarrow \mathcal{S}$, and the reader $\mathcal{R}$ is set up with a list of triples $(K, S, 0)$.*
2. *In the first phase, the tag and reader execute the authentication protocol q times, where* A *is allowed to change any message from the reader to the tag, and vice versa, as well as seeing if the resulting protocol transcript T_σ is thought to be valid by the reader ($\mathsf{Output}(T_\sigma) \neq \perp$).*
3. *In the second phase, the reader begins a single new protocol session with the adversary. Let T_σ^* be the resulting transcript between the adversary and the reader. The adversary wins if $\mathsf{Output}(T_\sigma^*) \neq \perp$ and if T_σ^* involves at least one message change: that is, there exists some i for which the reader sends a_i to the adversary but the adversary sends $a_i^* \neq a_i$ to the tag, or the tag sends b_i to the adversary but the adversary sends $b_i^* \neq b_i$ to the reader.*

Definition 4.1.3 (Anonymity) *An adversary* A *is considered to have broken the anonymity of an RFID protocol P if $\mathsf{ADV}_{\mathsf{ANON}}(\mathsf{A}, q, t)$ is non-negligible in q and where q is non-negligible in the key length and where $\mathsf{ADV}_{\mathsf{ANON}}(\mathsf{A}, q, t)$ is the probability that an adversary takes time t, interacts q times, and succeeds in the following game:*

1. *Two tags $\mathcal{T}_1$, $\mathcal{T}_0$ are set up each with a key randomly selected from $\mathcal{K}$ and a state randomly selected from $\mathcal{S}$, and triples for each tag are given to $\mathcal{R}$.*
2. A *is allowed full man-in-the-middle access between the readers and the tags*
3. *A random bit b is flipped.* A *can now interact with $\mathcal{T}_b$, the other tag $\mathcal{T}_{\bar{b}}$ and the reader q'' additional times (where $q = q' + q''$) .*
4. A *outputs a bit b' and succeeds if $b' = b$.*

Definition 4.1.4 (Fully secure) *An RFID protocol is fully secure if it is accurate, anonymous and unforgeable.*

4.2 hCAP construction

We now give our construction of a fully secure RFID protocol. Our construction requires the use of a pairwise independent hash function family $\mathcal{H}$. These functions families are extremely efficient to implement and as such are suitable for RFID tags.

Definition 4.2.1 *A set of functions $\mathcal{H}$ where each function $h_y \in \mathcal{H}$ maps k bits to $m(k)$ bits is considered a pairwise independent hash function family if $\forall m, m' \in \{0,1\}^k$, $\forall \tau, \tau' \in \{0,1\}^{m(k)}$, $Pr_y[h_y(m') = \tau' \wedge h_y(m) = \tau] = \frac{1}{2^{2m(k)}}$.*

For our protocols we do not actually require that the probability in the above experiment is exactly $\frac{1}{2^{2m(k)}}$, we merely require that it is negligible in k.

We can now give our construction of a fully secure RFID protocol. Let $\mathcal{H}$ be a pairwise independent hash function family that takes $2n + k$ bits as input and uses a k bit key. Each tag $\mathcal{T}$ receives as a key two matrices $\mathbf{C}$, $\mathbf{C}'$ which are generator matrices for randomized $[n, k, t]$ linear codes.

- Initially, Reader sends a random k-bit message a to the tag.
- Tag(a): Compute $\beta = \mathbf{C}\mathbf{y} \oplus \mathbf{e}$ and $\beta' = \mathbf{C}'\mathbf{y} \oplus \mathbf{e}'$ where $\mathbf{y} \leftarrow \{0,1\}^k$ and $\mathbf{e}, \mathbf{e}' \leftarrow \mathcal{H}_t^n$. Return $\tau = h_{\mathbf{y}}(\beta, \beta', a)$, β, and β'.

- Output(a, β, β', τ): For each tag in its list, the reader knows D and D$'$. The reader attempts to decode β using D and β' using D$'$. If $\mathbf{y} = \mathsf{D}(\beta) = \mathsf{D}'(\beta')$, and if $h_{\mathbf{y}}(\beta, \beta', a) = \tau$, output the id of this tag. If this does not succeed for any tag, output $\perp$.

Before proving the security of our construction, we state some necessary theorems. The first theorem is that for a random linear code, random biased codewords are indistinguishable from random vectors.

Theorem 4.2.2 *Let* A$'$ *be a distinguisher such that:*

$$|Pr[\mathbf{C} \leftarrow \mathbb{M}_n^k; (\mathbf{A}, \mathbf{Z}) \leftarrow M_{\mathbf{C},p,q,n} : \mathsf{A}'(\mathbf{A}, \mathbf{Z}) = 1] - Pr[\mathbf{A}, \leftarrow \mathbb{M}_k^q; \mathbf{Z} \leftarrow \mathbb{M}_n^q; \mathsf{A}'(\mathbf{A}, \mathbf{Z}) = 1]| \geq \epsilon(k)$$

for some non-negligible ϵ *and where* p, q, n *are all polynomial in* k*. Then there exists* A *such that* A *can solve the LPN problem.*

To do this, we need to use the following theorem from [36]:

Theorem 4.2.3 *Let* A$'$ *be a distinguisher such that:*

$$Pr[\mathbf{x} \leftarrow \{0,1\}^k; (\mathbf{C}, \mathbf{z}) \leftarrow C_{\mathbf{x},p,q}; \mathsf{A}'(\mathbf{C}, \mathbf{z}) = 1] - Pr[\mathbf{C} \leftarrow \mathbb{M}_q^k; \mathbf{z} \leftarrow \{0,1\}^q; \mathsf{A}'(\mathbf{C}, \mathbf{z}) = 1] \geq \epsilon$$

for non-negligible ϵ*. Then there exists an* A *such that* A *can solve the LPN problem.*

We now prove Theorem 4.2.2

Proof. We first define the hybrid h_i as the pair $\mathbf{A}, \mathbf{Z}$ where for $1 \leq l \leq i$, for $1 \leq j \leq n$ $[z]_j^l = \langle [\mathbf{c}]_j, [\mathbf{a}]^l \rangle \oplus e_l^j$ for $e_l^j \leftarrow B_p$ and for all l such that $i < l \leq q$, $[z]_l^j$ is random. By this definition h_i is the hybrid where the first $i-1$ columns of $\mathbf{Z}$ are correctly produced according to the $M_{\mathbf{C},p,q,n}$ distribution and the remaining columns are random bits. As such, we have

that h_0 is equivalent to the case where the matrix $\mathbf{Z}$ is random, while h_n is equivalent to the case where $\mathbf{Z}$ is produced by $M_{\mathbf{C},p,q,n}$. By the hybrid lemma we must have that if A' can distinguish between the two experiments in Theorem 4.2.2 it can distinguish between h_s and h_{s+1} for some s. We can now construct $\mathsf{A}^{\mathsf{A}'}(\mathbf{C}, \mathbf{z})$ to distinguish between the two experiments in Theorem 4.2.3.

We construct $\mathsf{A}'' = \mathsf{A}^{\mathsf{A}'}$, a distinguisher to meet the conditions of Theorem 4.2.3. First, A'' selects a random k by $s-1$ bit matrix $\mathbf{A}$. A'' then constructs a matrix $\mathbf{Z}$ such that for $1 \leq j < n$, $1 \leq l < s$, $[z]_j^l = \langle [\mathbf{c}]_j, [\mathbf{a}]^l \rangle \oplus e_j^l$ where $e_l^j \leftarrow B_p$, for $l = s$, $[z]^l = \mathbf{z}$ and for $l > s$, $[z]_j^l$ is random. If $\mathbf{z}$ is a random vector, then A'' has just created the hybrid h_s, otherwise it has just created h_{s+1}. Give $\mathbf{A}, \mathbf{Z}$ to A' and return its result.

Since A' can distinguish between the two hybrids, A'' is a distinguisher as required in Theorem 4.2.3. Thus, there is an A''' that can solve the LPN problem. □

Theorem 4.2.2 allows us to prove the following theorem which shows that random words remain pseudorandom, even given their error prone encoding under a random linear code.

Theorem 4.2.4 *For all probabilistic polynomial time adversaries* A *we have that:*

$$|Pr[\mathsf{A}(\mathbf{A}, \mathbf{Z}) = 1] - Pr[\mathsf{A}(\mathbf{R}, \mathbf{Z}) = 1]| \leq \nu(k)$$

for some negligible ν *where* $\mathbf{A}, \mathbf{Z} \leftarrow M_{\mathbf{C},p,q,n}$ *for random* n *by* k *bit matrix* $\mathbf{C}$ *and where* $\mathbf{R} \leftarrow \mathbb{M}_k^q$.

Proof. This proof comes from Theorem 4.2.2 as well as the following four hybrids:

- Hybrid 1: $\mathbf{A}, \mathbf{Z}$ where $\mathbf{A}, \mathbf{Z} \leftarrow M_{\mathbf{C},p,q,n}$.
- Hybrid 2: $\mathbf{A}, \mathbf{U}$ where $\mathbf{A}, \mathbf{Z} \leftarrow M_{\mathbf{C},p,q,n}$, $\mathbf{U} \leftarrow \mathbb{M}_n^q$.
- Hybrid 3: $\mathbf{R}, \mathbf{U}$ where $\mathbf{R} \leftarrow \mathbb{M}_k^q$, $\mathbf{U} \leftarrow \mathbb{M}_n^q$.
- Hybrid 4: $\mathbf{R}, \mathbf{Z}$ where $\mathbf{A}, \mathbf{Z} \leftarrow M_{\mathbf{C},p,q,n}$, $\mathbf{R} \leftarrow \mathbb{M}_k^q$.

It is easy to see that Hybrids 2 and 3 are indistinguishable as $\mathbf{U}$ is independent of $\mathbf{A}$ and $\mathbf{R}$. Hybrids 1 and 2 are indistinguishable by Theorem 4.2.2. Finally, if hybrids 3 and 4 are distinguishable, note that this gives us a distinguisher that can distinguish between "valid" codewords and random vectors, without having access to the associated words, (the matrix $\mathbf{A}$). If this is possible, we can construct a distinguisher A' for Theorem 4.2.2 by ignoring $\mathbf{A}$ and replacing it with a random matrix $\mathbf{R}$. □

We can now prove the security of our RFID protocol. We begin with the unforgeability game.

Theorem 4.2.5 *The hCAP protocol is unforgeable.*

Proof. Let a_i be the reader's message to the adversary in the ith execution of the protocol and let a_i^* be the ith message from Ato the tag. Let $\beta_i, \beta_i', \tau_i$ be the tag's response to a_i^* and let $\beta_i^*, \beta_i'^*, \tau_i^*$ be the ith message from A to the reader. We give a reduction taking any adversary A that during any round of the protocol can create a new $\beta_i^*, \beta_i'^*, \tau_i^*$ such that $\mathsf{Output}(a_i, \beta_i^*, \beta_i'^*, \tau_i^*) \neq \perp$, even given $\beta_i, \beta_i', \tau_i$, to an adversary solving the SHCF problem. To begin, note that if $\mathsf{Output}(a_i, \beta_i^*, \beta_i'^*, \tau_i^*) \neq \perp$ then it must be the case that $\mathsf{D}(\beta_i^*) = \mathsf{D}'(\beta_i'^*)$.

A', given $\mathbf{A}, \mathbf{Z}$ from $M_{\mathbf{C},p,q,n}$ creates a generator matrix for another randomized linear code $\mathbf{C}'$. To simulate the first message from the reader during the i'th execution of the protocol, A' returns a random string a_i. To simulate the i'th message from the tag, on input a_i^* A' selects a column $[\mathbf{a}]^l$ from $\mathbf{A}$, sets $\beta_i = [\mathbf{z}]^l$, $\beta_i' = \mathbf{C}'[\mathbf{a}]^l \oplus \mathbf{e}_l$ where $\mathbf{e}_l \leftarrow \mathcal{H}_t^n$ and $\tau_i = h_{[\mathbf{a}]^l}(\beta_i, \beta_i', a_i^*)$. To simulate the output of the reader on the ith execution of the protocol, A' outputs the tag id if $a_i = a_i^*, \beta_i = \beta_i^*, \beta_i' = \beta_i'^*$, and $\tau_i = \tau_i^*$, otherwise A' outputs $\perp$.

A' picks a random index i in $[1, q]$, simulates the unforgeability game for A, and waits for A to produce its answer $\beta_i^*, \beta_i'^*, \tau_i^*$ in the ith execution of the protocol (whether in the first or second phase of the game). A' decodes to learn $\mathbf{y}_i^* = \mathsf{D}'(\beta_i'^*)$ and returns $(\mathbf{y}_i^*, \beta_i^*)$.

Note that the simulation of the unforgeability game is perfect up until the point where the adversary first produces $(\beta^*, \beta'^*, \tau^*)$ which is different from (β, β', τ) (or for a different a^*), such that $\mathsf{Output}(a, \beta^*, \beta'^*, \tau^*)$ would be $\neq \perp$. The idea is that we are randomly selecting i and hoping that the first time this occurs, it is in round i. Given that the adversary is successful, we are correct about i with probability $\frac{1}{q}$.

There are four cases:

In the first case, $(a_i, \beta_i, \beta'_i, \tau_i) = (a^*_i, \beta^*_i, \beta'^*_i, \tau^*_i)$. In this case, the answer A' gives is incorrect because $\mathbf{y}^*_i$ is not a new word.

In the second case, $a^*_i, \beta^*_i, \beta'^*_i = a_i, \beta_i, \beta'_i$ but $\tau^*_i \neq \tau_i$. In this case we know that τ^*_i is not a valid tag for a_i, β_i, β'_i, (as the valid tag for a_i, β_i, β'_i is τ_i) and as such $\mathsf{Output}(a_i, \beta^*_i, \beta'^*_i, \tau^*_i) = \perp$ with probability 1.

In the third case $(a^*_i, \beta^*_i, \beta'^*_i) \neq (a_i, \beta_i, \beta'_i)$, but $\mathbf{y}^*_i = \mathbf{y}_j$ for some $j \leq i$. In this case, we know that A only has a negligible chance of selecting the "correct" $\tau^*_i = h_{\mathbf{y}^*_i}(\beta^*_i, \beta'^*_i, a_i)$ due to the pairwise independence of the hash function family and the fact that, since a_i is randomly selected by the reader, $\beta^*_i, \beta'^*_i, a_i$ must be a new input.

Finally, if for all $1 \leq j \leq i$ we have that $\mathsf{D}(\beta^*_i) = \mathsf{D}'(\beta'^*_i) \neq \mathsf{D}(\beta_j)$ then $\mathbf{y}^*_i = \mathsf{D}'(\beta'^*_i)$ is new. That is, A' has never seen a codeword for $\mathbf{y}^*_i$ under $\mathbf{C}$. Thus, the output $(\mathbf{y}^*_i, \beta^*_i)$ is a correct answer for A' whenever $\mathsf{D}(\beta^*_i) = \mathsf{D}'(\beta'^*_i)$.

Thus, if A is successful with non-negligible probability ϵ then with probability ϵ/q A makes its first correct and distinct response in round i. Until that point, A' perfectly simulates the unforgeability game. In this case, we have shown that with all but negligible probability, A' produces a correct output. Thus, A' is correct with non-negligible probability ϵ/q, which contradicts the SHCF assumption. □

Theorem 4.2.6 *The hCAP protocol is anonymous.*

Proof sketch. By Theorem 4.2.2 we know that the β_i, β'_i vectors produced by the tag are pseudorandom, even given τ_i, a, and y_i. As such, each message consists of two pseudorandom

codewords β, β' and a correct τ: an adversary distinguishing the tags in an anonymity attack would directly defeat this pseudorandomness. □

Theorem 4.2.7 *The hCAP protocol is accurate.*

Proof sketch. The accuracy of the hCAP protocol comes directly from its unforgeability. If the hCAP protocol is not accurate, then a reader interacting with a tag $\mathcal{T}_i$ will output something other than $\mathcal{T}_i$. It will not output $\perp$ as $\mathcal{T}_i$ is a valid tag, so all steps of Output will pass when the reader tests tag $\mathcal{T}_i$. If with non-negligible probability the reader outputs another tag identifier $j \neq i$ then with non-negligible probability the reader cannot tell the difference between two tags that have randomly generated keys. As such, to impersonate a tag $\mathcal{T}_i$ an adversary need only create its own tag with its own random key. □

4.3 hCAM Protocol and construction

We now construct hCAM, a very efficient related-key secure MAC whose security is based off of the SHCF problem.

The construction of hCAM is based off the hCAP protocol described earlier. The main observation is that in the earlier RFID protocol, the fact that the a_i selected by the reader during the second phase of the unforgeability test was random is never utilized in the unforgeability proof. We only required that a_i is different than all previous vectors selected by the reader. As such, we consider the notion of using a_i as a the message, and the tag's response to the reader as the tag of a_i in our MAC construction.

Definition 4.3.1 (Related-key secure MAC) *A MAC which is related-key secure under Δ is a trio of functions* KeyGen, MAC, Verify *that possess the following properties:*

KeyGen(1^k) *returns a key K.*

MAC(m, K) *returns a tag τ.*

$\mathsf{Verify}(m, \tau, K)$ *returns a bit, such that* $\mathsf{Verify}(m, \mathsf{MAC}(m, K), K) = 1$.

Related-key unforgeability: $\forall \mathsf{A} \in PPT, \exists \nu$ *negligible:* $Pr[K \leftarrow \mathsf{KeyGen}(1^k); m, \tau \leftarrow \mathsf{A}^{\mathsf{F}_{\mathsf{MAC}(\cdot,K),K}} : \mathsf{Verify}(m, \tau, K) = 1 \wedge \mathsf{A}$ *never queried* F *on* (m, δ) *for any* $\delta] \leq \nu(k)$ *where* $\mathsf{F}_{\mathsf{MAC}(\cdot,K),K}$ *returns* $\tau = \mathsf{MAC}(m, \delta(K))$ *on input* (m, δ) *for a perturbation function* $\delta \in \Delta$.

We now restate our previous construction as a related-key strongly secure MAC. Let $\mathcal{H}$ be a pairwise independent hash function family which takes $2n + k$ bits as input and uses a key of length k.

MAC Construction

$\mathsf{KeyGen}(1^k) = \mathbf{C}, \mathbf{C}'$, where $\mathbf{C}, \mathbf{C}'$ are generator matrices to randomized $[n, k, t]$ linear codes.

$\mathsf{MAC}(m, \mathbf{C}, \mathbf{C}')$ treats m as a vector $\mathbf{m}$, selects a random $\mathbf{y} \in \{0,1\}^k$ and computes $\beta = \mathbf{C}\mathbf{y} \oplus \mathbf{e}$, $\beta' = \mathbf{C}'\mathbf{y} \oplus \mathbf{e}'$ where $\mathbf{e}, \mathbf{e}' \leftarrow \mathcal{H}_t^n$. Return β, β' and $\tau = h_{\mathbf{y}}(\beta, \beta', \mathbf{m})$.

$\mathsf{Verify}(m, \beta, \beta', \tau, \mathbf{C}, \mathbf{C}')$ decodes β using D and β' using D'. If either does not decode, or $\mathsf{D}(\beta) \neq \mathsf{D}(\beta')$ reject otherwise take $\mathbf{y} = \mathsf{D}(\beta)$ and see if $h_{\mathbf{y}}(\beta, \beta', \mathbf{m}) = \tau$. If it does accept, if it does not reject.

Theorem 4.3.2 *The above construction is a* $\Delta_\oplus$ *related-key secure MAC where* $\Delta_\oplus = \{\delta_z : \delta_z(x) = x \oplus z\}$.

Proof. We give a reduction between any adversary A who can forge with success probability ϵ given q' queries to MAC and q'' queries to Verify, to an adversary A' that solves the SHCF problem. A' begins by selecting a random index l between 1 and $q'' + 1$.

We first show how A' can simulate the function MAC. Let $\mathbf{A}, \mathbf{Z}$ be the matrices A' receives from $M_{\mathbf{C},p,q,n}$. Let $\mathbf{C}'$ be the matrix for another randomized linear code. When A makes a query to $\mathsf{F}_{\mathsf{MAC}(\cdot,K),K}$ of the form $\mathbf{m}_i, \delta_i, \delta_i'$ (where δ_i, δ_i' are n by k matrices), A' selects a column l of $\mathbf{A}$ and the corresponding column of $\mathbf{Z}$. A' sets $\mathbf{y}_i = [\mathbf{a}]^l$, $\beta_i^* = [\mathbf{z}]^l$ and

computes $\beta_i = \beta_i^* \oplus \delta_i \mathbf{y}_i$, $\beta_i' = \mathbf{C}'\mathbf{y}_i \oplus \delta'\mathbf{y}_i \oplus \mathbf{e}'$, $\tau_i = h_{\mathbf{y}}(\beta, \beta', \mathbf{m})$ and returns $\mathbf{m}, \beta, \beta', \tau$ to A. It is obvious that this is the correct MAC of $\mathbf{m}$ under the key $\mathbf{C} \oplus \delta$, $\mathbf{C}' \oplus \delta'$.

To simulate Verify, A' when receiving a message tag pair $\mathbf{m}_j, (\beta_j, \beta_j', \tau_j)$ from A checks to see if there was a MAC query j made by A such that $(\mathbf{m}_j, \beta_j, \beta_j', \tau_j) = (\mathbf{m}_i, \beta_i, \beta_i', \tau_i)$. If there was, A returns 1, if there is not, A' returns $\perp$.

If $l \leq q''$ this process continues until the l'th verification query. Then A' decodes β_l' using D', obtaining $\mathbf{y}_l$, and then checks to see if $\mathbf{y}_l \neq \mathbf{y}_j$ for all $\mathbf{y}_j$ generated by A'. If $\mathbf{y}_l$ is different, A' outputs $\mathbf{y}_l, \beta_l$ otherwise A fails. If $l = q'' + 1$ then A' waits until A produces a forgery $\mathbf{m}_*, \beta_*, \beta_*', \tau_*$ and checks to see if $\mathsf{D}'(\beta_*')$ is a new key $\mathbf{y}_*$ and outputs $\mathbf{y}_*, \beta_*$ if so. The idea here is that A' is hoping to select the *first* message tag pair given by A that would successfully verify. At least one such pair must exist with probability at least ϵ and A' will select that message/ tag pair with probability at least $\frac{\epsilon}{q''+1}$.

We now show conditioned on A''s selection of the first message tag pair that would successfully verify, its answer is correct with overwhelming probability. Let $\mathbf{m}_l, (\beta_l, \beta_l', \tau_l)$ be the pair selected by A'. If $\mathsf{D}(\beta_l) = \mathsf{D}'(\beta_l') = \mathbf{y}_l = \mathbf{y}_j = \mathsf{D}(\beta_j) = \mathsf{D}(\beta_j')$ for some previous β_j, β_j' then with overwhelming probability $\tau_l \neq h_{\mathbf{y}_l}(\beta_l, \beta_l', \mathbf{m}_l)$ due to the pairwise independence of the hash function family. As such, we must have that $\mathsf{D}'(\beta_l') = \mathbf{y}_l \neq \mathbf{y}_j$ for all previous $\mathbf{y}_j$. Since this is a valid message tag pair, we must also have that $\mathbf{y}_l = \mathsf{D}(\beta_l)$ and thus $\mathbf{y}_l, \beta_l$ is a valid answer to the SHCF problem.

□

It is interesting to note how we gain related key security in our MAC construction. Our MAC is related key secure in that it has a certain type of related key *insecurity*. Namely, we find that for a given message, nonce and tag tuple under one key, it is easy to find valid tags for that message and nonce under any chosen offset of the key.

Also note that this is a very Fiat-Shamir like methodology of taking an authentication protocol and turning it into a message authentication scheme. Note that the Fiat-Shamir heuristic requires a 3-round protocol where the second round message is random. Our protocol is two rounds, but note that the 2nd round of the protocol does require the prover

the ability to create some randomness $\mathbf{y}$ and use that randomness to respond to message a from the reader (verifier).

Chapter 5

Fuzzy Sketches and Fuzzy Extractors

5.1 Introduction

The practical utilization of cryptographic protocols often requires the distribution and storage of secret keys. Keys must contain a high level of entropy, yet must be easily and accurately reproducible. However, generating randomness is expensive and the storage/reproduction of said randomness may be extremely difficult, especially when human beings are involved. High entropy keys are hard to remember, while low entropy keys may render a protocol insecure, regardless of any security guarantees that protocol possess. While the ability for humans to produce and remember high entropy strings may be small, there is a plethora of easily accessible information that while containing a some entropy is not uniformly random, or unreliably reproduceable, or both, a good example being biometric information. Because of this, an important question is how to create cryptographic primitives and protocols that utilize biased or unreliable sources to create and store keys. A *secure sketch* provides a "sketch" of a secret value in such a way that the sketch reveals little information about the secret, yet allows for the secret to be recovered from any "close" secret. A *fuzzy extractor* allows two parties with noisy secrets to agree on a random value

even over an insecure error-prone channel.

Building from the original ideas of Dodis, Reyzin and Smith, several new security properties of sketches have been introduced. A sketch is *reusable* if many sketches of the same secret do not reveal any additional information about that secret, and a sketch is *robust* if an adversary cannot create a new given valid sketch of a secret after seeing a single example.

Our Work A question that is not answered by previous work is whether or not these properties can exist in the same sketch construction. To analyze this we propose the idea of a *strongly robust* sketch. We show that the ideal conception of a reusable robust sketch where all the security properties have statistically strong guarantees is impossible. A reuseable sketch will not be strongly robust against an unbounded adversary. We go on however, to leverage our results in the previous chapter in order to give constructions that allow a computational form of robustness under multiple queries, while preserving the statistical reusability property. In addition, we demonstrate the utility of strongly robust sketches by showing how these sketches can be used to add new security properties to cryptographic protocols by hardening them against related key attacks. This results in showing that not only does a related-key secure MAC imply a strongly-robust fuzzy extractor, but such a fuzzy extractor implies the existence of a related key secure MAC.

5.2 Definitions

In this section we give the various definitions of fuzzy sketches and fuzzy extractors found in previous literature.

A *sketch* is a method of securely storing a value w such that it can be recovered by a user as long as the user knows w' which is "close" to w [22].

Definition 5.2.1 *An (m, m', d) sketch is defined by two algorithms* Gen *and* Rec *which have the following properties:*

For all random variables W where $H_\infty(W) \geq m$ we have $H_\infty(W|\mathsf{Gen}(W)) \geq m'$.

For all w, w' *such that* $||w - w'|| \leq d$, $\mathsf{Rec}(w, \mathsf{Gen}(w')) = w'$.

From the definition alone, a secure sketch may not be sufficient when the adversary knows many sketches $\mathsf{Gen}(w; r)$. However, a *reusable sketch* introduced by Boyen [13], hides the secret w even when given many sketches of w and when the adversary is allowed to choose adaptive perturbations of the input.

Definition 5.2.2 *A* (m, m', d)*-sketch is* Δ-reusable *if the probability of winning in the following game is* $\leq 2^{-m'}$ *for* all *adversaries* A.

Preparation: *The adversary sends to the challenger the specification of a random variable* W.

Randomization: *The challenger samples* w *from* W.

Queries: *The adversary may present to the challenger an arbitrary number of queries of the form* $\delta_i \in \Delta$ *for a perturbation* δ_i. *The challenger runs* $\mathsf{Gen}(\delta_i(w)) = P_i$ *and returns* P_i *to the adversary.*

Test: *The adversary selects a word* w^* *and wins if* $w = w^*$.

When Δ is clear from context or unimportant we will simply call a sketch *reusable.* Unless specifically stated, we will usually assume that Δ is the family of functions where $\delta_i(x) = x \oplus i$.

We also note that, due to the fact that we allow A to be computationally unbounded, this definition implies that $H_\infty(w|\{P_0, P_1, \cdots, P_q\} \geq m'$, where $\{P_0, P_1, \cdots, P_q\}$. is the set of all sketches given to A from the above game.

We gain the following theorem from [13, 22]:

Theorem 5.2.3 (Reuseable Sketch Construction) *Let* C *be an* $[n, k, t]$ *linear code. Let* $\mathsf{Gen}_C(w; r)$ *be defined by* $r \leftarrow \{0,1\}^k, P = \mathsf{Gen}_C = w \oplus \mathsf{C}(r)$. *Let* $\mathsf{Rec}_C(w', P)$ *be defined as* $P \oplus \mathsf{C}(\mathsf{D}(w' \oplus P))$. *Then* Gen_C *and* Rec_C *are a reusable fuzzy sketch.*

We note that from [13] that this sketch reveals the same information about w as the deterministic linear sketch $\mathbf{H}w$, the *syndrome sketch* of Dodis et al. [21]. We note that in general, a deterministic linear sketch can be trivially shown to be reusable under vector addition as one cannot get multiple sketches of the same secret and all sketches of perturbed values are computable given a non-perturbed sketch due to linearity.

Neither the definition of a sketch nor the definition of a reusable sketch preclude the adversary from modifying sketches. A sketch is said to be *robust* if no adversary can produce a (new) valid sketch after observing *one* valid one.

Definition 5.2.4 *An (m, m', d, ν) robust sketch is an (m, m', d) sketch such that* $\mathsf{Rec}(w, P)$ *can output $\perp$ and that the maximum advantage,* $\mathsf{ADV\text{-}ROBUST}(\mathsf{A}, \mathsf{Gen}, \mathsf{Rec})$ *over all adversaries and (d, m)-pairs is $\leq \nu$, where the advantage is defined as the probability that* A *succeeds in the following game:*

Setup: *$w \leftarrow W$, $w' \leftarrow W'$ where W and W' is a (d, m)-pair.*

Challenge: A *receives* $P = \mathsf{Gen}(w)$.

Test: $\mathsf{A}(P)$ *outputs $P^* \neq P$ and wins if* $\mathsf{Rec}(w', P^*) \neq \perp$.

We note that, similar to the definition of reusability, a robust sketch provides security assurances against unbounded adversaries.

A strong extractor enables the "extraction" of randomness from an imperfectly random source.

Definition 5.2.5 *An (n, m', l, ϵ)-strong randomness extractor is a polynomial time randomized algorithm* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\} \rightarrow \{0,1\}^l$ *such that for any random variable W over $\{0,1\}^n$ with min-entropy m' it holds that $SD(\langle \mathsf{Ext}(W; U_s), U_s \rangle, \langle U_l, U_s \rangle) \leq \epsilon$ where U_s is the uniform distribution on s bits.*

Definition 5.2.6 *An (n, m', l, ϵ)-strong extractor is* linear *if* $\mathsf{Ext}(w \oplus x; i) = \mathsf{Ext}(w; i) \oplus \mathsf{Ext}(x; i)$ *for all i.*

A fuzzy extractor combines the ideas of a fuzzy sketch and a strong extractor. A fuzzy extractor allows for the reproduction of a random string R given a public sketch and an imperfectly random, imperfectly reproducible string w. The standard construction of fuzzy extractors utilizes fuzzy sketches [22].

Definition 5.2.7 *A (m, l, d, ϵ) fuzzy extractor is given by two algorithms,* Fsk *and* Rep *with the following properties:*

1. Fsk *is a probabilistic algorithm that on input $w \leftarrow W$ where $H_\infty(W) \geq m$ produces (R, P) such that $SD(\langle R, P\rangle, \langle U_l, P\rangle) \leq \epsilon$.*
2. $\mathsf{Rep}(w', P) = R$ *is the reproduction procedure with the property that, $\forall w, w' : ||w - w'|| \leq d$ and $(R, P) \leftarrow \mathsf{Fsk}(w)$, we have $\mathsf{Rep}(w', P) \rightarrow R$.*

We should note that the only "public" output of Fsk is the value P. We say that $\mathsf{Fsk}(w) = (R, P)$ to denote the idea that R is associated with w. In the future we consider $\mathsf{Fsk}(w)$ to output the public value P only.

The notion of a "reusable" fuzzy extractor follows directly from the idea of a reusable sketch:

Definition 5.2.8 *A $(m, l, d, \Delta, \epsilon)$ reusable fuzzy extractor is a (m, l, d, ϵ) fuzzy extractor such that for all state-preserving adversaries* A *and for all random variables W such that $H_\infty(W) \geq m$, for all i,*

$$SD(\langle R_i, P_0, \ldots, P_q\rangle, \langle U_l, P_0, \ldots, P_q\rangle) \leq \epsilon,$$

where $w \leftarrow W$, $P_0 = Gen(w)$, and for each i, $P_i = Gen(\delta_i(w))$ where $\delta_i \leftarrow \mathsf{A}(P_1, \cdots, P_{i-1})$, and where q is the number of perturbations A chooses to specify before halting.

We may similarly define the notion of a robust fuzzy extractor, however there are some additional considerations which need to be considered. Namely, does the adversary just receive a sketch P, or does an adversary receive a sketch P and the resulting key R?

The first notion is known as "pre-application" robustness, the second "post-application" robustness.

Definition 5.2.9 (Pre-Application Robust Fuzzy Extractor) *A (m, l, d, ϵ)-fuzzy extractor is λ-*pre-application robust *if the maximum advantage* $\mathsf{ADV\text{-}ROBUST(A)}$ *over all* A *is $\leq \lambda$, where* $\mathsf{ADV\text{-}ROBUST(A)}$ *is defined as the probability that the adversary succeeds in the following game:*

Setup: *The challenger samples w from W and w' from W' where W, W' are a (d, m)-pair, and produces $(P, R) = \mathsf{Fsk}(w)$.*

Test: $\mathsf{A}(P)$ *outputs P^* and wins if* $\mathsf{Rep}(w', P^*) \neq \perp$ *and $P^* \neq P$.*

A fuzzy extractor is "post-application" robust if the adversary receives P and R in the **Test** phase of the above game.

It should be noted that in all the definitions so far, reusability and robustness apply to all adversaries A, not just computationally bounded adversaries. One can consider the notions of *computational* reusability and robustness where these properties only hold against polynomial time adversaries as well.

5.3 Combining Reusability And Robustness

The previous literature on fuzzy sketches does not consider robustness under multiple queries, nor does it consider whether or not reusability and robustness can be combined in a meaningful sense. To investigate combining these two properties we define the notion of a strongly robust sketch and a strongly robust extractor.

Definition 5.3.1 (Strong Robustness Advantage) *Define* A*'s success probability in the following game with a sketch* $(\mathsf{Gen}, \mathsf{Rec})$ *as* $\mathsf{ADV\text{-}ROBUST'(A, Gen, Rec}, \Delta)$.

Setup: *Two values are sampled from a (d, m) pair, $w \leftarrow W$, $w' \leftarrow W'$.*

Queries: *For $i = 1...q$,* A *selects a $\delta_i \in \Delta$, and receives* $\mathsf{Gen}(\delta_i(w)) = P_i$.

Test: $\mathsf{A}(P_1, P_2, ..., P_q)$ *outputs* P^*, δ^* *where* $\forall i\ P^* \neq P_i$ *and wins if* $\mathsf{Rec}(\delta^*(w'), P^*) \neq \perp$.

With the previous definitions in mind, we formally define the notion of both a *statistically strongly robust sketch* as well as a *strongly robust sketch.* The difference is that the former is strongly robust against an unbounded adversary, while the latter is only computationally strongly robust.

Definition 5.3.2 *An* (m, m', d, Δ, ν) statistically strongly robust sketch *is an* (m, m', d) *sketch where for all* A *and all* (d, m) *pairs,* $\mathsf{ADV\text{-}ROBUST}'(\mathsf{A}, \mathsf{Gen}, \mathsf{Rec}, \Delta)$ *is* $\leq \nu$.

Definition 5.3.3 *An* (m, m', d, Δ, ν) strongly robust sketch *is an* (m, m', d) *sketch where for all probabilistic, polynomial-time* A *and for all* (d, m) *pairs,* $\mathsf{ADV\text{-}ROBUST}'(\mathsf{A}, \mathsf{Gen}, \mathsf{Rec}, \Delta)$ *is* $\leq \nu$.

We can extend the notion of a strongly robust sketch to that of a strongly robust fuzzy extractor.

Definition 5.3.4 *An* $(m, l, d, \Delta, \epsilon, \nu)$ strongly pre-application robust fuzzy extractor *is an* $(m, l, d, \Delta, \epsilon)$ *reusable fuzzy extractor such that the maximum advantage* $\mathsf{ADV\text{-}ROBUST}'_{fe-pre}(\mathsf{A}, \mathsf{Fsk}, \mathsf{Rep}, \Delta) \leq \nu$. *We define* $\mathsf{ADV\text{-}ROBUST}'_{fe-pre}(\mathsf{A}, \mathsf{Fsk}, \mathsf{Rep}, \Delta)$ *as the maximum probability over all* (d, m)*-pairs of* A *succeeding in the following game:*

Setup: $w \leftarrow W$, *and* $w' \leftarrow W'$ *for a* (d, m) *pair* W, W'.

Queries: *For* $i = 1, \ldots, q$, $\mathsf{A}(P_1, \cdots, P_{i-1})$ *makes a query* $\delta_i \in \Delta$ *and receives* P_i *where* $(P_i, R_i) = \mathsf{Fsk}(\delta_i(w))$.

Test: A *outputs* P^* *and succeeds if* $\mathsf{Rep}(w', P^*) \neq \perp$ *and* $P^* \neq P_i$ *for any* i.

We would like to define a post-application robust fuzzy extractor to be an extractor such that for each query made by the adversary, he receives both the sketch P_i and the resulting key R_i. We note however, that this security definition is impossible to meet. Namely, since each key R_i is an extraction on the secret w, it is easy to show that each such good extraction must reduce the min-entropy of w by at least one bit. As such, given enough keys the adversary can reproduce w and thus break robustness.

Definition 5.3.5 *An* $(m, l, d, \Delta, \epsilon, \nu)$ strongly post-application robust fuzzy extractor *is an* $(m, l, d, \Delta, \epsilon)$ *reusable fuzzy extractor such that the maximum advantage* $\mathsf{ADV\text{-}ROBUST}'_{fe-post}(\mathsf{A}, \mathsf{Fsk}, \mathsf{Rep}, \Delta) \leq \nu$, *where we define* $\mathsf{ADV\text{-}ROBUST}'_{fe-post}(\mathsf{A}, \mathsf{Fsk}, \mathsf{Rep}, \Delta)$ *as the maximum probability over all* (d, m)*-pairs of* A *succeeding in the following game:*

Setup *and* **Test** *are as in Definition 5.3.4.*

Queries: *For* $i = 1, \ldots, q$, A *can make two types of queries,* key *queries and* sketch *queries. For a* sketch *query* A *specifies* δ_i *and receives* P_i *where* $(R_i, P_i) \leftarrow \mathsf{Fsk}(\delta_i(w))$. *When* A *makes a* key *query he can either give a* $\delta_i \in \Delta$ *and receive* (P_i, R_i) *where* $(P_i, R_i) = \mathsf{Fsk}(\delta_i(w))$ *or specify an* j *and receive* R_j *where* $(R_j, P_j) \leftarrow \mathsf{Gen}(\delta_j(w))$ *and where* δ_j *is a previous sketch query made by* A. A *can make as many sketch queries as he wishes, but only one key query.*

5.4 General Impossibility Results

In this section we prove several impossibility results. Specifically, we show that it is impossible to construct a keyless or logarithmic-key, statistically strongly robust sketch under "reasonable" Δ. Our impossibility results easily extend to show that it is impossible to construct keyless or logarithmic-key strongly (pre-application) robust fuzzy extractors for such Δ that are strongly robust against unbounded adversaries.

Specifically we consider a set of perturbations Δ to be *reasonable* if: (1) There is a $\Lambda \subset \Delta$ such that Λ is a group of isometric permutations, that is, permutations δ such that $||w - w'|| = ||\delta(w) - \delta(w')||$, and (2) there is a $\delta^1 \in \Lambda$ such that for all x, $||x - \delta^1(x)|| = 1$. We feel these assumptions represent any reasonable choice for Δ. Boyen notes that if we allow general types of functions in Δ it may impossible to design reusable sketches. As an example, consider the family of functions $\delta_{i,x}$ where for $i = 1, \cdots, n$, $\delta_{i,x}(w) = x$ if the i'th bit of w is 0, a random value otherwise. The code offset sketch mentioned earlier in Theorem 5.2.3 can easily be shown not to be reusable against such a family of functions yet is known to be reusable against vector addition $\mod 2$. Because of this limitation, Boyen

restricts his observations about reusable sketches to sets of perturbations Δ that contain a group of isometric permutations. As for the existence of δ^1, recall that the adversary's ability to specify $\delta \in \Delta$ to apply to w is meant to be a worst-case emulation of variance in w; as such, it would be unreasonable to expect that a low-difference perturbation such as δ^1 cannot be specified.

We first give a review of results originally developed by Boyen [13]. Let $\mathsf{Gen}^*(w)$ be the set $\{P : \exists r : P = \mathsf{Gen}(w; r)\}$. For any subset $\mathcal{E} \subset \mathcal{W}$ let $\mathsf{Gen}^*(\mathcal{E})$ be the union $\bigcup_{w \in \mathcal{E}} \mathsf{Gen}^*(w)$. For a sketch P, let $\mathsf{Gen}^{-1}(P) = \{w : \exists r : \mathsf{Gen}(w; r) = P\}$. Similarly, if $\mathcal{S}$ is a set of sketches, let $\mathsf{Gen}^{-1}(\mathcal{S}) = \{w : \exists r\ \mathsf{Gen}(w; r) \in \mathcal{S}\}$.

Lemma 5.4.1 (Boyen [13]) *Let* Gen *and* Rec *be an* (m, m', d, Δ) *reusable sketch where* $\Lambda \subset \Delta$ *is a group of isometric permutations. Then:*

1. *The reusable sketch* Gen *and* Rec *divide* $\mathcal{M}$ *into at most* $2^{m-m'}$ *equivalence classes,* $\mathcal{E}_i$ *where* $w, w' \in \mathcal{E}_i$ *iff* $\mathsf{Gen}^*(w) = \mathsf{Gen}^*(w')$.
2. $\forall \delta \in \Lambda$, $\forall i$, $\delta(\mathcal{E}_i) = \mathcal{E}_j$ *for some* j.
3. *These classes* $\mathcal{E}_i$ *are determined by the sketch protocol alone.*
4. *Each class* $\mathcal{E}_i$ *can be considered an error correcting code with minimum distance* d.
5. *For all* i, j, $|\mathcal{E}_i| = |\mathcal{E}_j|$. *(As such, let* $|\mathcal{E}|$ *be the size of any class* $\mathcal{E}$*).*

To prove that it is impossible to construct a strongly robust sketch secure against computationally unbounded adversaries, we show that any reusable sketch is not statistically strongly robust. The conclusion follows as a statistically strongly robust sketches must be reusable, else an adversary can view multiple sketches, recover w and make a valid sketch.

As a high level idea of our approach, we first give the following inefficient attack, which suffices to prove our main impossibility result in the keyless case:

Theorem 5.4.2 *Let* Gen, Rec *be a reusable sketch for* reasonable Δ. *Then* Gen, Rec *is not statistically strongly robust for* $d > 1$.[1]

[1] If $d = 0$, our proof may not apply; the construction may be "robust" simply because there exists no different P^* that could be output. Such a sketch can no longer be legitimately called "fuzzy" however.

Proof. Let W, W' be a $(d-1, m)$-pair, which is always a (d, m)-pair.

The attack is relatively simple. We simply obtain, through queries, the entire class $\mathsf{Gen}^*(w)$. (We make queries until, with high probability, every random tape for Gen will have been used at least once.) $\mathsf{Gen}^*(w)$ uniquely determines $\mathcal{E}_i$, the equivalence class containing w. Find $\delta^1(\mathcal{E}_i)$. By Lemma 5.4.1, there is some $\mathcal{E}_j = \delta^1(\mathcal{E}_i)$. Select $w'' \in \mathcal{E}_j$ and find a $P^* \in \mathsf{Gen}^*(w'')$ such that $P^* \notin \mathsf{Gen}^*(\mathcal{E}_i)$, and let δ^* be the identity. Output δ^*, P^* as the forged sketch.

We know we can find such a P^* because $||w - \delta^1(w)|| = 1 < d$, and the minimum distance between values in $\mathcal{E}_i$ is $\geq d$ (this follows from the fact that $\mathcal{E}_i$ can be thought of as a code, by Lemma 5.4.1). As such $\delta^1(w) \in \mathcal{E}_j$ and $\delta^1(w) \notin \mathcal{E}_i$ There must be one sketch $P^* \in \mathsf{Gen}^*(\mathcal{E}_j)$ that is not in $\mathsf{Gen}^*(\mathcal{E}_i)$, else $\mathcal{E}_i = \mathcal{E}_j$ which would be a contradiction.

Due to the error correcting properties of Rec, $\mathsf{Rec}(w', P^*)$ will output $w''' \in \mathcal{E}_j$. This is due to the fact that $P^* = \mathsf{Gen}(\delta^1(w); r)$ for some value r and $||w', \delta^1(w)|| \leq (d-1)+1 = d$.
□

There are three things the adversary in the above attack is required to do.

1. The adversary must successfully determine the "correct" equivalence class $\mathcal{E}_i$ of a secret w given enough sketches of w.
2. The adversary must successfully sample the equivalence class $\delta^1(\mathcal{E}_i)$ knowing only $\mathcal{E}_i$.
3. The adversary must be able to find a sketch P^* such that $P^* \in \mathsf{Gen}^*(\delta^1(\mathcal{E}_i))$ yet $P^* \notin \mathsf{Gen}^*(\mathcal{E}_i)$.

For an unbounded adversary, these three tasks are easy to accomplish, though they do require exponential memory / computation time. We give a more efficient attack in the Section 5.5 that allows us to determine the correct equivalence class $\mathcal{E}_i$ much quicker for generic sketch protocols.

We also note that for most known sketch constructions it is relatively easy to learn $\mathcal{E}_i$, (in fact for the code offset sketch $\mathcal{E}_i$ is determined by only one sketch), and also that for all i, j, $\mathsf{Gen}^*(\mathcal{E}_i) \cap \mathsf{Gen}^*(\mathcal{E}_j) = \emptyset$. So while this attack *as written* is not very efficient, for most

known sketch constructions it can be used to develop a polynomial time attack against the robustness the strong robustness of the construction.

The above attack pertains to sketches that do not use a secret key. If a sketch protocol uses a secret key μ then Gen may take that key as input and the above attack may not work anymore. An extension of the attack to small length μ is relatively easy however.

Theorem 5.4.3 *Let* Gen *and* Rec *be a sketch which utilizes a secret key* μ*, where* $|\mu| = O(log|w|)$*. Then if (*Gen, Rec*) is reusable for* Δ *that contains a group of isometric permutations including a* δ^1*, it is not robust.*

Proof. The attack proceeds similarly to the attack in the previous theorem, with the addition that the adversary guesses at the secret key μ at the very beginning. If A successfully guesses the key, then A can run the previous attack and thus breaks strong robustness. The probability of A selecting the correct key is non-negligible in $|w|$ because of the size of μ. □

We do not give any results concerning a statistically strongly robust sketch which utilizes a secret key larger than logarithmic in size of the secret. Such a key can be expanded using a pseudorandom generator to any size needed to provide an authenticated channel and many techniques for key agreement over such channels are known.[2]

5.4.1 Impossibility Results on Fuzzy Extractors

Our results also extend to the ideas of strongly robust fuzzy extractors. It is easy to see how this is the case for all "standard" fuzzy extractor constructions, constructions where Fsk outputs a sketch Gen(w) for some sketch that is later used to recover w by Rep. As the validity of Fsk is based on the validity of the sketch Gen, we can utilize all the results of Lemma 5.4.1 and Theorem 5.4.2. However, for general constructions of fuzzy extractors it may not be the case that we can use the results of Boyen. Most notably, we do not immediately know if $\delta \in \Lambda$ maps pre-image sets to pre-image sets, and we do not know if the preimages induced by Fsk form a code.

[2]Assuming the existence of exponentially hard cryptography, a polylogarithmic key suffices against any polynomial adversary.

We can utilize Claims 11.1, 11.2, 11.3, and 11.4 from Boyen to demonstrate that Fsk must split up $\mathcal{M}$ into equivalence classes, and that $\delta \in \Lambda$ must map classes to classes as the proofs of these claims generalize to reusable fuzzy extractors, since the claims only rely on the fact that there must be some min-entropy remaining in w, even after seeing all conceivable sketches of w a fact which must be true for a strongly robust fuzzy extractor as well, else an unbounded adversary could easily break the strong robustness of the fuzzy extractor.

However, if $\mathsf{Rep}(w, \mathsf{Fsk}(w)) = \mathsf{Rep}(w', \mathsf{Fsk}(w'))$ for $w, w' : ||w - w'|| \leq d$, then it may be the case that the equivalence classes induced by Fsk do not form an error correcting code. As long as we allow that there is a δ^k, $k < n$ such that δ^k maps one class to a different class then we can maintain both of the assumptions necessary for Theorem 5.4.2 and as such break strong robustness of that extractor.

We also note that as long as the fuzzy extractor has the property that we need not see *every* sketch of a secret w before we can find the equivalence class $\mathsf{Fsk}^{-1}(\mathsf{Fsk}(w))$, then we need make no assumptions about δ at all as our more efficient attack in Section 5.5 will be able to find a valid new sketch of w as it will not need to see all valid sketches of the w to determine the correct equivalence class.

5.5 Specific Impossibility Results

Our impossibility results in Section 5.4 demonstrate that previous constructions cannot be statistically strongly robust. In this section, we extend these results and show that some previous constructions are not strongly robust even in a computational sense. We give an attack that works on the constructions of [35, 20]. We then go on to enhance our previous attack against generic fuzzy extractors to greatly increase its efficiency.

We first give the construction of a robust fuzzy extractor that is found in [35, 20]. Let $\mathsf{SS}(w)$ be a deterministic, linear sketch. As such, there is a matrix $\mathbf{S}$ such that $\mathsf{SS}(w) = \mathbf{S}w$. Let $\mathbf{S}'$ be a matrix such that $\frac{\mathbf{S}}{\mathbf{S}'}$ has full rank. Let $\mathsf{SS}'(w) = \mathbf{S}'w$. For $c = \mathsf{SS}'(w)$, let a be

the first half of c, b the second half, where both a and b are viewed as elements of $\mathbb{F}_{2^{n'/2}}$. Set $L = 2\frac{k}{n'}$. Let $s = \mathsf{SS}(w)$. Pad s such that $|s| = Ln'/2$ and then split s into L bit strings of size $n'/2$. Define $f_{s,i}(x) = x^{L+3} + x^2(s_{L-1}x^{L-1} + s_{L-2}x^{L-2} + ... + s_0) + ix$. Fsk is now defined as $\mathsf{Fsk}(w) = (s, i, \sigma)$ where i is randomly selected, σ is the last v bits of $f_{s,i}(a) + b$ and where R is the remaining bits of $f_{s,i}(a) + b$.

We give two attacks. The first attacks the post-application robustness of this scheme, the second attacks the pre-application robustness.

- A makes two public queries where $\delta = 0$, receiving s, i, σ, R and s', i', σ', R' (A receives R and R' due to the fact that it is a post-robustness attack).
- A denotes $X = R||\sigma$ and $X' = R'||\sigma'$.
- A computes $X - X' = (i - i')a$ due to the fact that $\mathsf{SS}(w)$ is deterministic and for both queries $\delta = 0$.
- A finds $a = (X - X')(i - i')^{-1} = (i - i')^{-1}(i - i')a = a$.
- A finds $b = X - f_{s,i}(a)$.

Once A finds both a and b and given that SS' is linear and SS is linear, A can easily compute a new sketch for any $\delta \neq 0$ and any i.

This next attack demonstrates that this protocol is not pre-application robust, so as such the adversary does not receive the extracted key R.

- A makes two public queries where $\delta = 0$, receiving s, i, σ and s', i', σ'.
- A computes $\sigma - \sigma' = f_{s,i}(a) + b]_1^v - (f_{s',i'}(a) + b)]_1^v = (i - i')a]_1^v$.
- A makes a new public query where $\delta = 0$, and receives s'', i'', σ''.
- A creates a new sketch where $s^* = s''$, $i^* = (i - i') + i''$ and $\sigma^* = \sigma'' + \sigma - \sigma'$.

Due to the fact that for all these public queries, $\delta = 0$ we have that $s = s' = s'' = s^*$. We have $f_{s,i^*}(a) = f_{s,i''}(a) + (i - i')a = f_{s,i''}(a) + \sigma - \sigma'$ as such $\sigma^* = \sigma'' + \sigma - \sigma'$ and so the s^*, i^*, σ^* is a valid sketch.

More efficient generic attack Next, we describe a general attack that should be more efficient than the one given in our general impossibility result. Let w^* be the secret value.

Attack. We note that we can break all sketches into two cases: 1) $\mathsf{Gen}^{-1}(P) = \mathcal{E}^*$ where $\mathcal{E}^*$ is a single class and 2) $\mathsf{Gen}^{-1}(P) = \mathcal{E}_{i_1} \cup \cdots \cup \mathcal{E}_{i_k}$ for some k equivalence classes where $k \geq 2$. In case 1, we know the correct equivalence class after only one sketch. As such we need only output another sketch from P^*, set δ^* to the identity and we are done. This attack will work due to the fact that all sketches in $\mathcal{E}^*$ are capable of being produced by the secret w^*.

If we are in case 2 however, we have two options. If $|\mathsf{Gen}^*(\mathcal{E}_{i_1}) \cap \cdots \cap \mathsf{Gen}^*(\mathcal{E}_{i_k})| \geq 2$ then there exists another sketch P^* such that $P^* \in \mathsf{Gen}^*(\mathcal{E}_{i_1}) \cap \cdots \cap \mathsf{Gen}^*(\mathcal{E}_{i_k})$, $P^* \neq P$, $P^* \in \mathsf{Gen}^*(\mathcal{E}^*)$ and as such P^* is a valid sketch of w, and we are done.

Therefore, we assume that this is not the case and $|\mathsf{Gen}^*(\mathcal{E}_{i_1}) \cap \cdots \cap \mathsf{Gen}^*(\mathcal{E}_{i_k}) \backslash P| = 0$. A then requests another sketch of w, receiving a new P not equal to any prior P. By Lemma 5.4.1, if it is true that $\mathsf{Gen}^{-1}(P_2) \cap \mathsf{Gen}^{-1}(P) = \mathcal{E}_{i'_1} \cup \cdots \mathcal{E}_{i'_{k'}}$ where $k' < k$, and $\mathcal{E}_{i'_j} \subset \mathsf{Gen}^{-1}(P)$ for all j otherwise we contradict our assumption that $\mathsf{Gen}^*(\mathcal{E}_{i_1}) \cap \cdots \cap \mathsf{Gen}^*(\mathcal{E}_{i_k})$ contains only P.

Thus, for every sketch that we see, we are able to eliminate at least one equivalence class from the set of possible classes. Denote the current set of q sketches seen by A as $\mathcal{P}_q$. A continues by examining the current intersection of the images of the possible equivalence classes, excluding all sketches in $\mathcal{P}_q$. If this set is non-empty, then a new valid sketch of w was found, and we are finished. Otherwise, A requests another sketch, removing at least one equivalence class from consideration. This process continues until we have found either a valid sketch of w, or have determined the correct equivalence class $\mathcal{E}^*$ and as such we can run the attack from Theorem 5.4.2.

For most sketches / fuzzy extractors this should result in a polynomial time attack against the strong robustness of the sketch. This is due to the fact that the classes $\mathcal{E}_i$ are often efficiently samplable, and the fact that for most known protocols $\mathsf{Gen}^{-1}(P)$ equals one and only one equivalence class $\mathcal{E}_i$.

5.6 Strongly Robust Fuzzy Extractor Constructions

In this section we give a construction of a strongly post-application robust fuzzy extractor. We modify the construction in [19] to obtain both resuability and strong-robustness.

We construct our fuzzy extractor in the common random string (CRS) model. While it would be preferable to construct a fuzzy extractor without resorting to a common string, we note that the only properties we require out of our string is that it is common to all parties involved, that it is random, and that it is resistant to modification by the adversary. Similarly to [19] our common string need only be chosen once when the system is designed, can be hard coded into all software implementing the system or can be chosen by the parties involved in using the sketch, and can be observed (though not modified) by the adversary. We do not believe that this significantly increases the amount of trust required, a view shared by Cramer et al.

Our construction will rely on a linear strong extractor, as well as an xor related-key secure MAC. While there has been little successful work on constructing provably related-key secure primitives, we do note that some papers (including Cramer et al. [19]) make use of MACs that are *one-time* related-key secure. In addition we note that a practical construction of a xor related key secure MAC exists under the SHCF problem, which was discussed earlier.

Definition 5.6.1 *A family of functions* $MAC_k^{rel} : \{0,1\}^* \rightarrow \{0,1\}^n$ *is an xor-related key secure MAC if the maximum advantage* $\mathsf{ADV\text{-}MAC_{RK}}(\mathcal{A}, \mathsf{MAC}^{rel}, n)$ *is negligible in* n*, where* $\mathsf{ADV\text{-}MAC_{RK}}(\mathcal{A}, \mathsf{MAC}^{rel}, n)$ *is defined to be*

$$\mathsf{ADV\text{-}MAC_{RK}}(\mathcal{A}, MAC^{rel}, n) = Pr[k \leftarrow \{0,1\}^n; (x, \sigma, \delta) \leftarrow \mathcal{A}^{\mathcal{O}_k^{rel}} : \mathsf{MAC}_{k\oplus\delta}^{rel}(x) = \sigma]$$

Where x *was not a query made to* $\mathcal{O}_K^{rel}$ *and where* $\mathcal{O}_k^{rel}$ *returns* $\mathsf{MAC}_{k\oplus\delta}^{rel}(x)$ *on input* (x, δ).

We now give our construction. Let $\mathcal{M} = \{0,1\}^n$ under the Hamming metric. Let Gen be

a (m, m', d) deterministic linear sketch. As such, $\mathsf{Gen}(w) = \mathbf{H}w$ for some $n-k \times n$ matrix $\mathbf{H}$. Note, the syndrome sketch described earlier is such a sketch. Let l be a parameter such that $l \leq \lceil m' - 2\log\frac{1}{\epsilon}\rceil$. Parse our CRS as a matrix $\mathbf{S}$ such that $\frac{\mathbf{H}}{\mathbf{S}}$ is an $n \times n$ matrix. Let $\mathbf{S}_M$ be the first l_{MAC} rows of $\mathbf{S}$ and let $\mathbf{S}_K$ be the remaining l_{KEY} rows, so $l_{MAC}+l_{KEY} = l$. Let MAC_μ be an xor-related key secure MAC using key μ. We now construct our strongly robust fuzzy extractor.

Definition 5.6.2 ($\mathsf{Fsk}(w)$)

1. *Let* $\mu = \mathbf{S}_M w$.
2. *Let* $Q = \mathsf{Gen}(w) = \mathbf{H}w$.
3. *Let* $R = \mathbf{S}_K w$.
4. *Let* $\tau = \mathsf{MAC}_\mu(Q)$
5. *Output* $P = (Q, \tau), R$

Definition 5.6.3 ($\mathsf{Rep}(w', P)$)

1. *Run* $w'' = \mathsf{Rec}(w', P)$.
2. *Set* $\mu' = \mathbf{S}_M w''$ *and* $R = \mathbf{S}_K w''$.
3. *Set* $\tau' = \mathsf{MAC}_{\mu'}(Q)$.
4. *If* $\tau = \tau'$ *output* R *else output* $\perp$.

We now prove some theorems bounding the entropy loss on w due to an unbounded adversary seeing multiple sketches. We note that although H and S are chosen randomly, $\frac{\mathbf{H}}{\mathbf{S}}$ is of full rank with overwhelming probability. In what follows, we assume that $\frac{\mathbf{H}}{\mathbf{S}}$ is of full rank.

Lemma 5.6.4 *Let* $\mathcal{E}_i$ *be an equivalence class of the sketch* Gen. *Then if* $\frac{\mathbf{H}}{\mathbf{S}}$ *is of full rank,* $\mathbf{S}$ *is a bijection from* $\mathcal{E}_i$ *to* $\{0,1\}^l$.

Proof. We first show that $\mathbf{S}$ is injective. If it is not, then for some $X, X' \in \mathcal{E}_i$ such that $\mathbf{S}X = \mathbf{S}X'$, we know that $\frac{\mathbf{H}}{\mathbf{S}}X = \frac{\mathbf{H}}{\mathbf{S}}X'$ because of the fact that $\mathbf{H}X = \mathbf{H}X'$ by the definition

of $\mathcal{E}_i$. Thus $\frac{\mathbf{H}}{\mathbf{S}}$ is not full rank as it is not injective. Surjectivity comes from the fact that $\mathbf{S}$ is injective and that $\frac{\mathbf{H}}{\mathbf{S}}$ is n by n. □

Immediately from this we get the following corollary:

Corollary 5.6.5 *$\mathbf{S}_M$ can be thought to partition each class $\mathcal{E}_i$ into a partition of "subclasses" $\mathcal{T}_i^z$ where $X \in \mathcal{T}_i^z$ if $X \in \mathcal{E}_i$ and $\mathbf{S}_M X = z$.*

Note that this is not necessarily true of a generic linear extractor.

This allows us to prove the following Theorem. Let $\mathsf{Fsk}^*(w) = \{\delta, \mathsf{Fsk}(\delta(w)) | \delta \in \Delta\}$.

Theorem 5.6.6 $H_\infty(W|\mathsf{Fsk}^*(W), \mathbf{S}'_M w) \geq m' - l_{MAC}$.

Proof. We prove this theorem by bounding the min-entropy for $\delta = \delta_0$, then showing that by the linearity of this construction no further min-entropy is lost for $\delta \in \Delta_\oplus \neq \delta_0$. With overwhelming probability we may assume $\frac{\mathbf{H}}{\mathbf{S}}$ is of full rank.

By the reusability of Gen we know that Gen divides $\mathcal{M}$ into equivalence classes $\mathcal{E}_i$ such that $\mathsf{Gen}(w_i) = Q_i$ for all $w_i \in \mathcal{E}_i$. By the previous lemma we can say that $\mathbf{S}$ divides each class $\mathcal{E}_i$ into subclasses $\mathcal{T}_i^\mu$ where $\forall w, w' \in \mathcal{T}_i^\mu$, $\mathsf{Gen}(w) = \mathsf{Gen}(w')$, $\mathbf{S}_M w = \mathbf{S}_M w'$. Since $\mathbf{S}_M$ is a permutation on each class, for each class $\mathcal{E}_i$, there are $2^{l_{MAC}}$ classes $\mathcal{T}_i^\mu$. As such, by Lemma 5.4.1 and the previous sentence there are at most $2^{m-m'+l_{MAC}}$ equivalence classes and by setting $m = n$, the maximum entropy, each class is of size $2^{m'-l_{MAC}}$.

We now consider an adversary who makes queries to $\mathsf{Fsk}(\delta(w))$ where $\delta \neq \delta_0$. Since the sketch is deterministic and linear we know that $\mathsf{Gen}(\delta(w))$ can be calculated directly from $\mathsf{Gen}(w)$ and δ. We also know that if $\mu = \mathbf{S}_M w$, then $\mu_{\delta_x} = \mathbf{S}_M \delta_x(w) = \mathbf{S}_M w \oplus \mathbf{S}_M x$. As such, the adversary can pre-calculate the values of Q and τ before making the query to Fsk and as such the query adds no additional information. □

We now prove that (Fsk, Rep) is an $(m, l_{KEY}, d, \Delta_\oplus, \epsilon)$ reusable fuzzy extractor.

Theorem 5.6.7 (Fsk, Rep) *is an $(m, l_{KEY}, d, \Delta_\oplus, \epsilon)$ reusable fuzzy extractor.*

Proof. This comes from the fact that $\mathbf{S}w$ is a linear extractor for $\Delta_\oplus$ by the fact that it is xor-universal and the leftover hash lemma. As such $\mathbf{S}_K w$ is statistically close to random,

even given τ and $\mathbf{S}_M w$. We maintain the advantage for $\mathbf{S}_K \delta_x(w)$ as $\Delta_\oplus$ is a group and as such any adversary capable of calculating $\mathbf{S}_K \delta_x(w)$ can calculate $\mathbf{S}_K w$. □

We now show our construction is strongly post-application robust.

Theorem 5.6.8 $(\mathsf{Fsk}, \mathsf{Rep})$ *is a strongly post-application robust fuzzy extractor for* $\Delta = \Delta_\oplus$. *Specifically, for all polynomial time adversaries* A, $\mathsf{ADV\text{-}ROBUST}'_{fe-post}(\mathsf{A}, \mathsf{Gen}, \mathsf{Rec}, \Delta_\oplus) \leq \mathsf{ADV\text{-}MAC_{RK}}(\mathsf{A}, \mathsf{MAC}, l_{MAC}) + 2^{-m'+l_{MAC}}$

Proof. Before giving the reduction, we remind ourselves that since we are selecting $\mathbf{S}$ randomly $\mathbf{S}w$ is an xor-universal hash function. Thus, by the leftover hash lemma we can consider μ to be random, even given $\mathbf{S}$. Moreover, we can consider R to be statistically indistinguishable from random even given $\mathbf{S}$ and μ.

We now transform an adversary A who violates post-application robustness to an A' which defeats the related-key security of the MAC. A' plays the part of the challenger, using his oracle to help him create sketches. A' first selects a (d, m)-pair W, W' and samples them to obtain w, w'. When A requests a sketch $\mathsf{Fsk}(\delta_i(w))$, A' computes $Q_i = \mathsf{Gen}(\delta_i(w))$, $\delta'_i = \mathbf{S}_M \delta_i$, and asks for $\tau_i = \mathcal{O}^{rel}(Q_i, \delta'_i)$. A' then returns $P_i = (Q_i, \tau_i)$ to A. When A makes a key query for query j, A' returns a random value R_j. Eventually A returns a sketch $(Q^*, \tau^*), \delta^*$ and A' outputs Q^*, τ^* as its forgery under the key offset $\mathbf{S}_M \delta^*$.

The main difficulty in the proof is that when A' makes a sketch Q of a query w, and asks for a MAC of Q, the MAC oracle $\mathcal{O}^{rel}$ will with high probability not be using the key $\mathbf{S}_M w$ and will rather be using a different random key K.

To overcome this difficulty we note that for the sketch Q there is a secret w' such that $\mathbf{S}_M w' = K$ and that $Q = \mathsf{Gen}(w')$. This comes from the fact that $\mathbf{S}_M$ divides each equivalence class $\mathcal{E}_i$ into subclasses, and there is one subclass for each value K, (because $\mathbf{S}_M$ is surjective). Moreover, the subclass of the secret w is information theoretically hidden just given the sketches. Thus the sketches produced by A' can be considered valid sketches of an appropriately chosen secret w', and as such A receives a consistent transcript of sketches. As for the one key query, due to Theorem 5.6.7 we know that each R_i individually is statistically

indistinguishable from random even given μ and all Q_i's. As such, the adversary A cannot tell the difference between receiving a random R_i and the correct one. Therefore, A' forges with the same probability as the chance that A breaks strong robustness, unless A happens to guess the correct value w a probability which is bounded by $2^{-m'+l_{MAC}}$.

By this bound on $\mathsf{ADV\text{-}ROBUST}'_{fe-post}$ and Theorem 5.6.6, we have that $(\mathsf{Fsk}, \mathsf{Rep})$ is strongly post-application robust. □

We note that if we are in the random oracle model we can create a strongly post application robust extractor from more general components. Our construction is again similar to [19], and is also similar to the insider-secure construction of Boyen [13]. As Boyen does, we prove our construction in a "limited-query" random oracle model; we assume $\mathcal{O}$ is a random oracle giving l-bit outputs, and let $l_{TAG} + l_{KEY} = l$. Let $(\mathsf{Gen}, \mathsf{Rec})$ be any (m, m', d, Δ) reusable sketch.

Definition 5.6.9 $(\mathsf{Fsk}(w))$

1. *Compute* $Q = \mathsf{Gen}(w)$.
2. *Select a random value* r.
3. *Compute* $X = \mathcal{O}(w, Q, r)$. *Denote the first* l_{TAG} *bits as* τ, *the last* l_{KEY} *bits as* R.
4. *Output* $P = ((Q, r), \tau), R$.

Definition 5.6.10 $(\mathsf{Rep}(w', P))$

1. *Compute* $w'' = \mathsf{Rec}(w', Q)$.
2. *Compute* $\tau' || R' = \mathcal{O}(w'', Q, r)$, *where* $|\tau'| = l_{TAG}$.
3. *If* $\tau = \tau'$, *output* R', *else output* $\perp$.

Theorem 5.6.11 $(\mathsf{Fsk}, \mathsf{Rep})$ *is a* $(m, l_{KEY}, d, \Delta, \epsilon)$ *strongly robust fuzzy extractor in the limited random oracle model.*

Proof. Since Gen and Rec are a reusable fuzzy sketch, the min-entropy of w given all Q_i values received by the adversary is m'. Consider each tuple (Q_i, τ_i, r_i, R_i). We claim that the additional values τ_i, R_i do not substantially reduce the min-entropy of w.

Let S be the set of values w such that there is a Q and an r such that the adversary queries $\mathcal{O}(w, Q, r)$. Note that in the limited random oracle model, we may assume that $|S|$ is polynomial in 1^m. Unless the real w is in S, all that is learned from these random oracle queries is that $w \notin S$. This information eliminates at most $|S|$ values, each with probability at most $2^{-m'}$ given the known Q. Thus, $H_\infty(W|(Q_i, \tau_i, r_i), \mathcal{O}(S)) \geq m' - \log(1 - |S|2^{-m'})$. We denote $m' - \log(1 - |S|2^{-m'})$ as α.

By Boyen and [2] we know that a random oracle represents an optimal randomness extractor, and thus for each i, and r, $SD(\langle \mathcal{O}(w, Q, r), r, Q\rangle, \langle U_l, r, Q\rangle) \leq \epsilon$ where $\epsilon = \sqrt{2^{l_{KEY}-\alpha}} = \sqrt{2^{l_{KEY}-\alpha}}$. □

Corollary 5.6.12 *The min-entropy of w, given* $\mathsf{Fsk}(w) = Q_i, \tau_i, r_i, R_i$ *is $\geq \alpha$ with overwhelming probability, given only a polynomial number of sketches.*

Theorem 5.6.13 Fsk *and* Rep *constitute a strongly post-application robust extractor.*

Proof. The probability that the adversary makes a successful forgery is the probability that the adversary can either guess the correct w, or that the sketch Q', r', τ' is such $\mathcal{O}(w, Q', r') = \tau'$. By Corollary 5.6.12 the min-entropy of w given the tuples Q_i, τ_i, r_i, R_i is α. For any tuple Q', τ', r' the probability that for a given w, $\mathcal{O}(w, Q', r') = \tau'$ is $\frac{1}{2^{l_{TAG}}}$. Thus, the forgery probability of the adversary is $2^{-\alpha} + \frac{1}{2^{l_{TAG}}}$. □

5.7 Insider Security

Boyen [13] introduced the notion of an "insider secure" fuzzy extractor – a fuzzy extractor which is secure even when the adversary is allowed to see the extracted values for adversarily generated sketches and permutations. We prove that our strongly robust fuzzy extractor is insider secure.

Definition 5.7.1 (Insider Security) *A fuzzy extractor* Fsk *and* Rep *is considered to be insider secure for an adversary* A *if* ADV-INSIDE(A, Fsk, Rep, Δ) *is negligible, where* ADV-INSIDE (A, Fsk, Rep, Δ) *is the probability of* A *winning in the following game:*

Setup: *The challenger samples W to obtain w.*

Pre-challenge Queries: *The adversary* A *presents up to q queries to the challenger where each query is either a* public *or* private *query.*

Public Queries: A *selects $\delta_i \in \Delta$ and receives P_i where* $\mathsf{Fsk}(\delta_i(w)) = P_i, R_i$.

Private Queries: *The adversary selects $\delta_i \in \Delta$ and a public sketch P_i' and receives* $\mathsf{Rep}(\delta_i(w), P_i') = R_i$.

Challenge: *The adversary selects any public sketch P^* that was returned via a public query, under the constraint that for all private queries δ_i, P_i' such that $P_i' = P^*$, δ_i must have the property that for all $w \in \mathcal{M}$, $||\delta_i(w) - w|| > d$.*

Post-challenge Queries: *The adversary may make further private and public queries, with the stipulation that no private query δ_i, P^* can be made unless $||\delta_i(w) - w|| > d$ for all w.*

Test: *The adversary succeeds if he outputs R^* such that* $\mathsf{Rep}(w, P^*) = R^*$.

We now give a construction of a insider secure fuzzy extractor. Our construction is similar to the construction in Section 5.6 with the exception that instead of a deterministic linear sketch, we use the code offset sketch of Theorem 5.2.3. Let Gen and Rec be the codeoffset sketch using a code with parity check matrix $\mathbf{H}$. Let S_M and S_K be defined as in Section 5.6 with respect to this $\mathbf{H}$.

Definition 5.7.2 (Fsk)

$Q = \mathsf{Gen}(w)$.

$\mu = \mathbf{S}_M w$.

$\tau = \mathsf{MAC}_\mu(Q)$.

$R = \mathbf{S}_K w$.

Output $P = (Q, \tau)$ and R.

Definition 5.7.3 ($\mathsf{Rep}(w', P')$)

Let $w'' = \mathsf{Rec}(w', Q')$.

$\mu' = \mathbf{S}_M w''$

$\tau' = \mathsf{MAC}_{\mu'}(Q')$

If $\tau' = \tau$ *output* $\mathbf{S}_K w''$ *else output* $\perp$.

Theorem 5.7.4 *Our construction in Section 5.6, replacing* Gen *and* Rec *by the code offset sketch is an insider secure fuzzy extractor.*

Proof. This extractor can be shown to be reusable via the same techniques in Theorem 5.6.5, Theorem 5.6.4 and Theorem 5.6.6. We demonstrate how any adversary playing the game defined for the advantage ADV-INSIDE can simulate its private queries. Once we do that, we can limit ourselves to adversaries which make only public queries. The rest of the proof follows from the idea that if an adversary makes only public queries, the game defined for advantage ADV-INSIDE is the same as the reusability of the fuzzy extractor.

There are two cases. If A makes a private query with a Q never returned from a public query he can simulate the result by outputting $\perp$. This is because if A can create a new Q, τ pair that will not return $\perp$, then $\mathsf{ADV\text{-}ROBUST}'(\mathsf{A}, \mathsf{Fsk}, \mathsf{Rep}, \Delta)$ is non-negligible.

We now deal with the case where A makes a private query using a Q, τ, δ_y tuple returned in a public query. The first time this occurs, A can simulate the output by selecting a random l_{KEY} sized bit string R. For all subsequent private queries of this type, if the associated public query was made with δ_x, then A knows the output of Rep will be $R \oplus \mathbf{S}_K(x \oplus y)$ due to the linearity of the extractor.

We next deal with the case where A makes a private query using a Q, τ, pair that was returned in a public query, while specifying a different $\delta_{x'} \neq \delta_x$. In this case, then $\mathsf{Rec}(w \oplus x', w \oplus x \oplus \mathsf{C}(r)) = w \oplus x \oplus \mathsf{C}(r) \oplus \mathsf{C}(\mathsf{D}(x' \oplus x \oplus \mathsf{C}(r)))$. If $x \oplus x'$ has weight less than d, then $\mathsf{C}(\mathsf{D}(x \oplus x' \oplus \mathsf{C}(r))) = \mathsf{C}(r)$ and as such Rec recovers $w \oplus x$ and as such we are in the previous case. If $x \oplus x'$ has weight greater than d, then $\mathsf{C}(\mathsf{D}(x' \oplus x \oplus \mathsf{C}(r))) = \mathsf{C}(r')$, a different codeword, if the error correcting program D can run at all. As such Rec either outputs $\perp$ or $w \oplus x \oplus \mathsf{C}(r \oplus r')$. Since $\mathbf{S}_M w \oplus x \oplus \mathsf{C}(r \oplus r') = \mathbf{S}_M \mathsf{C}(r \oplus r') \oplus \mu$, where μ is the key used in the previous invocation of the sketch protocol, any adversary making this

type of query can be said to know the offset between the previous key μ and the new key μ'. As such, the adversary can simulate all queries in this case by outputting $\perp$.

As this covers all the possible private queries, our proof is complete. □

We note that the main reason why we needed to use the code offset sketch in our insider secure fuzzy extractor, as opposed to a generic reuseable sketch, was that we required that $\mathsf{Rec}(w \oplus x', \mathsf{Gen}(w \oplus x))$ produce a known offset of $w \oplus x$, based only on x and x'. This "linearity" property is not just found in the code-offset sketch, it is common to all known reusable sketches, including the code offset sketch, its equivalent sketch the "syndrome" sketch, and the generic reusable sketch made by Boyen [13].

5.8 Related Key Attacks and Authentication

The related key security of various cryptographic protocols is a relatively recent issue. On one side, it has been found that many widely used constructions (such as DES and AES) are insecure when subject to related key attacks and that it is difficult to construct new protocols that are secure against related key attacks. On the other hand, it is somewhat difficult to see how a related key attack could be practically applied to many different cryptographic protocols, especially when those protocols are used in isolation as one usually assumes that the key is stored in as secure a location as possible.

Consider a strongly post-application robust fuzzy extractor with the additional property that for any sketch $P \leftarrow \mathsf{Fsk}(w)$ and for any isometric permutation $\delta : ||\delta(w) - w|| > d$ where d is the error correcting distance of Rep we have that $\mathsf{Rep}(\delta(w), P) = \perp$. We note that our previous construction in the standard model satisfies this property as long as with overwhelming probability $\mathsf{MAC}_\mu(Q) \neq \mathsf{MAC}_{\mu \oplus x}(Q)$ for adversarially chosen x and random μ and that our construction in the random oracle model also satisfies this property. If a fuzzy extractor possesses this property we consider it to be "well-formed". We can show that such a well-formed strongly post-application robust fuzzy extractor enables us to construct a related-key secure MAC.

Let $\mathsf{MAC}^{rel}_{w,P}$ where $P \leftarrow \mathsf{Fsk}(w)$ be the construction that on input m, first computes $R_i \leftarrow \mathsf{Rep}(w, P)$. If $R_i = \perp$, MAC^{rel} outputs $\perp$, else $\mathsf{MAC}^{rel}_{w,P}(m)$ outputs $\mathsf{MAC}_{R_i}(m)$ where MAC is a normal, not necessarily related key secure MAC.

Theorem 5.8.1 *If* Fsk *and* Rep *are a well-formed strongly post-application robust fuzzy extractor and* MAC_K *is a secure MAC, then* MAC^{rel} *is a related key secure MAC.*

Proof. Let A be the adversary which can successfully break the related key security of MAC^{rel}. We will show that A must either break the security of MAC_K or the strong-robustness of Fsk and Rep.

We construct $\mathsf{A}'^{\mathcal{O}^{MAC}}$ to break the security of MAC_K as follow. Denote the queries made by A as $((\delta^w, \delta^P), m)$ where δ^P represents the change to the sketch and δ^w represents the change to w. When A makes a query $((\delta^w_x, \delta^P_y), m)$, A' examines δ^w_x, δ^P_y. If δ^P_y is not the identity or if δ^w_x is such that $||\delta^w_x(w) - w|| > d$ for all w, A' returns $\perp$. Otherwise A' returns $\mathcal{O}^{MAC}(m)$. When A returns its forgery $(\delta^w_x, \delta^P_y), m^*, \tau^*$, A checks the delta values again. If δ^P_y is the identity and δ^w_x is such that $||\delta^w_x(w) - w|| \leq d$ then A' outputs m^*, τ^* as its forgery on MAC_K.

We show that A' provides a correct simulation of $\mathcal{O}^{rel}$. We note that if for any query made by A if δ^P_y is not the identity then with overwhelming probability $\mathsf{Rep}(w, \delta^P_y(P)) = \perp$, due to the strong-robustness of the fuzzy extractor. Due to the well formed property of our extractor we know that if δ^P_y is the identity and δ^w_x is such that $||\delta^w_x(w) - w|| \leq d$ then $\mathsf{Rep}(\delta^w_x(w), P) = \mathsf{Rep}(w, P)$, otherwise $\mathsf{Rep}(\delta^w_x(w), P) = \perp$. As such we can say that A' provides an accurate simulation of $\mathcal{O}^{rel}$ for all queries made by A. When a query made by A will return a result besides perp, it will return $\mathsf{MAC}_K(m)$ for the same random key K. Moreover, since A successfully forges on MAC^{rel} with non-negligible probability, with non-negligible probability we can say that m^*, τ^* is such that $\tau^* = \mathsf{MAC}_K(m^*)$, and also that m^*, τ^* is not a valid message tag pair for a different key K'. □

This allows us to construct related-key secure MAC's from any already existing MAC primitive in the limited CRS model of our first construction, or in the random oracle model

used in the second construction. If a well-formed, strongly robust fuzzy extractor exists in the standard model, we then gain the ability to construct related-key secure MAC's from any MAC primitive in the standard model. Note that this does not give us the ability to construct related-key secure PRP's due to the fact that the sketch is not pseudorandom and the fact that related-key secure PRP's are not allowed to return $\perp$ on a related key query made by the adversary, as that allows an efficient distinguishing attack.

Bibliography

[1] BLUM A., FURST M., KEARNS M., AND LIPTON R.J. Cryptographic primitives based on hard learning problems. In *CRYPTO-93*, pages 278–291, 1993.

[2] JAIKUMAR RADHAKRISHNAN AMNON AND AMNON TA-SHMA. Tight bounds for depth-two superconcentrators. In *In Proc. of FOCS*, pages 585–594, 1997.

[3] DANA ANGLUIN AND MICHAEL KHARITONOV. Why won't membership queries help? In *23rd ACM Symposium on Theory of Computing*, pages 444–454, 1991.

[4] DANIEL AUGOT AND MATTHIEU FINIASZ. A public key encryption scheme based on the polynomial reconstruction problem. In *EUROCRYPT 2003*, page 645, 2003.

[5] JOONSANG BAEK, WILLY SUSILO, AND JIANYING ZHOU. New constructions of fuzzy identity-based encryption. Cryptology ePrint Archive, Report 2007/047, 2007. http://eprint.iacr.org/.

[6] M. BELLARE AND T. KOHNO. A Theoretical Treatment of Related-Key Attacks: PKA-PRPs, RKA-PRFs, and Applications. In *Advances in Cryptology – EUROCRYPT '03*, E. Biham, editor, volume 2656 of LNCS, pages 491–506, 2003.

[7] CHARLES BENETT, GILES BRASSARD, CLAUDE CRPEAU, AND UELI M. MAURER. Generalized privacy amplification. In *IEEE Transactions on Information Theory*, volume 41, pages 1915 – 1923, 1995.

[8] CHARLES BENETT, GILES BRASSARD, AND JEAN MARC ROBERT. Privacy amplification by public discussion. In *SIAM Journal on Computing*, volume 17, pages 210–229, 1988.

[9] ELI BIHAM. New types of cryptanalytic attacks using related keys. *Journal of Cryptology*, 7(4):229–246, Fall 1994. Also available at: `citeseer.nj.nec.com/biham94new.html`.

[10] A. BIRYUKOV, D. KHOVRATOVICH, AND I. NIKOLIC. Distinguisher and related-key attack on the full AES-256. In *CRYPTO 2009*, 2009.

[11] J. BLACK, M. COCHRAN, AND T. SHRIMPTON. On The Impossibility of Highly-Efficient Blockcipher-Based Hash Functions. In *Advances in Cryptology – Eurocrypt 2005*, volume 3494 of *LNCS*, pages 526–541. Springer Verlag, May 2005.

[12] AVRIM BLUM, ADAM KALAI, AND HAL WASSERMAN. Noise-Tolerant Learning, the Parity Problem, and the Statistical Query Model, 2000.

[13] XAVIER BOYEN. Reusable cryptographic fuzzy extractors. In *ACM Conference on Computer and Communications Security—CCS 2004*, pages 82–91. New-York: ACM Press, 2004.

[14] XAVIER BOYEN, YEVGENIY DODIS, JONATHAN KATZ, AND RAFAIL OSTROVSKY. Secure remote authentication using biometric data. In *Eurocrypt*, pages 147–163, 2005.

[15] EMMANUEL J. CANDES, MARK RUDELSON, TERENCE TAU, AND ROMAN VERSHYNIN. Error correction via linear programming. FOCS, 2005.

[16] JOSE CARRIJO, RAFAEL TONICELLI, HIDEKI IMAI, AND ANDERSON C A NASCIMENTO. A Novel Probabilistic Passive Attack on the Protocols HB and HB^+. Cryptology ePrint Archive, Report 2008/231, 2008. http://eprint.iacr.org/.

[17] EE CHIEN CHANG, VADYM FEDYUKOVYCH, AND QIMING LI. Secure Sketch for Multi-Sets. Cryptology ePrint Archive, Report 2006/090, 2006.

[18] EE CHIEN CHANG AND QIMING LI. Small secure sketch for point-set difference. Cryptology ePrint Archive, Report 2005/145, 2005.

[19] RONALD CRAMER, YEVGENIY DODIS, SERGE FEHR, CARLES PADRO, AND DANIEL WICHS. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Advances in Cryptology - EUROCRYPT*, pages 471–488, April 2008.

[20] YEVGENIY DODIS, JONATHAN KATZ, LEONID REYZIN, AND ADAM SMITH. Robust fuzzy extractors and authenticated key agreement from close secrets. In *CRYPTO*, pages 232–250, 2006.

[21] YEVGENIY DODIS, RAFAIL OSTROVSKY, LEONID REYZIN, AND ADAM SMITH. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *SIAM Journal on Computing*, volume 38, pages 523–540, 2008.

[22] YEVGENIY DODIS, LEONID REYZIN, AND ADAM SMITH. Fuzzy extractors: How to generate string keys from biometrics and other noisy data. In *Eurocrypt*, pages 523–540, 2004.

[23] LIMING FANG. Full security: Fuzzy identity based encryption. Cryptology ePrint Archive, Report 2008/307, 2008. http://eprint.iacr.org/.

[24] LIMING FANG, JIANDONG WANG, YONGJUN REN, JINYUE XIA, AND SHIZHU BIAN. Chosen-ciphertext secure fuzzy identity-based key encapsulation without rom. Cryptology ePrint Archive, Report 2008/139, 2008. http://eprint.iacr.org/.

[25] MARC P. C. FOSSORIER, MIODRAG J. MIHALJEVI, HIDEKI IMAI, YANG CUI, AND KANTA MATSUURA. A Novel Algorithm for Solving the LPN problem and its Application to Security Evaluation of the HB protocol for RFID authentication, 2006.

[26] NIKLAS FRYKHOLM AND ARI JUELS. Error-tolerant password recovery. In *8th ACM conference on Computer and Communications Security*, pages 1 – 9, 2001.

[27] HENRI GILBERT, MATTHEW J.B. ROBSHAW, AND YANNICK SEURIN. An active attack against HB^+ - a provably secure lightweight authentication protocol. In *ePrint*, 2005.

[28] ZBIGNIEW GOLEBIEWSKI, KRZYSZTOF MAJCHER, FILIP ZAGORSKI, AND MARCIN ZAWADA. Practical Attacks on HB and HB^+ Protocols. Cryptology ePrint Archive, Report 2008/241, 2008. http://eprint.iacr.org/.

[29] MICHAEL GORSKI AND STEFAN LUCKS. New related-key boomerang attacks on aes. In *INDOCRYPT*, Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, volume 5365 of *Lecture Notes in Computer Science*, pages 266–278. Springer, 2008.

[30] YANNICK SEURIN HENRI GILBERT, MATTHEW J.B. ROBSHAW. HB#: Increasing the security and efficiency of HB^+. In *Eurocrypt 2008*, pages 361–378, 2008.

[31] N. HOPPER AND M. BLUM. Secure human identification protocols. In *Advances in Cryptology - ASIACRYPT 2001*, pages 52–56, 2001.

[32] ARI JUELS AND MARTIN WATTENBERG. A fuzzy commitment scheme. In *Sixth ACM Conference on Computing and Communication Security*, pages 28–36, 1999.

[33] ARI JUELS AND STEPHEN WEIS. Authenticating pervasive devices with human protocols. In *Crypto*, pages 293–308, 2005.

[34] ARI JUELS AND STEPHEN WEIS. Defining strong privacy for RFID. In *Pervasive Computing and Communications Workshops*, pages 342–347, 2007.

[35] BHAVANA KANUKURTHI AND LEONID REYZIN. An improved robust fuzzy extractor. In *SCN*, pages 156–171, 2008.

[36] JONATHAN KATZ. Efficient cryptographic protocols based on the hardness of learning parity with noise. In *IMA Int. Conf.*, pages 1–15, 2007.

[37] MICHAEL KEARNS AND LESLIE G. VALIANT. Cryptographic limitations on learning boolean formulae and finite automata. In *21st ACM Symposium on Theory of Computing*, pages 433–444, 1989.

[38] JOHN KELSEY, BRUCE SCHNEIER, AND DAVID WAGNER. Related-key cryptanalysis of 3-way, biham-des, cast, des-x, newdes, rc2, and tea. In *ICICS '97: Proceedings of the First International Conference on Information and Communication Security*, pages 233–246, London, UK, 1997. Springer-Verlag.

[39] AGGELOS KIAYIAS AND MOTI YUNG. Cryptographic hardness based on the decoding of Reed-Solomon codes with applications. In *Proceedings of ICALP 2002, LNCS 2380*, pages 232–243, 2002.

[40] AGGELOS KIAYIAS AND MOTI YUNG. Cryptanalyzing the polynomial-reconstruction based public-key system under optimal parameter choice. *Des. Codes Cryptography*, 43(2-3):61–78, 2007.

[41] STEFAN LUCKS. Ciphers secure against related-key attacks. In *FSE*, Bimal K. Roy and Willi Meier, editors, volume 3017 of *Lecture Notes in Computer Science*, pages 359–370. Springer, 2004.

[42] VADIM LYUBASHEVSKY. The Parity Problem in the Presence of Noise, Decoding Random Linear Codes, and the Subset Sum Problem. In *Approximation, Randomization and Combinatorial Optimization*, pages 378–389, 2005.

[43] CHETAN NANJUNDA MATHUR, KARTHIK NARAYAN, AND K. P. SUBBALAKSHMI. High diffusion cipher: Encryption and error correction in a single cryptographic primitive. *Lecture Notes in Computer Science*, pages 309–324, 2006.

[44] CHETAN NANJUNDA MATHUR, KARTHIK NARAYAN, AND K. P. SUBBALAKSHMI. On the design of error-correcting ciphers. *EURASIP J. Wirel. Commun. Netw.*, 2006(2):72–72, 2006.

[45] ALFRED J. MENEZES, PAUL C. VAN OORSCHOT, AND SCOTT A. VANSTONE. *Handbook of Applied Cryptography.* CRC Press, 1997.

[46] DARAKHSHAN J. MIR AND POORVI L. VORA. Related-key statistical cryptanalysis. Cryptology ePrint Archive, Report 2007/227, 2007. http://eprint.iacr.org/.

[47] SHAFI GOLDWASSER ODED GOLDREICH AND SILVIO MICALI. How to construct random functions. In *Journal of the ACM*, 1986.

[48] JEAN PAUL LINNARTZ ANDPIM TUYLS. New shielding functions to enhance privacy and prevent misuse of biometric templates. In *In AVBPA 2003*, pages 393–402, 2003.

[49] DARAKHSHAN J. MIR POORVI L. VORA. Related-key linear cryptanalysis. In *2006 IEEE International Symposium on Information Theory*, pages 1609 – 1613, 2006.

[50] AMIT SAHAI AND BRENT WATERS. Fuzzy identity based encryption. Cryptology ePrint Archive, Report 2004/086, 2004. http://eprint.iacr.org/.

[51] THOMAS WORSCH. Lower and Upper Bounds for (sums of) Binomial Coefficients. 1994.

[52] WENTAO ZHANG, LEI ZHANG, WENLING WU, AND DENGGUO FENG. Related-key differential-linear attacks on reduced aes-192. In *Progress in Cryptology - INDOCRYPT 2007*, 2007.

VITA

David Goldenberg

David Goldenberg was born August 25, 1983. He went to high school at W.T. Woodson in Fairfax VA. Graduating high school he attended The College of William and Mary for his undergraduate degree, obtaining a B.A. in theatre with a minor in mathematics. After graduation he went straight into the graduate computer science program at William and Mary. David is interested in many things in computer science including, but not limited to: AI, Statistical learning, Video Games, Cryptography, Information Theory, and Simulations.