BUILDING INFORMATION-THEORETIC CONFIDENTIALITY AND TRAFFIC PRIVACY INTO WIRELESS NETWORKS

BY SUHAS MATHUR

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Professor Wade Trappe and Professor Narayan B. Mandayam

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October, 2010

ABSTRACT OF THE DISSERTATION

Building Information-Theoretic Confidentiality and Traffic Privacy into Wireless Networks

by Suhas Mathur

Dissertation Directors: Professor Wade Trappe and Professor Narayan B. Mandayam

This dissertation studies how information-theoretically secure mechanisms for confidentiality and data-traffic privacy can be incorporated into existing and emerging wireless systems.

The dissertation consists of three parts. In the first two parts, we study how certain properties of wireless channels can be employed to enhance confidentiality services that have traditionally been the responsibility of higher layers. We first explore the use of the wireless medium for the extraction of secret keys at the two ends of a wireless link, wherein the transceivers at either end are separated by a rich multipath scattering environment. We build a low complexity algorithm that allows two wireless devices to extract a common sequence of random secret bits, by repeatedly probing and estimating a time-varying channel between themselves. Further, we report on an implementation and evaluation of our algorithm on a modified 802.11a system.

Next, we study the problem of securely pairing wireless devices in proximity of one another by establishing a shared secret key using a public source of RF transmissions. We employ measurement data to characterize the rate at which bits can be extracted and explore the simultaneous use of multiple transmitters to increase rate. Finally, we study the case when the public transmitter itself is under the arbitrary control of an adversary and we demonstrate a method that can allow successful key-extraction even with such an active adversary.

In the final part of this dissertation, we introduce the problem of an unintended information-leakage channel in data traffic consisting of varying packet sizes. Packet sizes convey semantic information that can be related to their content, which can be used as a fingerprint for classification. We formally study the packet-size side channel and explore obfuscation approaches to prevent information leakage, while considering padded dummy traffic and delay as bounded resources. We show that randomized algorithms for obfuscation can be studied as well known information-theoretic constructs, such as discrete channels with and without memory, and often lead to efficiently solvable constrained optimization problems.

Acknowledgements

It is possible that there are engineers that do not need the support and encouragement of other people to accomplish their goals. I am not such an engineer. I would like to acknowledge some of the people who have helped me during my graduate studies and encouraged me to do research to the best of my ability. I extend my deepest gratitude to:

- My family. My parents and grandparents constantly encouraged me in my academic pursuits. They provided an environment conducive to study, ever since grade school, always putting my studies before their own plans. Along with my sister, they patiently waited as I took my time to complete my PhD. My wife, Ramya supports and inspires me and makes my life more enjoyable by sharing hers with me.

- My colleagues and friends at WINLAB. Hithesh Nama, Chandru Raman, Joydeep Acharya and Lalitha Sankar inspired and encouraged me in the analytical and theoretical portions of my work. They were always willing to listen to my ideas and mathematical problems, and often led me in the right directions. Kishore Ramachandran, Sachin Ganu, Sanjit Kaul, Baik Hoh, Wenyuan Xu and Rob Miller greatly influenced the development of my interest in systems aspects and building and validating real systems.

- My friends from IIT Madras, who inspired and stimulated me, and continue to do so through their impressive accomplishments: Girish Redekar, Hari Balaji, Abhishek Kumar, Niteesh Kumar, Anupam Bose. I often find myself applying ideas we discovered together in the stressful environment of college

to deal with the problems of real life.

- My dear friend Neville Clemens, who inspired me to be inquisitive about the larger questions of life.

- Professor David Koilpillai, my undergraduate advisor, who inspired me to consider working on wireless communications.

- Professors Marco Gruteser and Dipankar Raychaudhuri, who while not my formal academic advisor, served as informal guides, giving valuable ideas and suggestions during the course of my graduate study, and also as collaborators and source of ideas for systems research.

- Professors Wade Trappe and Narayan Mandayam, my research advisors, who taught me dozens of things about research and writing that I didn't even know that I didn't know. They both allowed me to grow as a researcher, by letting me watch them think, and placed their confidence in my abilities by allowing me generous amounts of academic freedom and supporting me through their grants. If I'm lucky, their influence will be felt in my work for a very long time.

Dedication

To my parents, who taught me the love of learning.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Security challenges in wireless networks

The broad security challenges in existing wireless systems of today, and emerging wireless networks, can be categorized as those of (i) confidentiality, (ii) authentication and (iii) privacy. The wireless medium is markedly different from wired networks in each of these three respects. The fact that securing wireless networks is particular hard compared to their wired counterparts is evidenced by the fact that although mechanisms are in place to address each of these challenges, new and innovative attacks on these mechanisms are constantly reported [4, 5], forcing security engineers to constantly produce more effective measures, and administrators to constantly upgrade their systems. Perhaps the most important reason contributing to this challenge is that fact that while wired communications limits all data to cables, the wireless medium is inherently a broadcast medium, implying that data transmitted by a terminal A intended for a terminal B, is easily available to unintended terminals. Coupled with this is the fact that wireless systems are relatively new compared to wired networks and over the course of evolution of wired networks, security designs have been perfected for wired networks. The designs do not carry over smoothly to wireless networks. In fact, the broadcast nature of wireless affects all the three aspects enumerated above. While physically limiting data bits to a cable travelling between a source terminal to a destination terminal may be considered to be secure in some sense, broadcasting data definitely requires strong guarantees on confidentiality. That is, it is necessary to

ensure that unintended recipients of the data being broadcasted cannot decipher the intended messages. In today's wireless systems, this problem has been addressed by encrypting the data. Of course, this requires terminals to subscribe to a key distribution system, either based on public keys, or symmetric keys, and key distribution can often be a non-trivial problem in itself. Similarly, the broadcast nature of the medium also makes it necessary to have in place a mechanism to allow the recipient of data to be able to make sure that claimed sender of the data is in fact the true sender. This is the problem of authentication. Breaches in authentication would allow one wireless device to spoof another device, or one device to assume the identity of multiple devices. Finally, privacy is a problem in ways than one - the mere presence of a transmission might reveal the physical presence of a transmitter and it might be possible to assign an identity to the transmitter through a number of possible means using identifiers or peculiarities identifiable in the transmitted wireless signal [6, 7, 8]. This leads to a location privacy problem [9]. Another problem, quite distinct from the problem if identity is that even after encrypting data for the purpose of providing confidentiality, the semantics of data transmission often reveal some information about the contents of the data. For example, a particular timing pattern of packet transmissions may be associated with web browsing traffic, and might be sufficiently distinct from the traffic pattern generated by a VoIP call [10, 11] to allow automated machine classification. We will refer to this as the *semantic privacy problem*. In this thesis, we will study the problems of providing confidentiality and traffic privacy through the lens of information theory. We will then build upon our study and design algorithms and protocols that can allow our findings to be incorporated into real systems. We begin, below, by first reviewing the dominant notion of secrecy today, namely computational secrecy, and then introducing the stronger notion of information-theoretic, or unconditional secrecy.

1.2 Two notions of secrecy

The vast majority of security mechanisms in use in communication systems and networks today rely on a notion of security that implicitly assumes that the adversary is computationally bounded. That is, the ability of the system to resist attack depends critically on the validity of the fact that the adversary does not possess unlimited computing power. The main reason behind this assumption is that the field of modern cryptography has largely evolved centered around problems of mathematical intractability. The hardness of performing certain types of computations form the basis of such cryptographic mechanisms, for e.g. factoring the product of two large prime numbers which is employed as the basis in the popular RSA public key cryptography algorithm or finding the discrete logarithm of a number within a cyclic group, which forms the basis of the Diffie Hellman key exchange protocol. What would happen if someone were to discover a mechanism to solve these mathematical problems efficiently? Apart from the fact that this would be a tremendous leap in progress in the fields of mathematics and computer science, unlocking a massive set of problems of practical significance that are considered to be hard[1], this information would be so valuable by virtue of the fact that it would defeat existing cryptographic mechanisms, that it would probably be keep secret itself [12].

Fortunately, there is a more fundamental notion of secrecy, known as *information theoretic secrecy*, wherein no assumptions about the computational ability of the adversary need be made. Secrecy is achieved in this model by guaranteeing that the information to be kept secret cannot be arrived at by the adversary no matter what her computational ability. Shannon formalized this notion of secrecy in his seminal paper [13] using the measure of entropy $H(\cdot)$. The entropy $H(X)$

[1] A number of problems in computing that fall in the class of 'hard' problems have the characteristic that they can be expressed in terms of another problem in the same class. Therefore if an efficient solution to one were to be known, then all the problems in the class would become solvable by a straightforward adaptation of this solution.

of a random variable X represents the uncertainty about the value of X. There is am implicit notion of an observer in the previous statement. If the observer is the adversary, then $H(X)$ is a measure of the average number of bits that the adversary does *not* know about X if she does not know anything about X. Suppose the adversary learns a related quantity Y that conveys some information about X but doesn't give away X completely (for e.g. X might be a random integer between 1 and 10 and $Y = X \mod 2$). The uncertainty of the adversary about X is then reduced to a smaller quantity $H(X|Y)$, the conditional entropy of X given Y. The reduction in the entropy of X, namely $H(X) - H(X|Y) \triangleq I(X;Y)$ is called the mutual information and represents how much information Y reveals about X. The field of information theoretic security is built around the idea that if we can create a system in which the sum total of the observations of the adversary Y reveal statistically nothing, about the secret X, i.e. $H(X|Y) = H(X)$ or equivalently, $I(X;Y) = 0$, then we have perfect secrecy, irrespective of the what level of computing power is available to the adversary. It is important to note that this is a significantly stronger notion of secrecy because it does not make any assumptions about the adversary's computing ability. For this reason, it is also called *unconditional* secrecy.

Research in information theoretic secrecy has, with one notable exception, been relegated to theoretical advances, largely because physical systems that might provide the strong notion above do not seem to be commonplace. The exception is quantum cryptography, which relies on a central principle of quantum mechanics, namely that the act of measuring a quantum system (such as a single photon), disturbs the system. Quantum cryptography is actually a technique to allow parties separated by a distance to share a common (symmetric) key. The key itself can then be used as an input to any cryptographic algorithm (which relies on computational secrecy) or as a one-time pad [14] which retains the unconditional secrecy at the cost of having a fresh secret bit for every single

data bit transmitted. The crucial property above that quantum key distribution (QKD) relies upon, allows the system to detect whenever an eavesdropper is making observations on the quantum particles. Still quantum cryptography is cumbersome, because it requires the controlled transmission of single photons, and large expensive installations. Existing commercial ventures are aimed primarily at governments and large corporations with stringent security requirements.

1.3 Key distribution using reciprocal wireless channels

It is intriguing therefore, that simple wireless channels should possess properties that suggest that it might be possible to build an unconditionally secure key distribution system using the wireless medium, without the need for any expensive specialized equipment. One of the goals of the research outlined in this thesis is to explore whether this is indeed possible, and to study the aspects of wireless channels that can make this possible, if at all. Like quantum key distribution, the possibility of unconditionally secure key agreement using wireless channels is also based on fundamental principles of physics, albeit, very different from those relied upon in QKD. The foundation of unconditionally secure secret key distribution using wireless channels is that when a sufficient number of scaterrers are present between are transmitter and receiver, the transmitted signal travels over a large number of separate paths, bouncing off reflectors, before reaching the receiver. The multiple paths add up at the receiver to produce a signal that is a randomly distorted version of the transmit signal. The nature of the distortion is a function of all the scaterrers encountered by the signal whilst travelling from the transmitter to the receiver. The distortion is random, and most often, time varying, because the precise locations of the scaterrers and their movements are not known apriori. This provides users with a source of 'natural' randomness - they only need to estimate the channel in between themselves to sample this

source of randomness. Indeed, if wireless channels could be tapped to provide unconditionally secure secret keys, this would be very valuable to communications security because it would mean that terminals, fixed or mobile, would be able to form secure links with one another on the fly, without requiring any specialized hardware beyond what is already present in most wireless systems for channel estimation. In this thesis, we will explore how this can be achieved, what types of algorithms and protocols are necessary to build a system that can exploit the above-stated properties of wireless channels, and what are the limitations of this method.

1.4 Key distribution using public sources

A second goal of this thesis is to study a problem that combines the notions of confidentiality and authentication in a scenario that is becoming increasingly common: secure communication between wireless devices that are in proximity of one another. When two wireless devices that have never interacted before wish to form a secure link, the exchange of information for generation of shared keys is a very tricky problem. The lack of a common secret key to begin with, implies that neither device can be sure that it is really forming a secure link with the intended device. In other words, a man-in-the middle attack is easy. We should point out here that the misguided belief that low power communications are good for protecting against eavedroppers that are 'out of hearing range' is a common and potentially disastrous security design policy. For e.g. many users believe that Bluetooth is a short range radio communication protocol and therefore provides confidentiality with respect to potential eavesdroppers that are out of range. In reality, the range of communication for a wireless link is not purely a function of the transmitter, but also the receiver. In particular, it has been shown that by using simple home-improvised directional antennas on the receiver, it is possible

to vastly increase the range of Wifi and Bluetooth links [15]. To solve the problem of securely paring two wireless devices, we consider the use of public sources of airwaves, that are almost ubiquitously available, such as televisions signals, beacons broadcast by cellular base stations and even public FM radio transmissions. We postulate that transmissions from these sources can serve as public sources of randomness - simply by virtue of the transmissions from a public source, it is possible to estimate the channel between the public source and a passive receiver. When two receivers are in physical proximity of one another, then the principles of electromagnetic wave propagation suggest that the channels estimated by these devices should be highly correlated, at least in theory. To be precise, theoretical models for propagation suggest that if two receivers are within $\lambda/2$, where λ is the carrier wavelength of the public transmissions, they should experience correlated time-varying channels. As before, we postulate that the time-varying aspect of the channels can serve as a source of fresh randomness. The resistance from passive attack in this case is spatial - an adversary who is more than λ away from the legitimate receivers should not be able to observe the same stochastic processes as the ones observed by the legitimate terminals. In this thesis we take a predominantly experimental approach to first verify the claims made by theory. Is it really possible to gather common randomness at receivers when they are within $\lambda/2$? At what separation does this correlation begin to degrade, and at what distance does eavesdropping become useless? We gather measurements from a number of public sources including TV towers, cellular base stations and FM radio stations to understand the nature of the channels. Based on our observations, we design and implement algorithms that will allow for secret key generation at device in proximity without yielding any useful information to an adversary that is not co-located with the legitimate terminals. Finally, we explore a strong active attack on this system: one in which the adversary is granted control over the transmitter and is free to design her own transmitted signal. Can she control the transmitted

signal in way so as to influence the bits extracted by the legitimate terminals?

1.5 Semantic traffic privacy

In the third and final part of this thesis, we consider the semantic privacy problem. While this is a general problem that applies equally well to wired networks, its importance is all the more exacerbated in wireless networks because the broadcast nature of the medium implies that the traffic on a link can be easily monitored by an adversary. In particular, we consider the following question: How much information about data traffic is leaked out by the *size* of data packets? Our work is motivated by a number of recent attacks on semantic traffic privacy that utilize some form of traffic intensity - packet sizes or packet sending rates. For e.g. in [10, 11], it has been successfully demonstrated that monitoring the stream of packet sizes on a encrypted VoIP call can be sufficient to identify the language being spoken, and even identify some phrases, using automated classification methods. Again, we find that information theory can be used as a tool to address this problem in a way that is practical and we believe, lends to system implementation. We conduct a systematic study of how much information the size of a packet really leaks. We propose the use of randomized packet padding and randomize delaying as a cost-effective and tunable means to obfuscate the sizes of packets in streams of data traffic, and purpose a way to adjust the trade-off between the level of privacy desired and the amount of data and delay overheads that must be paid for. In many cases, we find that optimal characterization of these tradeoffs are possible. Finally, we propose the construction of an automated obfuscatory. that can function as an independent layer in the protocol stack without any side information from higher layers, and yet make decisions about how to best obfuscate packet sizes to achieve a good trade-off between privacy and the costs involved.

1.6 Layout of the thesis

The remainder of this thesis is structured as follows. In chapter 2 we discuss the general problem of treating the wireless channel as a source of randomness, from which bits can be extracted for cryptographic use. We relate the randomness available from the channel to the movement of scaterrers between the transmitter and the receiver. In chapter 3 we study the problem of extracting secret random bits from the wireless channel between two wireless terminals using an active probing approach. In chapter 4, we elaborate on how we might exploit the presence of public sources of airwaves to produce a source of common randomness for wireless devices in proximity that wish to establish a secure link. Finally, in chapter 5, we define the semantic privacy problem and present our proposed method for achieving a balance between privacy and cost. In each section, we first define the problem clearly, we present a description of our system model and assumptions, we present our main results and at the end we state important research questions that we feel remain to be answered.

Chapter 2

The Wireless Channel as a Source of Randomness

Can the wireless channel be used as a *source* of randomness for cryptographic primitives? Let us examine this question closely by listing the essential requirements that such a source must fulfill. We assume that the wireless channel in consideration has a richly scattering environment. Physically, this means that the signal from the transmitter travels through a large number of separate paths in space, and encounters a large number of reflectors and scaterrers before reaching either of the two legitimate devices. This is, in fact, the most common type of physical channel encountered in terrestrial wireless communication systems (e.g. cellular phone systems, TV broadcast systems, FM and AM radio broadcast systems). An important feature of such wireless channels is their stochastic nature. Since reflectors, scaterrers and diffractors between the transmitter and the receiver, as well as the transceivers themselves, does not remain completely stationary, the phase and magnitude of the individual paths arriving at a receiver vary with time. If the wavelength is small, then the temporal variation of phase of any path at the receiver is large even for very small movements of reflectors on the path As a result, the signals from the separate paths have random phases and thereby add up to produce a signal of random phase and amplitude. For a more detailed treatment of wireless propagation see [16].

The randomness in the behavior of the channel from the point of view of the received signal arises from the fact that precise locations and velocities of all the reflectors and scaterrers and their material properties are not known to the

receiver. It is this randomness that we propose to closely examine as a source of cryptographic keying material for receivers that are in close proximity.

Since the stochastic behavior of the wireless channel is a detriment to the communication of information over the channel, the behavior of the wireless channel has been well studied and mathematically modelled by engineers in order to design systems that suitably mitigate its adverse effects. By far the most well understood and best accepted model for the wireless channel is the Rayleigh fading model [17]. This model has been shown to well fit the case when there is no line of sight path between the transmitter and receiver. As per the Rayleigh fading model, the complex baseband representation of the *channel* for a narrowband transmit signal is a circularly symmetric complex Gaussian random variable $h(t) = h_r(t) + jh_i(t)$. The Gaussian nature of the channel follows from an application of the central limit theorem and the fact that the received signal is a composite of a large number of paths. The in-phase and quadrature components have the same variance σ_h^2. Suppose $\tilde{s}(t)$ is a high frequency narrowband signal transmitted by a transmitter such that $s(t)$ is its complex baseband representation, then the received signal in complex baseband form is given by

$$r(t) = s(t)h(t) + w(t),$$

where $w(t)$ is an additive noise component with a circularly symmetric complex Gaussian distribution, with power $\sigma)N^2/2$ each in the in-phase and quadrature components.

The search for good natural sources of randomness (as against pseudorandomness) has been as old as the field of cryptography itself. Physicists have explored the use of various sources of randomness including shot noise in electronic diodes, temperature variations of components, etc. The key requirements of a natural source of randomness are:

1. The source must truly be a random source. There is no formal way to verify

if this is true for a given source, across arbitrary time scales. If very large samples of the output of the source are available they can be examined for statistical defects. Alternatively, a mathematical model for the source can be used to study its degree of randomness. Each approach, however, is limited - while the former cannot detect defects at very large time scales, the latter is only as good as the model.

2. The output of the source must not be accessible to the adversary, the entity that wishes to break the encryption system that uses the output of the random source to generate keying material.

3. The attacker must not be able to control the output of the random source without giving away his presence to the legitimate receiver. This is a restatement of the second condition above. However, it deserves special attention because a number of attacks have focussed on manipulating the output of a supposedly random source so as to provide the adversary with complete of partial information about the output of the source.

Let us examine the candidature of the wireless channel as a source of randomness with respect to each of the three conditions above. For this we will assume a fixed transmitter that transmits a publicly known signal. Without loss of generality, for our discussion we will assume the the transmit signal is a narrowband pilot tone at a known frequency. Any receiver can tune in to this frequency and use the received signal to derive channel estimates which can then be used to extract randomness. There may be multiple receivers utilizing the pilot signal from the transmitter as a 'service' - as long as they are sufficiently separated in space (compared to the wavelength λ of the tone), they will each experience an uncorrelated channel between the transmitter and themselves. Let us suppose the signal transmitted by the transmitter takes N paths to reach the receiver antenna, where N is usually a very large number in a richly scattering environment

(in free space, $N = 1$). Since each path has a different time-varying path length, the signal along these paths arrives at the receiver at different phases; moreover, the movement of scaterrers and reflectors, even by time absolute amounts, causes the signal at the receiver to have a time-varying phase. This is because at large frequencies, the wavelength is so small, that even a tine movement of a scaterrer changes length of the path involving that scaterrer by multiple wavelengths, causing the received phase to change unpredictably. Hence the phase of the signal obtained by adding the signals with similar path lengths is random. The signal from each path can be thought of as a phasor with a random phase and an amplitude equal to the transmit amplitude multiplied by the path loss incurred over that path. The sum of a large number of such phasors is a phasor with a random phase and magnitude. In the Rayleigh fading channel model, this phase is in fact uniformly distributed in the interval $[0, 2\pi]$, and the magnitude has Rayleigh distribution. If we denote the amplitude scaling of the i^{th} path by $H_i(t)$ and the phase of the i^{th} path by $\theta_i(t)$, then the received signal can be written as:

$$r(t) = \sum_{i=1}^{N} H_i(t).e^{j\theta_i(t)} s(t - \delta_i), \tag{2.1}$$

where δ_i is the time delay of the i^{th} path. The received signal can be written as a convolution of the transmit signal and a time-varying quantity called the impulse response of the channel. This allows us to view the channel as a time-varying linear system. When the number of paths is large (as in a richly scattering environment), the impulse response of the channel between the two points,

$$h(\tau, t) = \sum_{i=1}^{n} H_i(t).e^{j\theta_i(t)} \delta(\tau - \delta_i) \tag{2.2}$$

is, to a large extent, unpredictable in both, space and time because predicting it would require precise knowledge of the locations and velocities of all scatterrers in the environment as well as their reflectivities and absorption coefficients. If the number of paths is very large, then by the central limit theorem, the in-phase

and quadrature-phase components of the channel impulse response approach independent Gaussian random variables with equal variance and zero mean (assuming there is no line of sight path)[1] and as a result, the magnitude of the channel impulse response at any given time approaches a Rayleigh random variable and the phase approaches a uniform random variable. The time varying nature of the paths causes the impulse response to be time varying as well. Therefore, the impulse response of the channel between the two points can be modelled as a complex Gaussian stochastic process. Therefore, under the assumption that it is not possible for an adversary to obtain precise information of all scatterrers and instantaneous velocities of the transmitter and receiver, the first condition above is met.

The channel response $h(t, \tau)$ also decorrelates rapidly in space. In fact, over a distance of the order of half a wavelength λ, the channel impulse response is completely statistically uncorrelated. For Gaussian distributed components of the impulse response, such as the in phase and quadrature phase components of the Rayleigh fading process, this implies statistical independence. The physical reason behind this is that the individual paths along which the signal travels between the transmitter and the receiver change their phases randomly as the receiver moves from it's original position, thereby resulting in a a sum that has a random phase and magnitude. This implies that the output of the wireless channel (in this case, the channel impulse response between the transmitter and the receiver) will not available to an eavesdropper that a distance of at least $\lambda/2$ away from the receiver, thereby satisfying the second condition.

The question the arises as to whether an adversary can control the random output of the wireless channel (requirement 3 above). Here, by 'output' we mean the estimate of the channel, made by a receiver that naively assumes that the

[1]A a direct line of sight path is present, the in-phase and quadrature phase components are Gaussian distributed with non-zero means. The amplitude of the channel impulse response then has a Ricean distribution.

transmitter is not controlled by an adversary. In the remainder of this section, we will attempt to address the question of what happens if Eve transmits the pilot signal that a legitimate user is listening to? In particular, can she influence the channel estimate made by the receiver (and thereby influence the bits obtained)? We will stick to our assumption that the channel between the transmitter controlled by Eve and a legitimate receiver is a richly scaterring environment.

We note first that the phase of the received signal is not controllable by Eve because she does not have control over the positions of the scatterrers and their temporal movements, as well as the positions. However, when multiple parties are involved, as in the case when the channel is used as a source of common randomness at Alice and Bob, accurate estimation of the phase of the channel estimate is difficult because it is dependent upon the local oscillators (LO) at Alice and Bob which are neither phase-synchronized synchronized not are they guaranteed to have the same drift. However, if we focus on the *change* in phase instead of the actual value of the phase, then we may still be able to design a system in which secret bits can be extracted even when the adversary is the transmitter and can modulate the transmit signal. Even if the LOs at Alice and Bob have a slow drift (slow compared to the coherence time of the channel), it is still possible for them to observe highly correlated phase changes between one observation epoch and the next. Further, if the sampling rate is high (compared to the maximum Doppler), we can use averaging to diminish the phase noise in the observations and thereby improve the correlation between the differential phase.

This idea can be extended to allow Alice and Bob to extract differential phase information from any (modulated) signal by using a narrow band filter. The signal observed by them has a phase component that is varying because of the modulation, as well as because of the channel. However, if they only observe the phase at time instants that are separated by a coherence time interval or more,

then the difference in phase between successive observation epochs is uncorrelated and the transmitter has no information about these phase differences.

Let us assume that the legitimate public source transmits a pilot tone $\tilde{s}(t) = A\cos(2\pi f_c t)$ which is represented as a complex baseband equivalent in the complex plane by $s(t) = A$ such that its baseband representation is $\tilde{s}(t) = Re\{s(t) \cdot e^{j2\pi f_c t}\}$. Let the overall multipath fading channel be represented by a time varying phasor $h(t) = H(t) \cdot e^{j\theta(t)}$ in this complex plane The received signal at the receiver can then be written as

$$\begin{aligned} r(t) &= s(t) \cdot h(t) + w(t) && (2.3) \\ &= AH(t)e^{j(2\pi f_c t + \theta(t))} + w(t) && (2.4) \end{aligned}$$

The baseband equivalent of the above signal is obtained by the receiver by using a sinusoid from the local oscillator at frequency f_c to obtain the time varying phasor $AH(t)e^{\theta(t)}$ for the channel estimation at time t. Therefore, we may represent the relationship between the received and transmit signals and the channel simply as

$$r = Ah + w$$

where A is the amplitude of the sinusoid transmitted at a phase of zero in the complex plane, $h \sim \mathcal{CN}(0, \sigma^2)$ is the complex Gaussian channel and $w \sim \mathcal{CN}(0, \sigma_n^2)$ is the complex Gaussian noise added at the receiver. The job of the channel estimator at the receiver is to estimate h. The channel estimate $\hat{h} = r/A = h + \frac{1}{A}w$ is the ML as well as the MAP estimate of h.

Now, let us assume that the transmitter is controlled by an adversary who modulates both the magnitude and the phase of the transmit signal, and instead of transmitting $\tilde{s}(t) = A\cos(2\pi f_c t)$, decides to transmit $\tilde{s}_E(t) = A(t)\cos(2\pi f_c t + \phi(t))$ with a complex representation $s_E(t) = A(t) \cdot e^{j\phi(t)}$. As a result the channel estimate made by the receiver is

$$\hat{h}(t) = \frac{A(t)}{A} h(t) e^{j\phi(t)} + \frac{1}{A} w$$

Ignoring the noise term, we find that the adversary has inserted a multiplicative factor of $A(t)e^{j\phi(t)}$ in front of the true channel state $h(t) = H(t)e^{\theta(t)}$. This causes the channel to appear to have magnitude of $A(t)H(t)$ and a phase of $\phi(t) + \theta(t)$. The question now arises whether the adversary can statistically influence the bits that are extracted by Alice and Bob.

Limitations. While the wireless channel can serve as a useful source of shared randomness, there are a few important limitations that are worth pointing out. The randomness inherent in the wireless channel arises from multipath propagation of RF waves. Waves arriving along a large number of separate paths add up with random phases at the receiver, resulting in a signal with random phase and magnitude. However, the amount of randomness in wireless channels is not always the same. For example, if there is a clear line of sight path between the transmitter and the receiver, then the channel behaves in a much less of a random manner because the line of sight path dominates over all the other paths arriving at the receiver. It is clear that different environments will provide different amount of randomness. This leads to variations in both, the temporal and the spatial decorrelation that wireless channels exhibit, depending on the precise environment. One example of a real world system that relies on spatial decorrelation of wireless channels is MIMO systems. If the channel were to not decorrelate sufficiently between successive antennas in a MIMO antenna array, then the system would not register gains from spatial diversity. Similarly, if the channel does not doecorrelate sufficiently quickly, then it cannot be guaranteed that an adversary more than $\lambda/2$ away from a legitimate user will experience a completely uncorrelated channel. What is needed therefore, is a mechanism to ascertain how quickly the channel decorerlates in space and in time, *before* channel measurements are committed as inputs to a security protocol. The rate of decorrelation in time can be measured by a receiver fairly directly, by computing an estimate of the coherence time, which is empirically related to the level crossing

rate of the channel, and we incorporate this into our key extraction algorithms. However, estimating the rate of spatial decorrelation at a legitimate user is far more difficult – we do not attempt to address this problem in this thesis.

We begin by observing that the product of amplitudes $A(t)H(t)$ behaves very differently from the sum of the phases $\theta(t) + \phi(t)$. In particular, while the adversary can reduce the dynamic range of the amplitude by using a smaller transmit amplitude $A(t)$, this can affect both, the amount of entropy in the channel estimates, as well as the signal to noise ratio. On the other hand, the phase cannot be controlled by the adversary. If each legitimate user samples the phase once per coherence time[2], the resulting phases samples are both correlated at the two users, as well as independent from one sample to the next, irrespective of the function $\phi(t)$. We conclude that by using differential phase, it is possible to enforce a fairly strong result, namely, that even when the adversary controls the transmitter, then assuming that the environment is richly scattering, the output of the channel remains random and cannot be influenced by the adversary.

In the following two sections, we will study a special way in which the channel can be used as a source of randomness, namely, as a source of *common* randomness, which brings in an additional requirement apart from the three above - namely, that the channel must be able to provide random a random output as before, but two (or more) terminals must have access to statistically correlated versions of the channel's random output. Section III deals with the case of spatially separated terminals that exploit the *reciprocity* inherent in RF wireless channels as a means to establish common randomness, Section IV explores the use of public sources of RF waves to establish common randomness at terminals that are in spatial proximity.

---

[2]Here, it is important that the estimate of a coherence time of the channel be based on an a priori understanding of what the typical coherence time should be under the given conditions, rather than based on the actual channel estimates. Otherwise, the attacker can employ a rapidly varying $A(t), \phi(t)$ to fool the users into severely underestimating the coherence time.

Chapter 3

Key extraction using reciprocal wireless channels

In this chapter, we will explore how two wireless terminals can exploit reciprocal fading wireless channels to establish a shared secret key. We will discuss the systems need for such a provision for shared common randomness and we will discuss some ways of incorporating our methods into existing wireless systems.

3.1 Introduction

Many of the risks associated with securing wireless systems stem from challenges associated with operating in a mobile environment, such as the lack of a guaranteed infrastructure or the ease with which entities can eavesdrop on communications. Traditional network security mechanisms rely upon cryptographic keys to support confidentiality and authentication services. However, in a dynamic mobile wireless environment, with peer-to-peer associations being formed on-the-fly between mobile entities, it is difficult to ensure availability of a certificate authority or a key management center. Since such scenarios are likely to become more prevalent, it is necessary to have alternatives for establishing keys between wireless peers without resorting to a fixed infrastructure.

We explore an alternative for building cryptographic services by exploiting an untapped resource – the wireless channel itself. The specificity of the radio channel between two wireless devices, and its rapid decorrelation with distance, provide a basis for the creation of shared secret information, such as cryptographic

keys, even in the presence of an eavesdropper. In typical multipath environments (see Figure 3.1), the wireless channel between two users, Alice and Bob, produces a time-varying, stochastic mapping between the transmitted and received signals. This mapping varies with time in a manner that is location-specific and reciprocal, i.e., the mapping is the same whether Alice is the transmitter with Bob as the receiver or vice-versa. The time-varying mapping, commonly termed *fading*, decorrelates over distances of the order of half a wavelength, λ. Thus, an adversary, Eve, who is more than $\lambda/2$ away from both Alice and Bob, experiences fading channels to Alice and to Bob that are statistically independent of the fading between Alice and Bob. These properties allow us to generate a common, secret cryptographic key at Alice and Bob such that Eve gets no information about the generated key. For example, at 2.4 GHz, we only require that Eve be roughly $\lambda/2 = 6.25$ cm away from Alice and Bob to ensure that she gets no useful information. Thus, while fading is typically considered harmful, we profitably exploit it to extract perfectly secret bits without leaking information to an adversary.

The extraction of secret bits from the wireless channel can be viewed as a 'black-box' that can be advantageous in various ways, putting to good use information that is already available from the channel. For example, in the current 802.11i standard, session keys for communication between a station and an AP are derived by hashing together authentication credentials and nonces exchanged in the clear. This ties the confidentiality of future messages to the authentication credentials, and if these credentials are ever compromised then an adversary will be able to derive the session keys and decrypt past encrypted messages. If the nonces can be derived in an information-theoretically secret manner from the channel between two users, then a passive adversary has no means to derive the session keys even if it learns the authentication credentials[18]. Further, session keys can be updated using these secret bits derived from the channel, instead of

relying on previously existing keys [18], thus ensuring that the confidentiality of each new session is protected independently of earlier sessions.

Yet another vulnerability in 802.11i stems from the fact that during the establishment of a secure link between a station and an AP, all messages exchanged over the air, including management frames, are sent unencrypted until both parties have obtained the session key (c.f. the *temporal key* (TK) in 802.11) and are therefore susceptible to eavesdropping and to spoofing by other users. While the 802.11w amendment seeks to protect some management frames from such attacks, it too fails to protect messages exchanged before the the establishment of TKs. Unfortunately, securing the initial exchanges between the parties requires them to share a key that is not established until later. Our key extraction mechanism provides a natural solution by allowing the parties to generate a temporary key that protects the interim exchanges before the formal keys are in place.

Ad hoc or peer-to-peer networks present another avenue where our technique can be useful. Alice may not care to establish Bob's identity if she merely wishes to employ his forwarding services. In such a scenario, she may nevertheless wish to establish a confidential link with Bob by using the channel to form a key prior to encrypting subsequent data, thereby preventing eavesdropping.

Prior work in information theory has noted the potential of using the wireless channel for generating shared secret bits, but most of this work has been aimed at computing theoretical limits and has not provided practical algorithms, nor a demonstrable and quantifiable impact on security. In this chapter,

1. We translate prior information-theoretic ideas into a practical protocol applied to wireless channels;

2. We build a new algorithm for key extraction that, unlike the schemes in prior literature, does not require an authenticated channel, and study performance for typical fading;

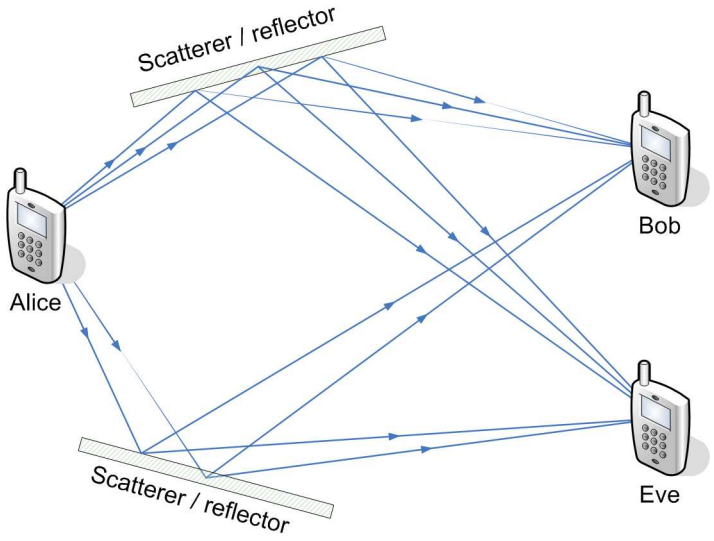


Figure 3.1: The multipath fading for a signal from Alice to Bob is different from that for the signal reaching Eve.

3. We validate our algorithm using channel impulse responses measured using the 802.11a packet preamble on a customized FPGA-based 802.11 development platform and a second study that uses only coarse per-packet RSSI information readily available to off-the-shelf 802.11 platforms.

Existing mobile radio platforms already provide the information we need, but such data are normally discarded after physical layer processing and can be profitably exploited to benefit security. The approach we present augments, rather than replaces existing cryptographic security mechanisms– it provides a new approach to establishing keys that is useful when there is no key management infrastructure.

3.2 Related work

Information-theoretic literature has explored the use of information from the physical layer in deriving security benefits. In [19, 20], the authors introduced the problem of generating identical bits based on correlated information available to two users such that a third eavesdropping user does not learn anything about the

generated key. They showed, provided Alice and Bob already share an authenticated public channel, that it is possible to generate identical keys at the two users. The standard method for generating secret keys at Alice and Bob under this assumption consists of three basic steps and has been utilized by a number of proposed systems [21, 22, 23]. In *advantage distillation* [19, 24], the legitimate users, Alice and Bob, obtain correlated information while Eve is allowed to eavesdrop, so that Alice & Bob share greater information[1] than that shared between Alice & Eve or Bob & Eve. Alice and Bob then convert their information into bits. In the *information reconciliation* stage [22], Alice and Bob exchange error-correcting messages over an authenticated public channel that allow them to agree on an identical string of bits. However, the publicly exchanged messages reveal a certain amount of information about the bit strings to Eve. In *privacy amplification* [26], Alice and Bob diminish the partial information revealed to Eve by systematically discarding some of their common bits. Efficient protocols have since been designed [22, 27][2] to allow key generation without leaking information to an eavesdropping adversary.

A central assumption in this entire body of work is that Alice and Bob have an *authenticated channel available to them* even before key generation begins. This is an unrealistic assumption in practice because the availability of an authenticated channel implies that Alice and Bob already share a secret key to begin with! Therefore, the purpose of generating a common secret key is defeated.

In [29], Maurer and Wolf showed that secret key extraction without an authenticated channel is possible only if Eve cannot possibly transmit a signal to Bob that is statistically indistinguishable from signals coming from Alice (and vice-versa). This provides an important insight that has not been translated into

[1]The amount of information between two observations X and Y is measured by the *mutual information* $I(X;Y)$[25].

[2]Much of this work was done in the context of quantum key distribution[28].

a practical algorithm. Our work is the first to build upon this result: we use the wireless channel to guarantee that Eve does not possess the required information to prevent key generation.

More recently, [30] examined PHY-layer based authentication and confidentiality in wireless systems. The work in [31, 32] looked at authentication using channel signatures between the transmitter and receiver(s). Our work is perhaps most closely related to[33], which proposes a scheme for generating secret bits from correlated observations of deep fades by two users communicating via a TDD link. This work focuses on the theoretical construction for extracting randomness through universal hash families. However, they do not demonstrate or evaluate the amenability of the wireless channel to detection of deep fades by both users, nor the precision needed in the TDD process for their scheme. A quantification of the secret key rate versus parameters associated with the underlying fading process or parameters involved in their algorithm was not provided. Additionally, we note that their method focuses primarily on a passive adversary. The reliance on deep fades may be exploited by an active adversary that produces greater interference power at one legitimate user than the other so that a deep fade for one user may not be a deep fade for the other. In [34], a method exploiting channel reciprocity using ultra-wideband (UWB) channels to generate secret bits was presented. In [35], specialized electronically steerable antennas were proposed for use in generating key bits by exploiting channel reciprocity. The methods in [34, 33, 35] all rely on conventional reconciliation for correcting bit-errors, and thus require an authenticated channel. In [36, 37], a method for secret key generation based on phase *reciprocity* of frequency selective fading channels was proposed. While this is attractive, it is difficult to implement as accurate phase information is hard to harvest from existing platforms.

In contrast to prior work, the algorithm we propose transcends the requirement of an authenticated channel, does not require specialized hardware and

is not limited to UWB channels. We provide a fundamental analysis between the performance of our scheme and underlying parameters governing fading and quantization. Further, we provide two real-world experimental implementations of our scheme and show that existing mobile platforms already provide sufficient information for producing secret bits. We evaluate the randomness of the bit-sequences produced by our algorithm, a generally overlooked aspect in prior work on secret key generation, and show that they are suitable for use as cryptographic keys. Lastly, we note that our technique may be compared with classical key establishment techniques such as Diffie-Hellman, which also use message exchanges to establish keys. However these rely upon unproven arguments of computational hardness of problems such as inverting discrete logarithms or factoring a product of large prime numbers. Our algorithm, on the other hand, provides information-theoretic secrecy, does not assume bounded computation power at the adversary and further, represents practical methods to achieve this type of security. The cost of enabling unconditional security must be borne out in some form – in our case this may take the form of collecting correlated information by probing – but in fact, depending upon how our method is used, much of the required information is already available in present day systems. In this way we provide a means to realize in wireless networks the same benefits that quantum cryptography has enabled using optical fiber links.

3.3 System model & Design issues

The crucial insight that allows the wireless channel to be amenable for generating a secret key is that the received signal at the receiver is modified by the channel in a manner that is unique to the transmitter-receiver pair. This distortion depends critically upon the location of the transmitter, the receiver, and scatterers. Typically, such distortion is estimated at the physical layer of the receiver

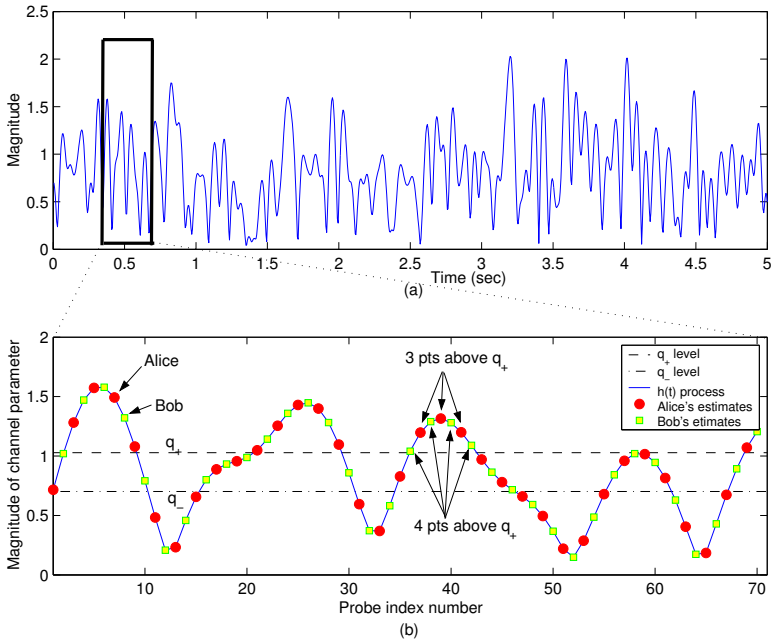


Figure 3.2: (a) A sample realization of a Rayleigh fading stochastic process. (b) Successive channel estimates of the process by Alice and Bob showing excursions above the q_+ and below the q_- levels on a magnified portion of (a).

and the associated distortion information dealt with for reliable physical layer decoding. Since this information is always present and uniquely corresponds to the transmitter-receiver pair, it also provides our transmitter (Alice) and receiver (Bob) a means to *privately* establish secret bits associated with this distortion. We now focus on the challenges of using the stochastic nature of the wireless channel to secretly establish bits. We break down our discussion to include a description of: (1) the underlying channel model associated with multipath fading; (2) the tools needed to obtain bits from the channel response; and (3) the design goals that need to be addressed in order to reliably establish these bits. To assist the reader, we provide notation in Table 3.1. Before we begin, we comment on our adversary. We assume an attacker that can either act as an eavesdropper or who may inject messages to impersonate Alice or Bob. We present further considerations of adversarial actions in Section 3.7.

Symbol	Meaning
$\mathbf{h}$	Stochastic channel parameter of interest
$h(t)$	Value of the stochastic process $\mathbf{h}$ at time t
$s(t)$	Probe signal transmitted to estimate $h(t)$
f_d	Maximum Doppler frequency (Hz)
f_s	Rate at which each user sends probes (Hz)
q_+, q_-	Quantizer bin boundaries (Upper and lower resp.)
m	Reqd. min. # of estimates in a excursion
N	Length of key in bits
R_k	Rate of generation of secret bits (s-bits/sec)
p_e	Probability of a bit error
p_k	Probability of key mismatch $= 1 - (1 - p_e)^N$

Table 3.1: A summary of the notation used

3.3.1 Channel model

Let $h(t)$ be a stochastic process corresponding to a time-varying parameter that describes the wireless channel between Alice and Bob. Although there are many choices for $h(t)$, for our discussion, we shall assume that $h(t)$ is the magnitude of the transfer function of the multipath fading channel between Alice and Bob evaluated at a fixed *test frequency*, f_0. Implicit in this formulation is the observation that the system transfer function of the channel is the same in the Alice $\rightarrow$Bob direction as in the Bob$\rightarrow$Alice direction *at a given instant of time*. This follows from reciprocity, which is a fundamental property of electromagnetic wave propagation [17, 38] in a medium and must not be confused with additive noise or interference, which may be different for different receivers. To distinguish between the channel parameter of interest, and its value at a given time, we denote the parameter by $\mathbf{h}$ and refer to its value as $h(t)$. To estimate the parameter $\mathbf{h}$, Alice and Bob must transmit known probe signals to one another. Each party can then use the received signal along with the probe signal to compute an estimate $\hat{h}$ of $\mathbf{h}$. Since practical radios are *half duplex* due to hardware constraints, Alice must wait to receive a probe signal from Bob before she can transmit a probe to him and vice-versa. In the time between the two successive probes, $h(t)$ changes

slightly in a manner that is modeled by an appropriate probability distribution. The received signal at Alice and Bob due to successive probes may be written as

$$r_a(t_1) = s(t_1)h(t_1) + n_a(t_1) \tag{3.1}$$

$$r_b(t_2) = s(t_2)h(t_2) + n_b(t_2), \tag{3.2}$$

where $s(t)$ is the known probe signal, n_a & n_b are the independent noise processes at Alice and Bob and t_1 & t_2 are the time instants at which successive probes are received by Alice and by Bob, respectively. Using the received signal, Alice and Bob, each compute (noisy) estimates of $\mathbf{h}$:

$$\hat{h}_a(t_1) = h(t_1) + z_a(t_1) \tag{3.3}$$

$$\hat{h}_b(t_2) = h(t_2) + z_b(t_2), \tag{3.4}$$

where z_a and z_b represent the noise terms due to n_a and n_b after processing by the function that estimates $\mathbf{h}$. We refer the reader to [39] for designing good estimators for $\mathbf{h}$. The estimates $\hat{h}_a$ and $\hat{h}_b$ are in all likelihood unequal, due in part to the independent noise terms and in part to the time lag τ. However they can be highly correlated if Alice and Bob send probes to one another at a fast enough[3] rate, i.e. if $\tau = t_2 - t_1$ is small. By repeatedly sending probes in an alternating manner over the time-varying channel, Alice and Bob can generate a sequence of n estimates $\underline{\hat{h}}_a = \{\hat{h}_a[1], \hat{h}_a[2], \ldots, \hat{h}_a[n]\}$ and $\underline{\hat{h}}_b = \{\hat{h}_b[1], \hat{h}_b[2], \ldots, \hat{h}_b[n]\}$, respectively, that are highly correlated, as in Figure 3.2. Although Eve can overhear the probe signals sent by each user, the signals received by Eve are completely different:

$$r_e^b(t_1) = s(t_1)h_{be}(t_1) + n_e(t_1) \tag{3.5}$$

$$r_e^a(t_2) = s(t_2)h_{ae}(t_2) + n_e(t_2), \tag{3.6}$$

[3]'Fast enough' here is in relation to the coherence time of the channel, which is inversely proportional to the maximum Doppler frequency f_d.

where h_{be} and h_{ae} denote the channel between Bob & Eve and between Alice & Eve, respectively, and n_e is the noise added at Eve. If Eve is more than $\sim \lambda/2$ away from Alice and Bob, then h_{ae} and h_{be} are uncorrelated with $\mathbf{h}$ [40]. Therefore, despite possessing knowledge of the probe signal $s(t)$, Eve cannot use her received signals to compute meaningful estimates of the Alice-Bob channel, $\mathbf{h}$.

3.3.2 Converting the channel to bits

Alice and Bob must translate their respective sequences of channel estimates into identical bit-strings suitable for use as cryptographic keys, thus requiring:

1. *Suitably long:* Keys of length 128 to 512 bits are commonly used in symmetric encryption algorithms. So they should be able to generate at least these many bits in a reasonable amount of time.

2. *Statistically random:* The bits should be random with equal probability of a '0' and a '1'. Also, the bit-sequences must not suffer from statistical defects that could be exploited by an attacker.

The second requirement guarantees that the generated key has desirable security properties. That is, an N-bit key must provide N bits of uncertainty to an adversary who only knows the key generation algorithm and nothing else.

We now briefly describe how to obtain bits from the channel estimates $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$, to provide the intuition behind our algorithm, while postponing a formal description to Section 4. The sequence of channel estimates $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$ are random variables drawn from an underlying probability distribution that characterizes the channel parameter $\mathbf{h}$. We assume, for the sake of discussion, that $h(t)$ is a Gaussian random variable and the underlying stochastic process $\mathbf{h}$ is a stationary Gaussian process. A Gaussian distribution for $\mathbf{h}$ may be obtained, for example, by taking $\mathbf{h}$ to be the magnitude of the in-phase component of a Rayleigh fading

process between Alice and Bob [17]. We note that the assumption of a Gaussian distribution on $\mathbf{h}$ is for ease of discussion and our algorithm is equally valid in the general case.

Since the channel estimates computed by Alice and Bob are continuous random variables, it is necessary to quantize their estimates using a quantizer $Q(\cdot)$ to obtain bits. However, a straightforward quantization of the vectors $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$ is not sufficient because it does not guarantee that an identical sequence of bits will be generated at the two users. In our scheme, Alice and Bob use the channel statistics to determine scalars, q_+ and q_- that serve as reference levels for the quantizer $Q(\cdot)$ as follows:

$$Q(x) = \begin{cases} 1 & \text{if } x > q_+ \\ 0 & \text{if } x < q_- \end{cases}. \tag{3.7}$$

Alice parses through her channel estimates $\underline{\hat{h}}_a$ to determine the locations of *excursions* of her channel estimates above q_+ or below q_- that are of a duration $\geq m$ estimates, i.e., m successive channel estimates in $\underline{\hat{h}}_a$ are $> q_+$ or $< q_-$, where m is a protocol parameter. She sends Bob a message over the public channel containing the locations of k such excursions in the form of an array of indexes $L = \{l_1, l_2, \ldots, l_k\}$. Bob then checks his own sequence $\underline{\hat{h}}_b$ at the locations specified in L to determine whether it contains an excursion above q_+ or below q_- for a duration greater than or equal to '$m-1$' samples, i.e. whether $\underline{\hat{h}}_a(l_i)$ is $> q_+$ or $< q_-$ for a duration that spans $m-1$ or more estimates, for $i = 1, \ldots, k$. Bob identifies 'good' indexes by finding all index values l in L that produce such an excursion in $\underline{\hat{h}}_b$. He places these indexes into an array $\tilde{L}$ to be sent to Alice publicly. Indexes in L but not in $\tilde{L}$ are dropped from consideration by each party. The indexes in $\tilde{L}$ are used by each user to compute a sequence of bits by quantizing: $Q(\underline{\hat{h}}_a(\tilde{L}))$ and $Q(\underline{\hat{h}}_b(\tilde{L}))$. If the bit-vectors $Q(\underline{\hat{h}}_a(\tilde{L}))$ and $Q(\underline{\hat{h}}_b(\tilde{L}))$ are equal, then Alice and Bob succeed in generating $|\tilde{L}|$ identical bits. We show later that provided the levels q_+, q_- and the parameter m are properly chosen, the bits generated by the

two users are identicall with very high probability. A variation of the protocol that copes with spoofing is detailed in Section 3.4.1.

3.3.3 Design goals

An important quantity of interest will be the rate of generation of secret bits, expressed in secret-bits per second or 's-bits/sec'. Naturally, it is desirable that Alice and Bob achieve a high secret-bit rate. According to 802.1x recommendations, it is generally desirable for master keys to be refreshed at one hour intervals[41]. Using these examples and AES key sizes of 128 bits as a guideline, a conservative key rate of roughly 0.1 bits per second is needed, though it is desirable to achieve higher secrecy rates. However, we are especially wary of bit errors. If the sequence $Q(\underline{\hat{h}}_a(\tilde{L}))$ is different from $Q(\underline{\hat{h}}_b(\tilde{L}))$ even by a single bit, then the two bit-strings cannot be used as cryptographic keys and consequently the entire batch of bits must be discarded. Therefore, we would like the bit error probability p_e to be extremely low, so that the probability p_k that the keys generated by the two users do not match is acceptably small. For example, in order to have a key-mismatch probability of $p_k = 10^{-6}$, assuming keys of length 128 bits, we must target a bit-error probability of p_e where

$$p_k = 1 - (1 - p_e)^{128}, \tag{3.8}$$

which gives $p_e \sim 10^{-8}$. A bit-error is defined as the event that Alice and Bob agree to use a certain index l_i contained in the list $\tilde{L}$ for generating a bit, but they end up generating different bits, i.e. $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$ both lie in excursions at the index l_i but the excursions are of opposite types.

The rate at which secret bits can be extracted from the channel is fundamentally limited by the rate of time-variation in the channel. We quantify this variation by the *maximum Doppler frequency*, f_d. In a fading channel, f_d determines both the rate at which the channel varies and the magnitude of the swings

produced. A simple measure of the maximum Doppler frequency in a given wireless environment is given by $f_d = \frac{v}{\lambda}$, where v is a measure of the effects of user mobility and the dynamic environment around the users, expressed in meters/sec and λ is the wavelength of the carrier wave. In our case $\lambda = \frac{c}{f_0}$, where c is the speed of light. It can be seen that increasing the value m or the magnitudes of the quantizer boundaries q_+ & q_- would not only result in a lower rate, but also a lower probability of error. Intuitively, this is because larger magnitudes of q_+ & q_-, or a larger value of m makes it less likely that Alice's and Bob's channel estimates lie in opposite type of excursions, thereby reducing the error rate. However, both types of excursions also become less frequent, thereby decreasing the number of secret bits that can be generated per second. Thus, there is a tradeoff between rate and probability of error, and the parameters q_+, q_- and m provide convenient controls to select suitable operating points over this tradeoff. Beyond rate and robustness, we also require the bits to be random and free from statistical defects, as discussed in Section 3.5.3.

Finally, the correlated information obtained by Alice and Bob can be utilized to build a secret key in a number of different ways and it is important to make sure the method employed does not allow Eve to infer any useful information. An alternative bit extraction scheme is to have each user estimate a statistical measure of the channel (e.g. the mean signal-strength, or variance in the estimates) using $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$ respectively. If the channel is stochastically stationary, then their respective statistical measures would each converge to the true value with time. In this way, Alice and Bob will each possess knowledge about a numerical quantity, without having sent messages over the air containing this quantity. They could then quantize their estimates of the statistical measure to generate bits. However, the trouble with using a statistical measure is that knowledge of the locations of Alice and Bob and their environment may allow Eve to infer the statistics of the channel between them. Indeed, publicly available tools, such as the WISE

ray-tracer [42], make it easy to predict the signal statistics at a receiver given the knowledge of the locations of the transmitter and receiver and the building's layout. Thus, it is important to recognize that using a statistical measure for key generation can be perilous. Our algorithm avoids statistical measures by relying on specific instantiations of the fading process.

3.4 Level-crossing Algorithm

We now detail our level-crossing based key-extraction algorithm. It is assumed that when the algorithm is run, Alice and Bob have collected a sufficiently large number of channel estimates $\underline{\hat{h}_a}$ and $\underline{\hat{h}_b}$, by alternately probing the channel between themselves. Further, it is assumed that the vectors $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$ are of equal length and their j^{th} elements $\hat{h}_a(j)$ and $\hat{h}_b(j)$ correspond to successive probes sent by Bob and Alice respectively, for each $j = 1, \ldots, length(\underline{\hat{h}}_a)$. Algorithm 1 describes the procedure and consists of the following steps:

1. Alice parses the vector $\underline{\hat{h}}_a$ containing her channel estimates to find instances where m or more successive estimates lie in an excursion above q_+ or below q_-.

2. Alice selects a random subset of the excursions found in step 1 and for each selected excursion, she sends Bob the index of the channel estimate lying in the center of the excursion, as a list L. Therefore, if $\underline{\hat{h}}_a(i) > q_+$ or $< q_-$ for some $i = i_{start}, \ldots, i_{end}$, then she sends Bob the index $i_{center} = \lfloor \frac{i_{start}+i_{end}}{2} \rfloor$.

3. For each index from Alice, Bob checks whether his vector of estimates $\underline{\hat{h}}_b$ contains *at least* $m-1$ channel estimates centered around that index in an excursion above q_+ or below q_-, i.e. whether $\underline{\hat{h}}_a > q_+$ or $< q_-$ for each index $\left\{l - \lfloor \frac{m-2}{2} \rfloor, \ldots, l + \lceil \frac{m-2}{2} \rceil \right\}$, for each $l \in L$.

4. For some of the indexes in L, Bob's channel estimates do not lie in either

excursion. Bob makes a list $\tilde{L}$ of all indexes that lie in excursions and sends it to Alice.

5. Bob and Alice compute $Q(\underline{\hat{h}}_a)$ and $Q(\underline{\hat{h}}_b)$ respectively at each index in $\tilde{L}$, thus generating a sequence of bits.

Algorithm 1: The basic level crossing algorithm

Input : $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$

Output: A cryptographic key $K_a = K_b$ at Alice and Bob

Alice:

for i = *1 to length*$(\underline{\hat{h}}_a) - m$ **do**
  if $Q(\hat{h}_a[i]) = Q(\hat{h}_a[i+1]) \ldots = Q(\hat{h}_a[i+m-1])$ **then**
    $i_{end} \leftarrow$ last index in excursion
    $L' \leftarrow [L' \;\; ; \lfloor \frac{i+i_{end}}{2} \rfloor]$
    $i \leftarrow i_{end} + 1$
  else
    $i \leftarrow i + 1$
  end
end

L = Random subset of L'

Alice sends L to Bob on `PUBLIC_CHANNEL` .

Bob:

for $l \in L$ **do**
  if $Q\left(\hat{h}_b(l - \lfloor \frac{m-2}{2} \rfloor)\right) = \ldots = Q\left(\hat{h}_b(l + \lceil \frac{m-2}{2} \rceil)\right)$ **then**
    $\tilde{L} \leftarrow [\tilde{L}; \;\; l]$
  end
end

$K_b = Q\left(\hat{h}_b(\tilde{L})\right)$

Bob sends $\tilde{L}$ to Alice on `PUBLIC_CHANNEL`.

Alice:

$K_a = Q(\hat{h}_a(\tilde{L}))$

Since Eve's observations from the channel probing do not provide her with any useful information about $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$, the messages L and $\tilde{L}$ do not provide her any useful information either. This is because they contain time indexes only whereas the generated bits depend upon the values of the channel estimates at those indexes. Further, the selection of a random subset in Step 2 from the set

of eligible excursions found in Step 1, guarantees that Eve cannot use L and $\tilde{L}$ to infer the values of the channel estimates of Alice or Bob at those time indexes.

3.4.1 Preventing a Spoofing Attack

Since Alice and Bob do not share an authenticated channel, Eve can impersonate Alice in Step 2, or Bob in Step 4 above. Such an attack would allow Eve to insert her own 'fake' L or $\tilde{L}$ messages, thus spoofing a legitimate user and disrupting the protocol without revealing her presence. Therefore we require a form of *data-origin authentication*, that assures each user that the L or $\tilde{L}$ message has originated at the legitimate transmitter.

Our protocol can be made to detect the adversary in each of the two cases above. We first focus on Eve inserting a fake L-message. Since Eve has no information about the locations of channel excursions apart from L, she can only make random guesses about which indexes to place into a fake L-message to Bob (apart from the ones Eve learns from L). If Eve inserts a significant number of random guesses into a fake L-message, Bob can detect her presence by computing the proportion of indexes in L that lead to excursions in $\underline{\hat{h}}_b$. Since Eve can only make random guesses, this quantity would be much lower than one resulting from a legitimate L-message from Alice. For each guess, she has a very low probability of choosing an index that lies in an excursion spanning $(m-1)$ or more estimates at Bob. Of these, the indexes that do not lie in an excursion in $\underline{\hat{h}}_b$ are discarded by Bob while those that do happen to lie in an excursion are considered eligible for quantization and placed into the $\tilde{L}$-message sent to Alice. Thus, an unsuccessful guess provides no benefit to Eve, while a successful guess, albeit improbable, causes $\tilde{L}$ to contain an index that was not present in L, thereby alerting Alice. Thus, Eve must also modify the message $\tilde{L}$ by deleting this index before it reaches

Algorithm 2: Modified algorithm incorporating data-origin authentication and resistance to an active attack.

Input : $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$

Output: A cryptographic key $\bar{K}_a = \bar{K}_b$ at Alice and Bob

Alice:

for i = *1 to length*$(\underline{\hat{h}}_a) - m$ **do**
  if $Q(\hat{h}_a[i]) = Q(\hat{h}_a[i+1]) = \ldots = Q(\hat{h}_a[i+m-1])$ **then**
    $i_{end} \leftarrow$ last index in excursion
    $L' \leftarrow [L' \;\; ; \lfloor \frac{i+i_{end}}{2} \rfloor]$
    $i \leftarrow i_{end} + 1$
  else
    $i \leftarrow i + 1$
  end
end
L = Random subset of L'
Alice sends L to Bob on `PUBLIC_CHANNEL`.

Bob:

for $l \in L$ **do**
  if $Q\left(\hat{h}_b(l - \lfloor \frac{m-2}{2} \rfloor)\right) = \ldots = Q\left(\hat{h}_b(l + \lceil \frac{m-2}{2} \rceil)\right)$ **then**
    $\tilde{L} \leftarrow [\tilde{L}; \;\; l]$
  end
end
if $\left\{\frac{|\tilde{L}|}{|L|} < 0.5 + \epsilon\right\}$ **then**
  DECLARE ACTIVE ATTACK
else
  $K_b = Q\left(\hat{h}_b(\tilde{L})\right)$
  $K_{au} = K_b(1, \ldots, N_{au})$
  $\bar{K}_b = K_b(N_{au}+1, \ldots, N)$
  $Package = \left\{\tilde{L}, MAC\left(K_{au}, \tilde{L}\right)\right\}$
  Bob sends *Package* to Alice on `PUBLIC_CHANNEL`.
end

Alice:

$K_a = Q\left(\hat{h}_a(\tilde{L})\right)$
$K_{au} = K_a(1, \ldots, N_{au})$
$\bar{K}_a = K_a(N_{au}+1, \ldots, N)$
if *MAC validation using* K_{au} *fails* **then**
  DECLARE ACTIVE ATTACK
end

Alice. Our protocol can be made to resist modification of the $\tilde{L}$-message using a *message authentication code* (MAC), by the following *additional* steps:

1. To make sure the L-message received is from Alice, Bob computes the fraction of indexes in L where $\underline{\hat{h}}_b$ lies in an excursion spanning $(m-1)$ or more estimates. If this fraction is less than $\frac{1}{2}+\epsilon$, for some fixed parameter $0<\epsilon<\frac{1}{2}$, Bob concludes that the message was not sent by Alice, implying an adversary has injected a fake L-message.

2. If the check above passes, Bob replies to Alice with a message $\tilde{L}$ containing those indexes in L at which $\underline{\hat{h}}_b$ lies in an excursion. Bob computes $K_b = Q(\underline{\hat{h}}_b(\tilde{L}))$ to obtain N bits. The first N_{au} bits are used as an authentication key to compute a message authentication code (MAC) of $\tilde{L}$. The remaining $N-N_{au}$ bits are kept as the extracted secret key. The overall message sent by Bob is $\left\{\tilde{L}, MAC\left(K_{au}, \tilde{L}\right)\right\}$.

Upon receiving this message from Bob, Alice uses $\tilde{L}$ to form the sequence of bits $K_a = Q(\hat{h}_a(\tilde{L}))$. She uses the first N_{au} bits of K_a as the authentication key $K_{au} = K_a(1,\ldots,N_{au})$, and using K_{au} she verifies the MAC to confirm that the package was indeed sent by Bob. Since Eve does not know the bits in K_{au} generated by Bob, she cannot modify the $\tilde{L}$-message without failing the MAC verification at Alice.

Even without an authenticated channel, Alice and Bob can successfully establish a common secret key despite an active adversary, provided there are no bit errors. This explains why we insist on a very low probability of error in Section 3.3.3. Further, the reduction in the secret-bit rate is negligible over a long run of the protocol because the N_{au} bits are a one-time expense that allow Alice and Bob to bootstrap data-origin authentication. A modified algorithm that incorporates the above ideas is presented as Algorithm 2 (see Figure 3.3).

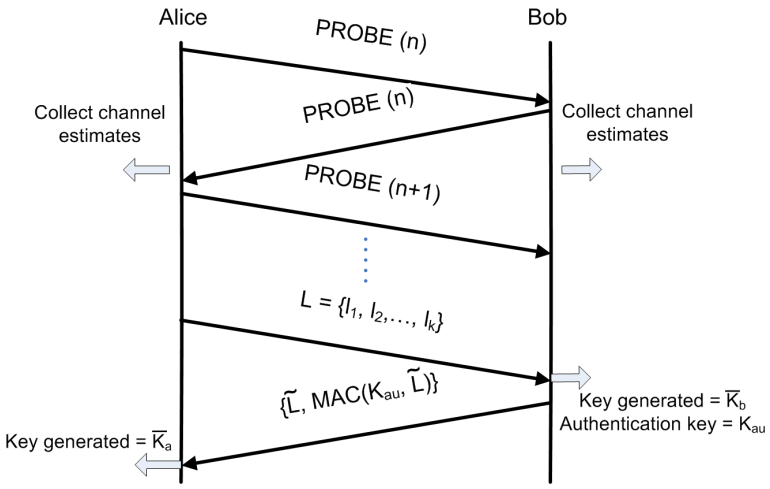


Figure 3.3: Timing diagram for the key-extraction protocol.

Another active attack involves Eve impersonating Alice or Bob during the channel-probing stage, i.e. Eve may begin sending probes to Bob pretending to be Alice or vice-versa. Such an attack can be detected using a hypothesis testing approach on the recent history probes received at each legitimate user, and this has been extensively studied in [31, 32]. The technique relies on the insight that given a sufficiently fast probing rate, successive probes received by a user are most likely to differ by a small amount. We provide further discussion related to the security of our scheme in Section 3.7.

3.5 Performance evaluation

The central quantities of interest in our protocol are the rate of generation of secret bits, the probability of error and the randomness of the generated bits. The controls available to us are the parameters: q_+, q_-, m and the rate at which Alice and Bob probe the channel between themselves, f_s. We assume the channel is not under our control, and as explained in Section 3.3.3, the rate at which the channel varies can be represented by the maximum Doppler frequency, f_d. The typical Doppler frequency for indoor wireless environments at the carrier frequency of 2.4 GHz is $f_d = \frac{v}{\lambda} \sim \frac{2.4\times10^9}{3\times10^8} = 8$ Hz, assuming a velocity v of 1

m/s. We thus expect typical Doppler frequencies in indoor environments in the 2.4 GHz range to be roughly 10 Hz and 20 Hz in the 5 GHz range. For automobile scenarios, we can expect a Doppler of $\sim$ 200 Hz in the 2.4 GHz range.

3.5.1 Probability of error

The probability of error, p_e is critical to our protocol. In order to achieve a robust key-mismatch probability p_k, the bit-error probability p_e must be much lower than p_k. A bit-error probability of $p_e = 10^{-7} \sim 10^{-8}$ is desirable for keys of length $N = 128$ bits. We have explained in Section 3.3.3 that there is a fundamental trade-off in the selection of parameters m, q_+ and q_- that affects the rate and probability of error in opposing ways.

The probability of bit-error, p_e is the probability that a single bit generated by Alice and Bob is different at the two users. The symmetry of the distribution of $\mathbf{h}$ allows us to consider just one type of bit error in computing p_e. Consider the probability that Bob generates the bit "0" at an index given that Alice has chosen this index but she has generated the bit "1". As per our Gaussian assumption on the parameter $\mathbf{h}$ and estimates $\hat{h}_a$ and $\hat{h}_b$, this probability can be expanded as

$$P(B=0|A=1) = \frac{P(B=0, A=1)}{p(A=1)} = \frac{\underbrace{\int_{q_+}^{\infty}\int_{-\infty}^{q_-}\cdots\int_{q_+}^{\infty}}_{(2m-1)\ terms} \frac{(2\pi)^{(1-2m)/2}}{|K_{2m-1}|^{1/2}} \exp\left\{-\frac{1}{2}x^T K_{2m-1}^{-1} x\right\} d^{(2m-1)}x}{\underbrace{\int_{q_+}^{\infty}\cdots\int_{q_+}^{\infty}}_{(m)\ terms} \frac{(2\pi)^{-m/2}}{|K_m|^{1/2}} \exp\left\{-\frac{1}{2}x^T K_m^{-1} x\right\} d^{(m)}x}, \tag{3.9}$$

where K_m is the covariance matrix of m successive Gaussian channel estimates of Alice and K_{2m-1} is the covariance matrix of the Gaussian vector

$$(\hat{h}_a[1], \hat{h}_b[1], \hat{h}_a[2], \ldots, \hat{h}_b[m-1], \hat{h}_a[m])$$

formed by the combining m channel estimates of Alice and the $m-1$ estimates

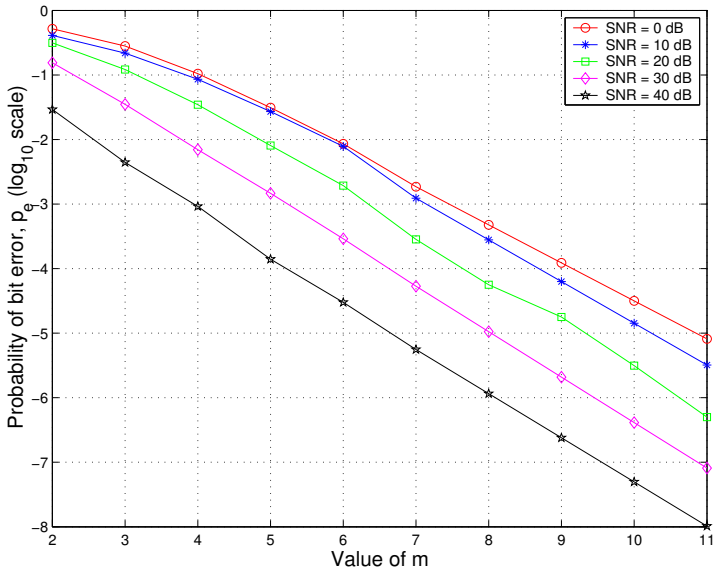


Figure 3.4: Probability of bit error p_e for various values of m at different SNR levels ($q_{\pm} = mean \pm 0.8\sigma$)

of Bob in chronological order. The numerator in (3.9) is the probability that of $2m-1$ successive channel estimates (m belonging to Alice, and $m-1$ for Bob), all m of Alice's estimates lie in an excursion above q_+ while all $m-1$ of Bob's estimates lie in an excursion below q_-. The denominator is simply the probability that all of Alice's m estimates lie in an excursion above q_+. We compute these probabilities for various values of m and present the results of the probability of error computations in Figure 3.4. The results confirm that a larger value of m will result in a lower probability of error, as a larger m makes it less likely that Alice's and Bob's estimates lie in opposite types of excursions. Note that if either user's estimates do not lie in an excursion at a given index, a bit error is avoided because that index is discarded by both users.

3.5.2 Secret-bit rate

The correct way to address the tradeoff between probability of error and rate of generation of secret bits is to upper bound the acceptable probability of error

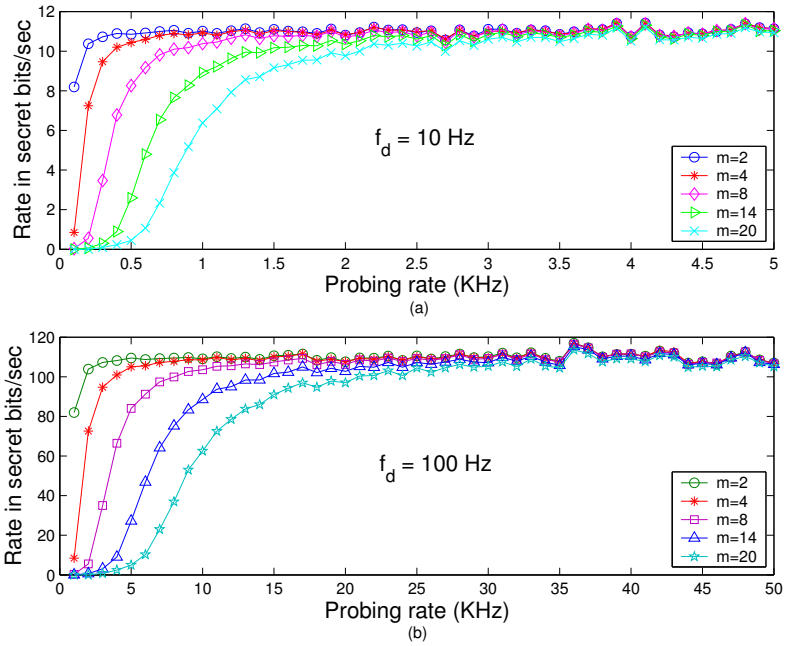


Figure 3.5: Rate in secret bits per second for various values of m, against probing rate for a channel with Doppler frequency (a) $f_d = 10$ Hz and (b) $f_d = 100$ Hz ($q_{\pm} = mean \pm 0.8\sigma$)

and then attempt to derive the greatest possible rate. How many s-bits/second can we expect to derive from a time-varying channel? An approximate analysis can be done using the level-crossing rate for a Rayleigh fading process, given by $LCR = \sqrt{2\pi} f_d \rho e^{-\rho^2}$ [17], where f_d is the maximum Doppler frequency and ρ is the threshold level, normalized to the root mean square signal level. Setting $\rho = 1$, gives $LCR \sim f_d$.

The above calculation tells us that we cannot expect to obtain more s-bits per second than the order of f_d. In practice, the rate of s-bits/sec depends also on the channel probing rate f_s, i.e. how fast Alice and Bob are able to send each other probe signals. In Figure 3.5 (a) and (b), we plot the rate in s-bits/sec as a function of the channel probing rate for a wireless channel with maximum Doppler frequencies of $f_d = 10$ Hz and $f_d = 100$ Hz respectively. As expected, the number of s-bits the channel yields increases with the probing rate, but saturates at a value on the order of f_d. More precisely, the number of s-bits/sec is the

number of s-bits per observation times the probing rate. Therefore

$$R_k = H(bins) \times p(A = B) \times \frac{f_s}{m} \tag{3.10}$$

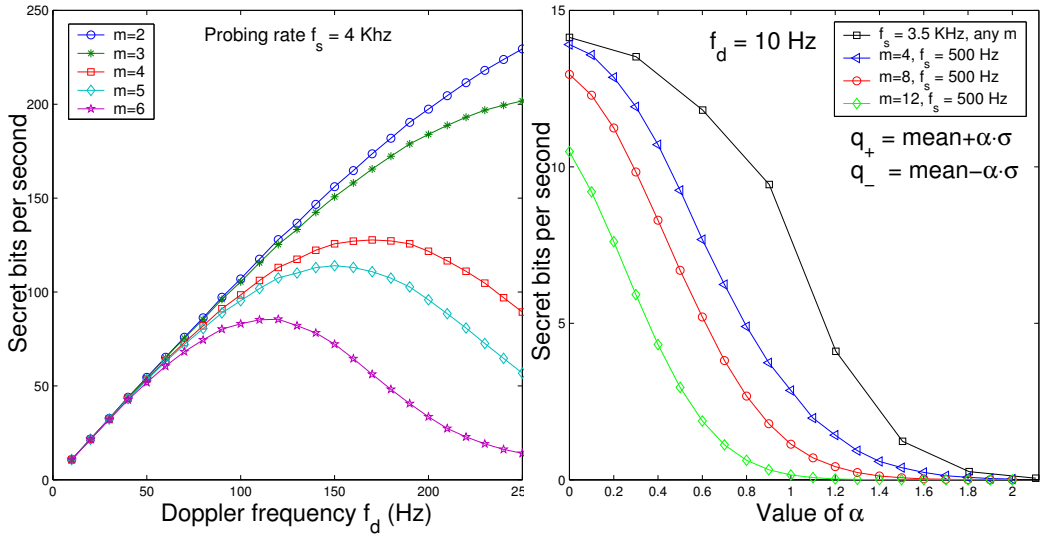
$$= 2\frac{f_s}{m} \times p(A = 1, B = 1) \tag{3.11}$$

$$= 2\frac{f_s}{m} \cdot \underbrace{\int_{q_+}^{\infty} \cdots \int_{q_+}^{\infty}}_{(2m-1)\ terms} \frac{(2\pi)^{\frac{1-2m}{2}}}{|K_{2m-1}|^{1/2}} e^{\left\{-\frac{1}{2}x^T K_{2m-1}^{-1} x\right\}} d^{2m-1}x, \tag{3.12}$$

where $H(bins)$ is the entropy of the random variable that determines which bin ($> q_+$ or $< q_-$) of the quantizer the observation lies in, which in our case equals 1 assuming that the two bins are equally likely[4]. The probing rate f_s is normalized by a factor of m because a single 'observation' in our algorithm is a sequence of m channel estimates. The expression in (3.12) is reminiscent of the probability of error expression in (3.9) and has been evaluated in Figure 3.5.

Figure 3.5 confirms the intuition that the secret bit rate must fall with increasing m, since the longer duration excursions required by a larger value of m are less frequent. In Figure 3.6 (a), we investigate how the secret-bit rate R_k varies with the maximum Doppler frequency f_d, i.e. versus the channel time-variation. We found that for a fixed channel probing rate (in this case, $f_s = 4000$ probes/sec), increasing f_d results in a greater rate but only up to a point, after which the secret-bit rate begins to fall. Thus, 'running faster' does not always help unless we can increase the probing rate f_s proportionally. This suggests that not only does each channel have an optimal minimum probing rate for deriving the best possible secret-bit rate, but each probing rate also corresponds to a most 'useful' maximum Doppler frequency. Figure 3.6(b) shows the expected decrease in rate as the quantizer levels q_+ and q_- are increased in magnitude. In this figure, α denotes the number of standard deviations from the mean at which the quantizer levels are placed.

[4]The levels q_+ and q_- are chosen so as to maintain equal probabilities for the two bins.



(a) (b)

Figure 3.6: (a) Secret-bit rate for varying Doppler f_d and fixed f_s for various values of m (b) Rate as a function of function of quantizer levels q_+ & q_- parametrized by α.

3.5.3 Randomness of generated bits

Guaranteeing that the generated bits are random is crucial because they are intended for use as a cryptographic key. Since we have assumed the adversary possesses complete knowledge of our algorithm, any non-random behavior in the bit sequences can be exploited by the adversary to reduce the time-complexity of cracking the key. For example, if the algorithm is known to produce a greater proportion of '1's than '0's, then the effective search space for the adversary would be reduced. Consequently, a variety of statistical tests have been devised to test for various defects [43].

In evaluating the randomness of bit sequences generated by our algorithm, we focus on Maurer's universal statistical test [44], a widely accepted benchmark for testing randomness. The test statistic relates closely to the per-bit entropy of the sequence, and thus measures the actual cryptographic significance of a defect as related to the running time of an adversary's optimal key-search strategy [44].

Test	**P-value**
Maurer's Test	0.8913
Monobit frequency	0.9910
Runs Test	0.1012
Approx. entropy	0.8721
Random excursions	0.5829
Lempel Ziv	1.0000

Table 3.2: Results from randomness tests on bit sequences (10^8 bits) produced by our algorithm for $f_d = 10$ Hz, $f_s = 30$ Hz, $m = 5$ and $q_+, q_- = mean \pm 0.2\sigma$. In each test, a p-value > 0.01 indicates the sequence is random.

Additionally, we ran a few other tests using the NIST public-domain test suite[45]. We refer the interested reader to [46] for a description of these tests and the definitions of $p - value$ for each test. The results for these are summarized in Table 3.2. Subsequent runs produced comparable results and thus support the conclusion that our algorithm provides random bits. In particular, Maurer's test showed the average entropy of our bit-sequences is very close to the value expected for a truly random sequence. This can be possible only if successive bits are almost independent, which in turn requires that they must be separated in time by at least a 'coherence time' interval. Since the coherence time of a channel is inversely proportional to the Doppler frequency, extracting bits from a channel at a rate significantly greater than f_d cannot possibly produce random bits. We observed in Section 3.5.2 that the rate at which our algorithm generates secret bits is bounded from above by approximately the maximum Doppler f_d. Therefore, our algorithm produces bits slow enough to be random. Finally, we note that the selection of a random subset of excursions by Alice effectively allows her some control on selecting the final key generated. Thus, even if a particular run happens to produce excursions at Alice containing a statistical defect in the resulting bit sequence, she can fix the defect to some extent by suitably choosing L from among eligible excursions.

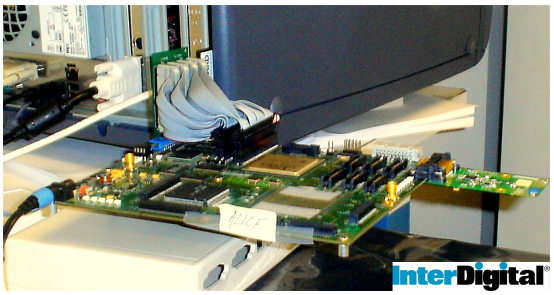
3.6 Experimental validation on 802.11a hardware

We now describe our experimental validation efforts for typical indoor environments. Our experiments were divided in two parts. In the first study, we delved into the structure of an 802.11 packet to access the preamble sequence [47] in the received signal to compute a 64-point *channel impulse response* (CIR) that showed one or more resolvable dominant paths as separate peaks. We used the magnitude of the tallest peak in the CIR (the dominant multipath) as the channel our parameter of interest. To access signal information at the sample level, we used an 802.11 development platform with FPGA-based customized logic added for processing CIR. Our results showed that our algorithm works very well for both static and mobile scenarios, producing error-free secret bits at rates ~ 1 s-bits/sec in the tested indoor environments.

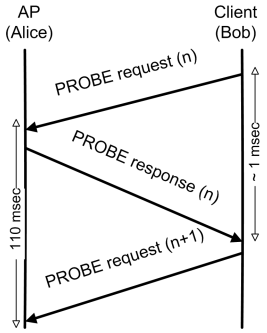
Encouraged by the CIR results, we sought to determine whether unmodified off-the-shelf 802.11 hardware could achieve comparable results. Therefore, for the second study, we used coarse RSSI measurements reported in the Prism headers of 802.11 packets exchanged between commercially available 802.11a radios, with Alice configured as an access point (`AP` mode) and Bob as a client (`station` mode), and a third user configured to listen (`station` mode) on transmissions from both legitimate users.

3.6.1 CIR method using 802.11a

Experiment setup: Our experimental platform (Figure 3.7(a)) consisted of an 802.11 development board with commercial 802.11a/b/g modem IP, to which we added custom logic to extract the channel impulse response from received packets. This allowed us to pull out received signal information at a level not normally accessible using commodity 802.11 hardware and drivers. Two such boards were set up as Alice and Bob, while a third board was configured to be Eve. Alice



(a)



(b)

Figure 3.7: (a) Our experimental platform - a development board for a commercial 802.11a/b/g modem IP, to which we added custom logic to process CIR information. (b) Timing diagram for collecting CIR information using PROBE packets

was configured to be an access point (`AP` mode), and Bob was configured to be a client (`station` mode). The experiment involved Bob sending `PROBE request` messages to Alice, who then replied with a `PROBE response` message (Figure 3.7(b)). Limitations of our development boards allowed us to have Eve listen on either Alice or Bob, but not both. In the results presented here, Eve has been configured to listen in on Alice. In the first experiment, Alice and Eve were placed in a laboratory, while Bob was placed in an office cubicle outside the lab, see Figure 3.8. In the second experiment, Alice and Eve remained in the same positions while Bob circled the cubicle area along the trajectory in Figure 3.8 in a cart on wheels.

Figure 3.9 shows a 64-point CIR obtained from a single 802.11a `PROBE request` packet received at Alice, along with the corresponding CIR computed from the `PROBE response` packet received by Bob in reply. Also shown is the CIR as computed by Eve, using the overheard `PROBE response` packet from Alice. For the purpose of our algorithm, we use only the magnitude of the main peak in the CIR.

Figure 3.10 shows the traces of the CIR's main peak's magnitude at Alice

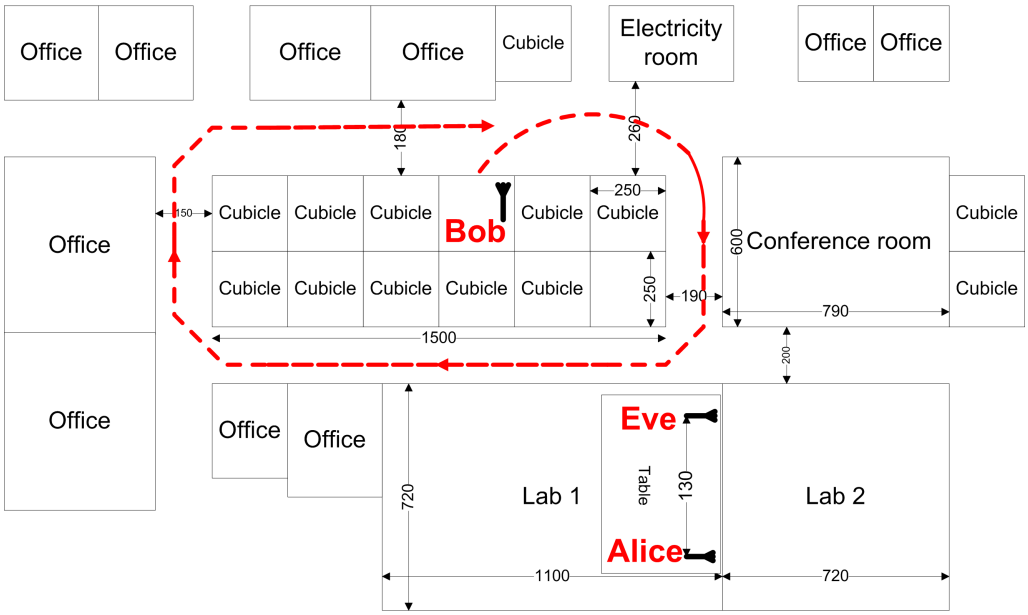


Figure 3.8: A layout of the experimental setup for the CIR method (distances in cm)

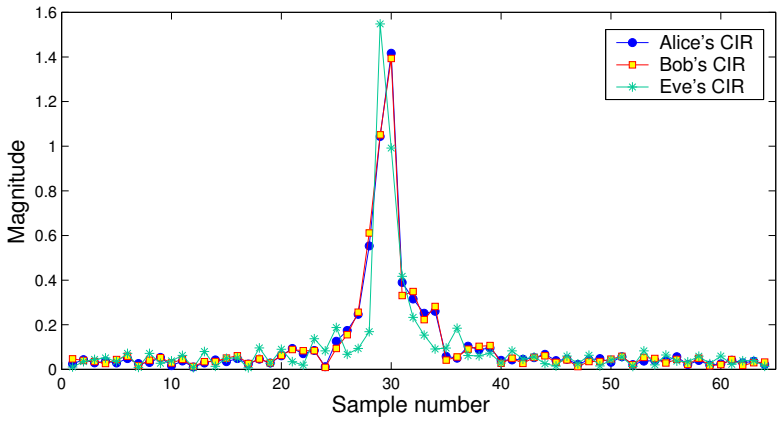


Figure 3.9: The 64-point CIR from a single 802.11 packet. For our key-extraction algorithm, we use the magnitude of the main peak as the channel parameter of interest.

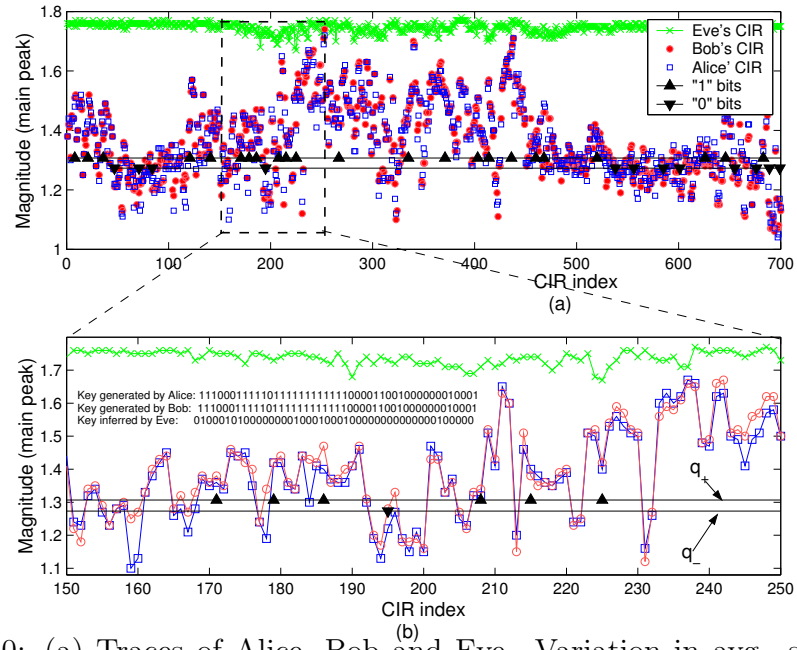


Figure 3.10: (a) Traces of Alice, Bob and Eve. Variation in avg. signal power produces longs strings of 1s and 0s. (b) A magnified portion of the traces.

and Bob for our first experiment. While our experiment ran for $\sim$ 22 minutes, in the interest of space and clarity we show 1000 CIRs collected over a duration of $\sim$ 110 seconds. The traces show significant changes in average signal power, ostensibly due to time-variations in the wireless environment between Alice and Bob (see Figure 3.8). If each user simply uses this data as input to the level-crossing bit-extraction algorithm, the generated key has long strings of 1s and 0s (see Figure 3.10). This is because we are attempting to include the effect of *shadow fading* [17] (also called large-scale fading) that produces large but slow swings in the average signal power into the key generation algorithm. In other words, the channel in Figure 3.10 is not stationary. Each user locally computes q_+ and q_- as:

$$q_+^u = mean(\underline{\hat{h}}_u) + \alpha \cdot \sigma(\underline{\hat{h}}_u) \tag{3.13}$$

$$q_-^u = mean(\underline{\hat{h}}_u) - \alpha \cdot \sigma(\underline{\hat{h}}_u), \tag{3.14}$$

where u can be Alice or Bob, $\underline{\hat{h}}_u$ is the set of magnitudes of the CIR's main peak collected by user u, and $\sigma(\underline{\hat{h}}_u)$ represents the standard deviation of $\underline{\hat{h}}_u$. The

factor α can be selected to vary the quantizer levels. We chose $\alpha = \frac{1}{8}$ for the CIR-method. The effect of the underlying shadow fading contained in the collected data can be removed by subtracting a moving average of each trace from the original trace. This leaves only the small scale fading that we wish to use in our algorithm. The result is shown in Figure 3.11. In this way, not only do we do away with the problem of long strings of 1s and 0s, we also prevent the average signal power from affecting our key generation process. Using the small scale fading traces, our algorithm generates $N = 125$ s-bits in 110 seconds ($m = 4$), yielding a key rate of about 1.13 s-bits/sec.

Contrasting Eve's attempts: Figures 3.10 shows a trace of Eve's CIR peak as overheard from Alice along with Alice's and Bob's traces. Figure 3.11 shows the bits that Eve would generate if she carried through with the key-generation procedure. The mutual information [25] (M.I.) between Eve's data and Bob's data is a useful measure of the information learned by Eve about Bob's measurements $\underline{\hat{h}}_b$ and can be compared to the mutual information between Alice's and Bob's estimates $\underline{\hat{h}}_a$ and $\underline{\hat{h}}_b$. Table 4.1 gives these mutual information values computed using the method in [2]. As a consequence of the data processing inequality [25], any processing of the received signal by Eve would only reduce her information about the Alice-Bob channel, and therefore, the M.I. values in Table 4.1 provide upper bounds on the information about the Alice-Bob channel leaked out to Eve. The results from our second experiment with a moving Bob are very similar to the ones shown for the first experiment, although with fewer bits produced. Due to space limits, we do not present plots for the mobile experiment but instead summarize our results in Table 4.1. It is notable that in the static case the M.I. between Eve and Bob is orders of magnitude smaller than that between Alice and Bob and very close to zero, indicating that Eve is unable to derive any significant information about the Alice-Bob channel. Further, the M.I. between Eve and Bob is lower in the mobile case compared to the static case, indicating that mobility

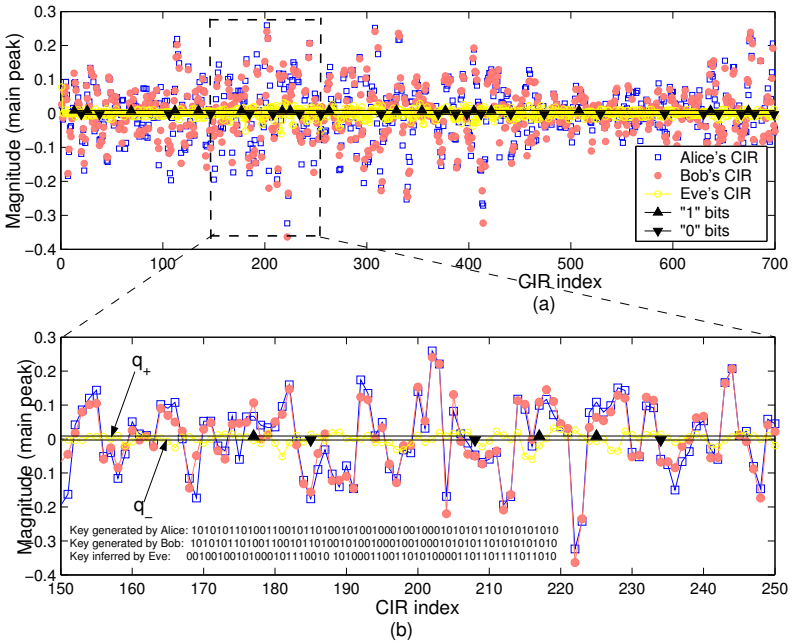


Figure 3.11: (a) Traces of Alice and Bob after subtracting average signal power. Using $m = 5$, $N = 59$ bits were generated in 110 seconds ($R_k = 0.54$ s-bits/sec) while $m = 4$ gives $N = 125$ bits ($R_k = 1.13$ s-bits/sec.) with no errors in each case. (b) A magnified portion of (a)

actually helps strengthen the secrecy of generated keys.

3.6.2 Coarse measurements using RSSI

Experiment setup: The setup consisted of three off-the-shelf 802.11 radios. Alice was configured in `AP` mode along with a virtual monitor interface to capture received packets. Bob was a client, consisting of a laptop with a 802.11a card in `station` mode, along with virtual monitor for capturing received packets. Eve was a third 802.11a node, identical in configuration to Bob, but capable of receiving packets from both Alice and Bob. In our experiment, Alice was stationary, while Bob and Eve moved along fixed trajectories. Atheros [48] WiFi cards based on the 5212 chipset were used at each end along with the Madwifi driver [49] for Linux. The experiments were done in the 5.26 GHz channel. The AP-station configuration ensured that MAC-layer clocks at the two nodes were synchronized. Figure 3.12 (b) shows the layout of the office building along with the location of

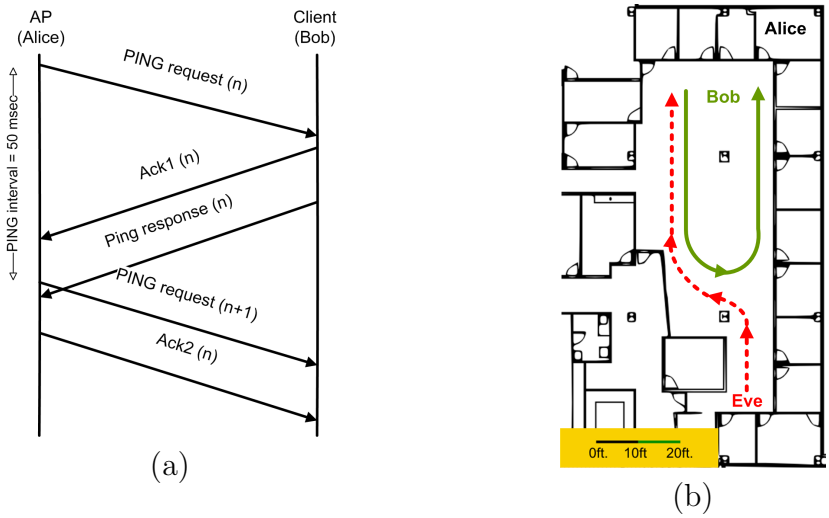


Figure 3.12: (a) Timing diagram for collecting RSSI information using PING packets in the RSSI-method. (b) Experimental Layout for RSSI-based method showing trajectories of Bob and Eve, while Alice (the AP) was kept stationary.

the fixed AP and path followed by the mobile client. ICMP `PING` packets were sent from the AP to the client at a rate of 20 packets per second. Each `PING request` packet received at the client generates a MAC-layer acknowledgment packet sent back to the AP, followed by a `PING response` packet. Upon receiving the `PING response` packet, the AP similarly replies with a MAC-layer ACK packet.

Figure 3.12 (a) shows the sequence in which these packets are sent. A `tcpdump` [50] application running on both the AP and the client recorded and time-stamped all packets received on the monitor interface of each user. The experiment consisted of sending $8,000$ packets from Alice to Bob. The `tcpdump` traces at each end were filtered using the MAC address to keep only the four types of packets described above. Further, RSSI and MAC-timestamps were pulled out of each packet to generate a $(timestamp, RSSI)$ trace. **Modification to handle timestamps:** We note that since the precise time instants at which the `PING response` and `PING request` messages are received by Alice and Bob respectively cannot be controlled, there was no way to guarantee that successive `PING`

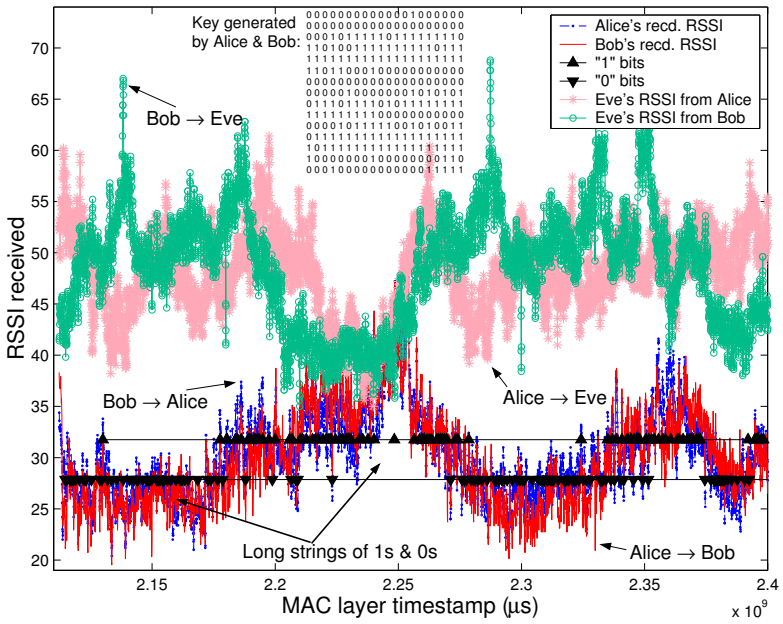


Figure 3.13: RSSI traces of Alice and Bob and bits generated. This plot includes the effect of shadow fading.

`request` messages received by Bob were separated in time by exactly one `PING response` received in between by Alice. Therefore, MAC-layer timestamps were essential to time-align RSSI information at Alice & Bob since we did not have index numbers with which to reference RSSI values. This required a slight variation in our algorithm to handle MAC-timestamps instead of indexes in the messages exchanged between Alice and Bob. Instead of sending index numbers to Bob, Alice now sends MAC-timestamps in the message L (see Algorithm 1 in Section 3.4). For each MAC-timestamp sent by Alice, Bob finds the MAC-timestamp in his own trace that is closest in time to the value of the timestamp sent by Alice. Bob uses the packet determined in Step 2 above as if it were the index sent by Alice. He checks for the presence of excursions above q_+ or below q_- centered at this packer as in Algorithm 1.

The *RSSI* field in the Prism header of received 802.11 packets reports RSSI as integers, thereby providing only coarse channel information. Moreover, the 802.11 cards at Alice and Bob may not be relatively calibrated and thus may

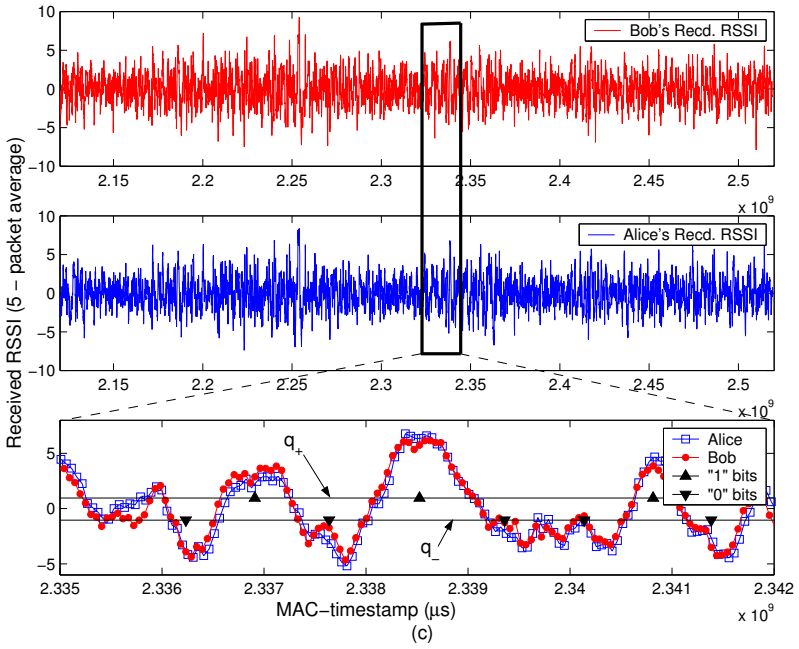


Figure 3.14: RSSI traces of Alice & Bob after subtracting windowed mean. We get 511 bits in 392 sec using $m = 4$ ($R_k = 1.3$ s-bits/sec.)

report different values of RSSI. We found in our experiments that although lacking calibration, the temporal *variations* in RSSI are matched in Alice's and Bob's traces. This problem was solved by subtracting out a moving average of the trace to remove the effects of slowly varying average signal power, as in the CIR method. Figure 3.13 shows the raw RSSI traces collected by Alice and Bob plotted against their received MAC-timestamps. As in the CIR-method, the traces exhibit strong variations in average signal power. We average out the large-scale variations and keep only the small scale fading effect. The result is shown in Figure 3.14. Our algorithm produces secret bits at a rate of almost 1.3 s-bits/sec using $m = 4$, where q_+ and q_- were computed independently by each user as in (3.13)-(3.14) with $\alpha = \frac{1}{2}$.

Contrasting Eve's attempts: We plot the RSSI traces captured by Eve for both Alice's and Bob's signal in Figure 3.13. The traces from Alice and Bob after considering only variations about a moving average, are shown in Figure 3.14. Even with coarse RSSI measurements that represent the average received

CIR-based method

Value of m used	4
Choice of q_+, q_-	mean $\pm 0.125\sigma$
Duration of experiments	1326 sec ($\sim$ 22 min.)
Inter-probe duration	110 msec.
Static case:	
Average secret-bit rate	1.28 s-bits/sec.
I(Alice; Bob)	3.294 *bits*
I(Bob; Eve)	0.0468 *bits*
Mobile case:	
Average secret-bit rate	1.17 s-bits/sec.
I(Alice; Bob)	1.218 *bits*
I(Bob; Eve)	0.000 *bits*

RSSI-based method

Value of m used	4
Choice of q_+, q_-	mean $\pm 0.5\sigma$
Average secret-bit rate	1.3 s-bits/sec
Inter-probe duration	50 msec.
Duration of experiment	400 sec.
I(Alice; Bob)	0.78 *bits*
I(Alice; Eve)	0.00 *bits*
I(Bob; Eve)	0.07 *bits*

Table 3.3: Summary of experimental results. $I(u_1; u_2)$ denotes the mutual information (M.I.) between the measurements of users u_1 and u_2.

signal power per-packet over the entire 802.11 channel bandwidth, Alice and Bob can exploit reciprocity of their channel to successfully generate secret bits at a fairly good rate. We compute the pair-wise M.I. between the traces of Eve, Alice and Bob in Table 4.1. As in the CIR-method, we find that Eve gets almost no information about the Alice-Bob channel.

3.7 Discussion and open problems

We now discuss insights related to our scheme, summarizing fundamental trade-offs, and further discuss potential security threats. We showed in Section 5 that the rate at which Alice and Bob derive secret bits from a time-varying channel is

limited by the rate of variation in the channel. To maximize rate, we must probe the channel rapidly. For the fastest probing rate, the parameters m, q_+ and q_- can be tuned to keep the probability of error within an acceptable bound. Increasing m or the magnitudes of q_+, q_- decreases the error probability at the cost of a decrease in the secret-bit rate. Increasing temporal variation in a channel increases the secret-bit rate up to a point, after which further increase produces a rate decrease, unless accompanied by a proportional increase in the channel probing rate.

The natural decorrelative properties of fading provides our scheme security against eavesdroppers. We confirmed this through our system implementation. Standard randomness tests indicate that our algorithm is resilient to an eavesdropper exploiting randomness defects. However, it is worth noting that key rates significantly greater than the maximum Doppler frequency cannot result in truly random bits. Thus we recommend conservatively setting the probing rates relative to the dynamics of the fading environment. Beyond a passive adversary, we have addressed the threat of an active adversary impersonating Alice or Bob. Coping with spoofing of probes can be dealt with using techniques similar to [32]. We have addressed spoofing of messages following probing by providing a modified algorithm that uses some of the shared secret bits for data-origin authentication. Thus, Eve cannot thwart the key-generation process by impersonating either legitimate user without getting detected.

A further concern common to all key establishment schemes is the man-in-the-middle attack. A man-in-the-middle attack against our algorithm is only possible if Alice and Bob cannot hear each other's probes (e.g. they are not within radio range, or Eve talks to Alice and Bob separately), otherwise Eve's attack causes discrepancies that are easily detectable by Alice and Bob. If Alice and Bob do fall victim to a man-in-the-middle attack, this can be detected by the following identity-based authentication mechanism: Alice asks Bob to send her the keyed

hash of the answer to a specific question using their (supposed) shared key as an input to a cryptographic hash function. If Eve relays this question to Bob, then Bob's answer will be useless to Eve (assuming only Alice and Bob know the answer to the question). We note this method requires that Alice and Bob share some secret information known only to them. This is necessary as each user must authenticate the *identity* of the other in order to prevent a man-in-the-middle attack, and is necessary even for classical key establishment schemes like Diffie-Hellman.

One might inquire whether varying levels of interference at different locations in the environment would affect our key generation process. We have provided fundamental tradeoffs relating signal-to-interference levels to quantizer parameter selection for an isotropic noise background. However, by conservatively selecting protocol parameters (e.g. selecting a larger value of m (see Figure 3.4)), we achieve improved robustness in the key generation process at the cost of lowering the rate.

An important unsolved question is whether the secret key generation rate can be improved significantly by using a vector quantizer instead of the excursions based approach we have used in our algorithm. We conjecture that it is possible to train a vector quantizer based on training data and that such a quantizer would allow us to extract a sequence of bits such that the fraction of bits that are in error between the sequences of Alice and Bob is low. This would in turn enable the use of an error-correcting code based reconciliation algorithm of the type that will be described in the next chapter. The problem with a reconciliation approach based on error correcting codes is that the reconciliation message needs to be authenticated in order for the protocol to be robust against an active attack. While our algorithm achieves this using a two way message exchange between Alice and Bob, it has the limitation that Eve can reduce the number of extracted bits (and hence the secret bit rate in s-bits/sec) by replacing upto a constant fraction of

the indices in L without going undetected. With a single-message reconciliation protocol based on error correcting codes, this problem can be aliviated as we will show in the next chapter; however, this requires a fairly low error rate between the bits of Alice and those of Bob - otherwise the block codes that are needed to correct the errors need to be of very long length, and this introduces a large delay before Alice and Bob can get a key.

Chapter 4
Proximity based extraction of shared secret keys

In this chapter, we explore another mechanism for harvesting common randomness from wireless channels that can have an important practical use: secure pairing of wireless devices that are in physical proximity. We first introduce the problem of why communication between wireless devices that are in physical proximity is fraught with

4.1 Introduction

The density of devices with wireless interfaces is growing at an increasingly rapid pace. With this growth in wireless interfaces, there will be a corresponding need for devices to establish spontaneous communications as they move about and come in proximity of each other. As an example, two people meeting for the first time may wish to exchange data between their mobile devices, or a passenger at a train station may wish to pay for a ticket by having his/her phone interact with an electronic ticket booth. Securing such interactions is essential, but is also a very challenging problem. Devices have varying user-interface components (e.g they may or may not have a screen, buttons, or LEDs), which makes it harder to design a uniform method to establish secure associations. Further, it is unrealistic to expect these devices to maintain and update cryptographic material, especially since back-end key management servers are not always available in mobile environments. Thus, in order to allow wireless devices to communicate securely, these devices must establish cryptographic keys on their own, as

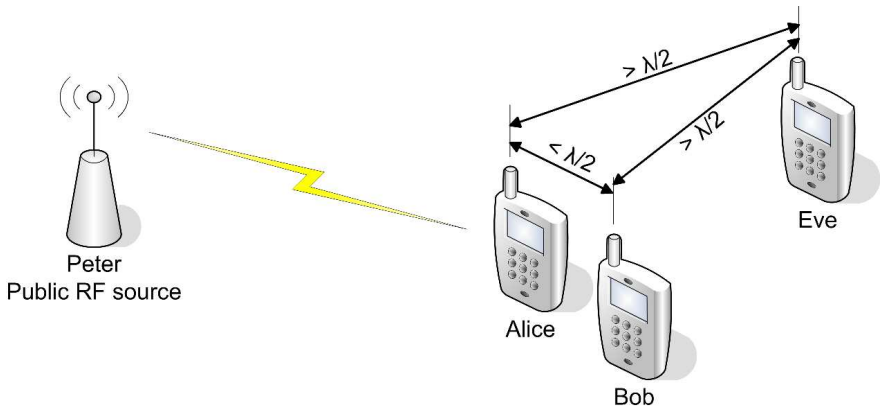


Figure 4.1: The wireless channel from a public source (Peter) of RF transmissions can be used as a source of shared randomness by the legitimate parties (Alice and Bob) who are in physical proximity compared to the adversary (Eve) to extract a secret key. Here, λ is the wavelength of the public RF transmission.

and when needed. However, given the broadcast nature of the wireless medium and in the absence of a prior relationship, how does one device know that it is really setting up a secure link (say, using the Diffie-Hellman key establishment protocol) with the device it intends to communicate with? As a result, setting up a secure link between wireless devices in proximity is presently a surprisingly cumbersome procedure and often requires significant human intervention in the form of setting up a shared key. This trend is all the more problematic for devices that are meant to communicate in a machine-to-machine manner, i.e., without human involvement. In this chapter, we show that by employing the principle of randomness inherent in radio propagation, one can securely establish shared secrets between two wireless devices in physical proximity.

Clique is based on the observation that wireless devices in proximity have access to a shared source of randomness – namely, small scale temporal variations in their wireless channels with respect to a common RF source, known as small-scale *fading* – which is not available to any other wireless device that is not in close proximity of the legitimate devices (see Figure (4.1)). The fact that the wireless

environment between a public transmitter (e.g. a television broadcast tower) and receivers is typically multipath rich, implies that the small-scale fading relative to the transmitter as witnessed by two receivers in proximity will intimately depend on their separation. As long as these devices are located within close proximity, the channel responses characterizing each device's fading process to the public source will be highly correlated, while the fading witnessed by any eavesdropper that is not in proximity, will be uncorrelated, providing almost no utility to an adversary. Hence, two devices in close proximity can listen to public RF transmissions to obtain correlated and hard-to-predict stochastic processes that can be exploited to form keys. Here, proximity is defined relative to the wavelength of the public RF source – devices are said to be in proximity, and capable of forming a shared key, if they are within a separation of half the wavelength that they are monitoring. Similarly, the minimum distance of an adversary from the legitimate devices, for which the method is secure, is also half of this wavelength.

It is important to note here that the choice of wavelength (and thereby the type of RF source) used by Clique depends on the application, e.g. for FM Radio and TV broadcasts, available wavelengths range from ~ 42 cm to 5.5 m. Applications that require devices to pair at larger separations must use larger wavelengths. Of course, this also proportionally increases the minimum safe distance for an adversary. Figure 4.2 illustrates the fading phenomenon through the temporal variations in the received power of an FM radio station, recorded at two receivers in close proximity. It is the correlation between the temporal variations that makes the extraction of a common key possible, and the decorrelation at larger separations that makes this key secure from sufficiently distant adversaries (See adversary model in Section 4.3.2). The formation of secret keys in Clique is *unconditionally* secure, i.e. it does not rely on the computational boundedness of an adversary to ensure the secrecy of the established key, unlike the Diffie-Hellman protocol.

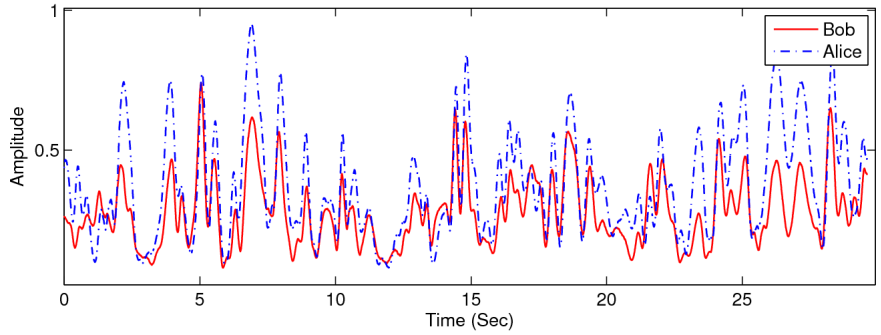


Figure 4.2: A 30-second trace of the temporal channel variations observed at two receivers tuned to a an FM radio broadcast frequency (98.7 MHz) when the receivers are $0.1 \times$ wavelength $\sim$ 30 cm apart. Clique exploits this spatial correlation.

Our contributions can be summarized as follows:

1. We describe a simple algorithm that allows wireless receivers in physical proximity to convert their correlated channel observations into identical sequences of bits. This sequence can then be used as a shared encryption key. We evaluate the performance of this algorithm, both analytically and through measurements of real wireless channels.

2. To increase the rate at which secret bits can be extracted by Clique (and reduce the time to form a key), we propose and evaluate the concurrent monitoring of multiple public RF sources.

3. Finally, we examine whether an active adversary can influence the bits that are extracted by the legitimate terminals if the adversary itself controls the public wireless source. To deal with this case, we develop and experimentally evaluate a method based on the *differential phase* of the received signals, which allows secret bits to be extracted despite adversarial control of the transmit signal.

4.2 Related Work

Establishing secret bits without a third party is perhaps best exemplified by the Diffie-Hellman key agreement protocol[51]. The security of Diffie-Hellman and public key methods is tied to the assumption of a computationally bounded adversary, and further, must also leverage some form of a priori shared information (e.g. a bootstrap key, as in the case of the Station-to-Station Protocol[52]) to prevent a man-in-the-middle attack. One approach to establishing secrets without trusted third parties or computational assumptions is to take advantage of a a physical resource that can facilitate the sharing of a key. The best known example of this is in the area of quantum key distribution (QKD) [53, 54] which typically relies on optic fiber links. Although Clique exploits a different type of channel, it also extracts bits from a source of shared randomness and there is a vast body of information-theoretic literature describing bounds and principles by which this common randomness can be extracted[19, 20, 55].

More recently, the wireless channel itself has received attention as being a resource for sharing keys[30, 56, 34, 35, 57, 58, 33, 59]. All these works explicitly make use of the *reciprocity* of the wireless channel between the legitimate terminals and assume that they engage in active probing of the channel *between* themselves in a time-division-dulpexed manner. Consequently, these methods: (i) *require* the legitimate terminals to be sufficiently distant in space, so they may utilize the fading process between themselves as a source of randomness[1], and (ii) cannot employ shared randomness as an authenticator of a shared context (unlike Clique which implicitly employs proximity as the shared context). In contrast, Clique requires proximity between the legitimate terminals, so as to

[1]Devices within spatial proximity, especially when in line-of-sight, do not offer an observable fading channel between themselves because the line-of-sight path dominates over multipath propagation.

leverage a public RF source as shared, but private source of randomness. Further, techniques based on channel-reciprocity are inherently limited in their key establishment rates by the rate of temporal variation of their channel. However, as there are numerous public RF sources (e.g. multiple FM radio stations), Clique easily scales to higher key rates by merely monitoring a greater number of public sources in parallel.

Earlier work on secure device pairing[60, 61, 62, 63, 64] has focused on building a common shared context unique to two mobile devices. [61, 62] propose a technique to utilize shared movement to accomplish secure pairing. In [60], the authors make use of Faraday cages to facilitate keying. Our work is perhaps most similar to [64], in which a computationally secure technique is presented to authenticate co-located devices using knowledge of their shared radio environment as proof of physical proximity. Another related technique is near-field communication (NFC) for limiting the range of communications [65]. A specific example of such work is [66], which uses LDPC codes to provide spatially bounded communication. Unfortunately, the security of near field and short-range communication assumes a weak adversary, and fails should an eavesdropper employ a directional antenna with even a modest amount of gain [15, 67].

4.3 System Model

We use *Alice* and *Bob* to refer to the two legitimate parties wishing to establish a secret key, while *Eve* is the adversary, and *Peter* is a public source of radio waves. Our constructions require a basic understanding of the stochastic behavior of wireless channels. We briefly summarize the relevant properties that we employ, and refer the interested reader to chapter 2 for a more detailed treatment of wireless propagation.

4.3.1 The wireless channel

We assume that a time-varying fading wireless channel exists between Peter and Alice, Bob and Eve. This is typically true for terrestrial wireless systems such as FM, TV or cellular networks. If two receivers are tuned to a common transmitter, the fading channels experienced by them are correlated if the separation between them is less than $\lambda/2$ but independent otherwise. The signal transmitted by Peter excites the channels between Peter and the three users, allowing each to make measurements of the channel's random state. We will refer to the time interval over which a channel decorrelates completely as the *coherence time*, T_c of the channel. Whenever necessary for *analysis*, we will assume a Rayleigh fading channel [68] from Peter.

4.3.2 Adversary Model

To start, we assume a passive adversary that can listen to the same public source as Alice and Bob, but that for a given public source, Eve is located far from Alice and Bob. The minimum distance that an adversary must be from Alice and Bob in order to guarantee secrecy of the extracted key, can be thought of as a security parameter that characterizes the quality of the overall system (the smaller, the better). This value is proportional to the wavelength of the public source used by Alice and Bob. We further assume that Eve is aware of all the parameters of the key-generation algorithm and the protocol followed by Alice and Bob, and that Eve only has 'read access' to messages exchanged between Alice and Bob during the secret key generation phase, i.e. she cannot replace or modify any such messages. The rationale for this assumption is the following: suppose Alice sends a message to Bob as part of the key extraction protocol. On the wireless medium, it is difficult for an active Eve to completely replace Alice's message or modify it en route to Bob without Bob receiving either a message garbled

by the overlap of two signals, or two separate messages. This is especially true when Alice and Bob are in close proximity. In other words, it is easy for Bob to detect the presence of an active Eve. Finally, we assume that any messages exchanged between Alice and Bob during the key-generation phase are error-free. This can be ensured by employing appropriate modulation, error correcting code, and transmit power. We will also consider an active attack, where Eve controls the public source itself. We will present a modification to our basic algorithm that allows the secret key generation to continue to work even when Eve controls the transmitter (see Section 4.5.3).

The most important factors in our secret key generation system are the relationship between the distance between Alice & Bob and their ability to extract a secret key, and the minimum distance at which a passive eavesdropper must be relative to Alice and Bob in order to guarantee that she cannot generate the same key as Alice and Bob. We will explore these relationships in the analysis in the next section, with an emphasis on the resulting bit error rates (BER) in the key extraction process. Our goal is to make the BER between the bit sequences of Alice and Bob as small as possible, and as close as possible to 0.5 for Eve's sequence, with respect to the sequences of Alice and Bob. If this can be achieved, then suitable error correcting codes can repair the errors in Alice's and Bob's bit sequences without allowing Eve to derive the same sequence. Further, complementing BER, we would also like Alice and Bob to extract these bits sufficiently fast under the guarantee that Eve cannot estimate these bits. In our experimental evaluation, we will make use of empirically computed *mutual information* [25] as a metric to measure the inability of Eve to derive any useful information about Alice's and Bob's channel estimates.

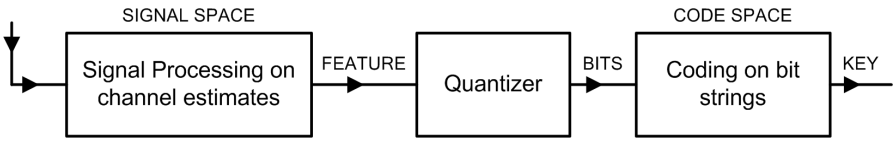


Figure 4.3: Each user firsts operates on the sequence of channel estimates in the signal space, the resulting extracted feature is then quantized to obtain bits, and finally error correcting codes are used to correct differences between the bit sequences of Alice and Bob.

4.4 Extracting secret bits

Our framework for extracting secret bits using the time-varying wireless channel between Peter and Alice/Bob involves the following steps (see Figure 4.3):

1. Each user independently estimates the channel between the public source and itself, periodically every T seconds ($T \ll T_c$).

2. The sequence of channel estimates is used to estimate the coherence time of the channel T_c.

3. The sequence of channel estimates is processed locally every T_c seconds to extract a scalar quantity, which is quantized to obtain one or more bits at each user.

4. The first three steps are repeated until a sufficient amount of bits are collected.

5. *Reconciliation*[22]: Alice sends Bob a set of parity bits with respect to an error correcting code, to repair, with high probability, the discrepancies between the bits generated by Alice and Bob.

6. *Privacy amplification* [26]: Information leaked to Eve in step 5 is removed by Alice and Bob, by shrinking the common sequence, such that Eve has no information about the final sequence.

4.4.1 Basic signal processing functions

Each user locally computes a single threshold value q to build its own quantizer $Q(\cdot)$. We use a single-bit quantizer since, at the typical distances between Alice and Bob, we found that the channel estimates of Alice and Bob do not have enough mutual information to reliably support more than one bit. For a Rayleigh fading process, q is taken to be the median of all the channel estimates collected by the user. This ensures that the '0' and '1' bits produced by the quantizer are equiprobable.

The discrepancies between the bits produced by the quantizers at Alice and Bob arise from three separate factors: (i) the fact that Alice and Bob are not at identical locations implies that their channels to Peter are not identical but only statistically correlated, (ii) independent noise in the channel estimates that corrupts the true channel from Peter, and (iii) calibration differences in the hardware at Alice and Bob. To extract identical bits we need to address these problems. The noise can be largely eliminated using a very simple stochastic averaging technique, while appropriate quantization and coding involving a message exchange between Alice and Bob will allow us to reconcile differences arising from channels not being identical. Differences in hardware can be mitigated by considering only variations in the channel estimates and normalizing the estimates to a common mean and variance. Without loss of generality, in the rest of this chapter, we assume Alice to be the protocol initiator and that she sends the error correcting information to Bob.

Stochastic noise averaging

The temporal rate of variation of the channels from Peter to Alice & Bob (characterized by the coherence time T_c) is typically much slower than the rate at which Alice and Bob can obtain new channel estimates, especially when there

is an always-on reference signal in Peter's transmit signal. This is the case for ATSC television signals that contain an always-on pilot tone, as well as for FM radio broadcasts, for which the 200 kHz-wide signal has constant transmit amplitude and, hence, any variations in received signal power are a consequence of the wireless channel. While the coherence time of a channel is typically of the order of hundreds of msec (if Alice and Bob are in a moving vehicle, this can be lower by a factor of $10 - 100$), the rate at which new channel estimates can be made is essentially equal to the sampling rate, which we set to 250 kilosamples per sec in all of our experiments. The ability to obtain new channel estimates more rapidly than the channel varies is beneficial for reducing the noise in the channel estimates: Each user computes a moving average of the channel estimates within a short window and uses the average estimate as the input to the quantization stage With frequent enough channel estimates, practical levels of noise can be almost eliminated by averaging. We then continuously slide the window by one sample to get new channel estimates.

T_c estimation, Normalization and Sampling

The coherence time T_c of a Rayleigh fading channel is related to the Doppler spread f_d, and hence can be estimated from the level crossing rate of the channel [17], which for a Rayleigh fading channel is [17]

$$LCR = \sqrt{2\pi} f_d \rho e^{-\rho^2} \tag{4.1}$$

where ρ the ratio of the level-crossing threshold considered and the root mean square threshold. From this we can estimate T_c as:

$$\hat{T_c} \approx \frac{3\rho e^{-\rho^2}}{2\sqrt{2\pi} LCR}. \tag{4.2}$$

The set of channel estimates obtained by Alice and Bob form a discrete-time stochastic process and, as noted earlier, to remove the influence of hardware differences, Alice and Bob each normalize their respective processes to have zero

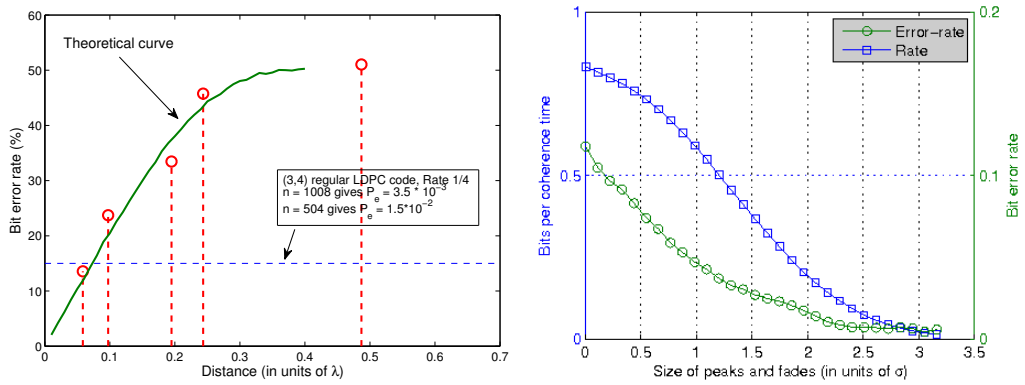
mean and unit variance by subtracting and then multiplying by suitable constants. The set of N noise-reduced channel estimates is then divided into blocks of size $\lceil(\hat{T}_c/T)\rceil$ estimates, where each block represents roughly one coherence time interval. Finally, the element in the center of the block is picked and quantized using the quantizer, $Q(\cdot)$.

4.4.2 Reconciliation using error control codes

We now summarize a known approach based on error correcting codes that allows the reconciliation of correlated bits into identical bit sequences, with very high probability[58]. We refer to this construction as a purely-code based construction for reconciliation and provide a sketch of the steps involved (see Appendix A for more detail).

Let w and w' denote n-bit strings obtained by Alice and Bob respectively after the quantization stage above. Since successive bits in each of these strings are derived from independent channel estimates, the bits are independent. Discrepancies between w and w' can therefore be modeled by the relationship $w = w' + e \pmod 2$, where e is an error sequence of n bits. Each bit in e is 1 with a probability $\epsilon < 0.5$, independent of all other bits in e. In other words, we can think of w' as being the result of transmitting each bit of w through a memoryless binary symmetric channel (BSC)[25] with a cross-over probability of ϵ. Accordingly, we will say that the 'bit error rate' between the bit string of Alice and that of Bob is ϵ. Note that the capacity of a BSC with cross-over probability ϵ is given by $C = 1 - H_b(\epsilon)$, where $H_b(x) = -x\log_e(x) - (1-x)\log_e(x)$ is the binary entropy function. Therefore, if Alice's and Bob's bits differ in a fraction $\epsilon \in (0, \frac{1}{2})$ of the places, they can theoretically obtain a maximum of $C < 1$ bits per raw bit that they possess.

The main reconciliation step involves considering w and w' to be corrupted versions of a codeword from an (n, k) linear block code, and the transmission



(a) (b)

Figure 4.4: (a) Fraction of bits in error at Alice & Bob as a function of the distance between them, using a TV pilot signal at 584.31 MHz. Also shown is the theoretical BER curve, and a suggested LDPC code (from [1]) that can fix an error rate of 15% using large blocklengths. (b) Larger peaks & fades cause lower values of error-rate at the cost of lower values of rate in bits/sec. σ is the mean power. The error rate plot is for $d = 0.1\lambda$.

of $(n-k)$ syndrome bits P (with respect to this code), from Alice to Bob in cleartext. Alice initiates the exchange by sending Bob the syndrome $P = Hw$ for the parity check matrix H. For a given value of ϵ (which depends only on the quantity d/λ), a code with a suitable error correcting ability can be chosen to allow Bob to decode w using w' and P. Since the $n-k$ syndrome bits are sent by Alice in the clear, Eve obtains *partial information*. At most Eve's learns $n-k$ bits of information about Alice's and Bob's n-bit strings. Therefore, in the final step, Alice and Bob reduce the size of their common bit-string by $n-k$ bits to obtain k-bit strings, about which Eve has absolutely no information. There are several ways in which this reduction can be accomplished. One is to use a universal hash function $f : \{0,1\}^n \mapsto \{0,1\}^k$, another is to simply use the k-bit pre-image of the n-bit codeword that the common n-bit string maps to with respect to the (n,k) linear block code that Alice and Bob employed.

Problem with purely-code based construction. For typical values of

d/λ in our problem, the error rate, ϵ is fairly large. When using an ATSC television pilot at 584.31 MHz (channel 33) and when Alice and Bob are only 1.5 inches apart, we have $e \approx 0.15$. Figure 4.4 shows error rates we measured for various values of d/λ using a TV signal on channel 33, along with the theoretically predicted error rates. To repair such a high error rate between Alice's and Bob's bits, we require either a very low rate code (i.e. we must pick n, k such that $n/k \ll C$), or a code with a very large block length n. The former results in very few secret bits per second, while the latter entails an impractically long delay before enough bits can be gathered to produce an output. It might be argued that a larger value of λ would address the problem. However, picking a larger λ also entails a proportional worsening of our security parameter by requiring an increase in the minimum distance at which Eve must be from Alice and Bob. It is possible to address this problem without worsening the security parameter, using a better quantization approach.

In most prior work on reconciliation, applications of the code-based procedure above typically assume a discrete time memoryless source that produces correlated random variables (X_i, Y_i) at Alice and Bob such that $X_i, i = 1, \ldots, n$ are independent, and so are the Y_i's. Having Alice and Bob sample their respective stochastic processes once per coherence time would convert their stochastic processes to such a discrete time memoryless source, but would ignore the temporal redundancy that the stochastic processes provide. It is this temporal redundancy in Alice's and Bob's *stochastic processes* that we exploit to improve the quality of the raw bits obtained by Alice and Bob.

List encoding. We now present a simple enhancement, which we term *list encoding*, that significantly lowers the bit error rate between the bits of Alice and Bob to a value suitable for error correcting codes with reasonably small block lengths (not to be confused with list-decoding, a topic of much recent activity in the theoretical computer science community).

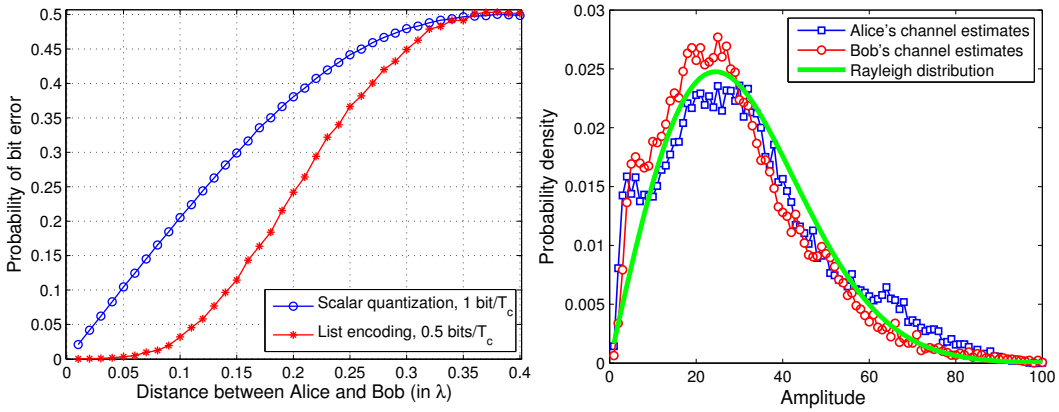
In list encoding, Alice sends a message to Bob to convey the time instances at which the quantized versions of their channel estimates (using some appropriately defined quantizer) are highly likely to agree, without revealing any additional information to Eve. In our construction, Alice locates deep fades (mapped to bit '0') and sharp upward peaks (bit '1') in her stochastic process, and sends Bob a list of time-indices of the corresponding minima and maxima respectively, in her stochastic process. In order to keep the successive extrema picked by Alice independent of one another, we require that Alice cannot choose any extrema that are within T_c time of each other. As a result, even though the values of the channel estimates at Alice and Bob are not necessarily better correlated at the location of extrema in Alice's process (i.e. quantizing the value at the extrema does not significantly improve the error rate), the *type* of extrema (i.e. minimum or maximum) at Alice and Bob at these locations have a much stronger correlation. Upon receiving this list, Bob determines for each time-index the location of the nearest extremum in his sequence of channel estimates. For a maximum, he assigns the bit '1', and for a minimum, he assigns a '0'. This set of rules forms the basis of the list-encoding quantizer.

Having Alice send the indices of her extrema reduces the fraction of bits that differ at Alice and Bob at the cost of the rate at which these bits are gathered because Alice can only pick an extremum at a rate strictly less than once per coherence time interval – once she picks an extremum, she must wait for an interval of at least one coherence time before picking the next one in order to maintain independence of extracted bits. However, the fraction of bits, ϵ_2, that differ at Alice and Bob using list encoding is much smaller than the error-rate ϵ we observed for the sampling and quantization approach in the purely code-based construction, and more than makes up for the drop in rate. More generally, the error-rate for list encoding can be lowered even further if the maxima and minima are chosen from peaks and fades that are larger than a certain threshold.

Here by the size of a peak/fade, we mean the magnitude of the fall and rise in amplitude involved in a single peak/fade. In Figure 4.4(b) we show how the error-rate between Alice's and Bob's bits decreases as this threshold (expressed as a multiple of the parameter σ of the Rayleigh distribution) is varied, at the cost of a decreasing rate. For our experimental study in the following section, we adopt a threshold of σ. For a more detailed comparison of the two approaches, with respect to rate and error-rate, we refer the interested reader to Appendix B.

Errors between Alice's and Bob's bits can be repaired as before, using an appropriate error correcting code. The lower incidence of errors ϵ_2 between their bits, however, implies that a greater number of identical secret bits can be generated by Alice and Bob per unit time without requiring impractically long block lengths. We have chosen to use the $(23, 12, 7)$ binary Golay code because it has a short blocklength, efficient decoding algorithms, and a reasonably good error correcting performance in the range of error-rates ϵ_2 that list encoding produces (see Figure 4.4(b)). For example, list encoding produces about 0.6 raw bits per coherence time, which, at a distance of 0.1λ, results in an error rate of $\epsilon_2 = 0.04$ (see Figure 4.4(b)). After applying the Golay code, the bit error rate reduces to about 2×10^{-3}, providing about $\frac{12}{23} \times 0.6 = 0.31$ secret bits per coherence time to Alice and Bob. It is possible to improve the error performance by choosing a code with a lower rate and/or a larger blocklength.

As a final comment, we have only presented the most basic form of list encoding involving a binary quantizer. List encoding can be generalized by noting that its enabling principle is that Alice chooses time-locations in her stochastic processes where a more general (non-binary) quantizer at Bob would have a greater chance of extracting the same symbols as Alice. Hence, we need not restrict our attention to extrema in Alice's and Bob's stochastic processes, but instead can generalize to other, longer temporal features, e.g. Alice and Bob can employ a set



(a) (b)

Figure 4.5: (a) Error rate between raw bits at Alice & Bob for a simple quantizer and for list-encoding using extrema. (b) Distribution of the amplitude of the pilot tone at 584.31 MHz in the ATSC television signal on channel 33. Amplitude, as a percentage of the maximum observed amplitude.

of time-constrained waveforms as templates for matching portions of their processes. While extrema provide a convenient and simple feature for matching, we believe that there is room for optimization over the feature space in list encoding and expect to study this in the future.

4.5 Evaluation

We now evaluate secret bit extraction at co-located receivers by first presenting simulation results illustrating the superior error performance of list encoding compared to a simple quantization approach, and then present results from our prototype implementation. We used Jakes model for Rayleigh fading channels[40] to describe the effects of fading. According to the Jakes model, the spatial correlation function for the in-phase (or equivalently, the quadrature-phase) component of the channel response experienced by two receivers (i.e. Alice and Bob) is $J_0(2\pi d/\lambda)$, where d is the receiver separation and λ is the carrier wavelength.

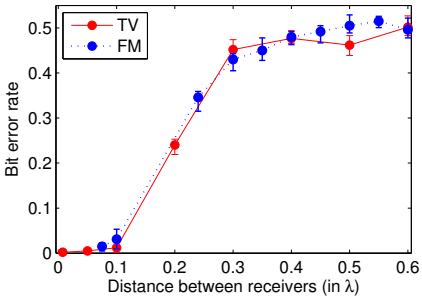
Similarly, the correlation between the *magnitudes* of the channel fading processes at Alice and Bob is $J_0^2(2\pi d/\lambda)$. Using these relationships, it is straightforward to generate complex Gaussian stochastic processes using autoregressive filters tuned to the parameters of the fading channel and which will have specified correlation. We conducted simulations where we generated two complex fading processes with varying separation d between Alice and Bob, and evaluated the fraction of bits where Alice and Bob differ after quantization using a simple scalar quantizer and our list-encoding quantization procedure on the magnitude of the resulting fading processes (but prior to application of error correction coding). Figure 4.5(a) shows the resulting error rate between Alice and Bob for the raw bits at different separation distances. From this curve, it is clear that list encoding significantly reduces the error rate, albeit at the cost of a reduced data rate. We show in Appendix B that the benefits that list encoding provides in terms of reduced error rate are critical to achieving good rate performance in final output bit streams. Since list-encoding is superior to a purely code-based construction, in the rest of this section we only present results for list-encoding.

4.5.1 Experimental Validation

Experimental setup. We conducted several real-world experiments using a software defined radio to validate our key generation procedure. In particular, we used GNU Radio [69] on the Universal Software Radio Peripheral (USRP) platform (see Figure 4.5(c)). GNU Radio is an open source software toolkit that provides a library of signal processing blocks. A single USRP supports the simultaneous transmission and reception of four real or two complex channels in real-time. The USRP interfaces with RF daughterboard hardware modules for RF transmission and reception. In our experiments, we used the TVRX daughterboard, which supports reception in the 50-860 MHz frequency band. It has a noise figure of 8 dB and thus provides adequate access to terrestrial

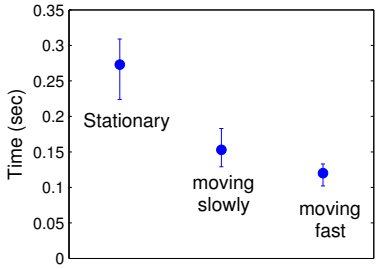
broadcast television and radio transmissions. For each experiment, Alice, Bob, and Eve were all outfitted with identical equipment (daughterboard, antenna, etc.).

We focused our measurement study on two widely separated frequency bands: (i) The television broadcast band between 512 and 608 MHz accommodating channels 21-36 in the US, and (ii) the FM radio broadcasting band between 88 and 108 MHz. While ATSC television signals themselves have a bandwidth of 6 MHz per channel, each channel contains an always-on constant-amplitude pilot tone. In our TV band measurements, we tune in to the frequency of the pilot tone for a given TV channel and track its amplitude. FM radio broadcasts also contain a pilot tone, but the pilot tone itself is FM modulated and therefore not useable for our purpose. Instead, we make use of the fact that the FM radio signal itself has a fairly narrow spectrum (200 kHz per channel) and is transmitted at constant power by the transmitter. Thus, we track the total received power over the 200 KHz band for a given FM radio channel as our estimate of channel gain since any temporal variations in the received power must be due to variations in the channel alone. In the television band, we carried out most of our measurements using the received signal from the ATSC channel 33, a 6 MHz wide channel, with an always-on pilot tone at 584.31 MHz. In the FM band, most of our data came from an FM radio station at 88.7 MHz. We also used multiple stations between 97.7 and 99.7 MHz to test whether multiple radio stations can scale up the amount of common randomness available to Alice and Bob (c.f. Section 4.5.2). In all, we collected close to 1.5 hours of data from the TV bands and 1 hour of data in the FM bands, spread over 6 days of measurements. In each experiment using TV signals, the receivers' channel measurements were time-synchronized using an external RF source as a reference trigger. For experiments with FM signals, the FM radio signal itself was used for (offline) time-synchronization between receivers. Since the two frequency bands differ by a factor of approximately 5

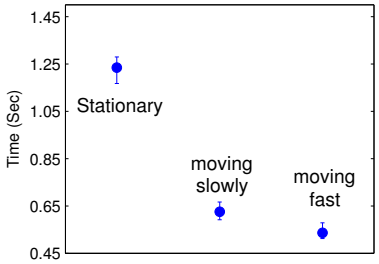


(a)

(b)



(c)



(d)

Figure 4.6: (a) The error rate between the bits obtained by two receivers using list encoding as the distance between the receivers is varied for a TV pilot at 584.31 MHz and an FM radio channel at 88.7 MHz. Each estimate is computed using six traces of duration one minute each. Error bars indicate the min and max estimates in each case. (b) A Universal Software Radio Peripheral (USRP) with two daughterboards and two antennas connected to laptop running GNUradio. Also shown are average estimates of the coherence time made using equation 4.2 for (c) TV (584.31 MHz) and (d) FM (98.7 MHz) signals. Each estimate is computed using six traces of one minute each. Errorbars indicate the min and the max estimates in each case.

to 6, the distance separating Alice and Bob required for extracting secret bits also differs by a similar factor. For example, 0.1λ is about 5 cm for a TV signal at 584.31 MHz, and about 33 cm for an FM signal at 88.7 MHz. The different requirements for the distance between Alice and Bob allow for different usage scenarios of our key extraction procedure. It must be noted that while the FM band, due to the larger wavelengths, allows key generation at larger distances between Alice and Bob, it has a proportionally weaker security parameter as it also requires eavesdroppers to be located farther from Alice and Bob.

We first verified that the amplitude of the fading process from our measurements follows our Rayleigh assumption. Figure 4.5(b) shows a plot of a normalized histogram of the amplitude of the pilot tone at 584.31 MHz on the ATSC television signal on channel 33, along with the main lobe of an ideal Rayleigh probability distribution. A similar agreement was observed on the FM band as well.

Spatial correlation

We measured the channel estimates for two receivers versus their separation (in terms of fraction of the wavelength λ). Figure 4.6 shows the error-rate between bits quantized using list-encoding at the two receivers for the TV and FM bands. As predicted by Figure 4.5(a), the error-rate between the bits obtained by Alice and Bob approaches 0.5 as the distance between the receivers approaches 0.4λ. Our results indicate that a practically 'useable' range of distances between legitimate receivers for secret bit extraction is $d < 0.1\lambda$. Since the bit error rate degrades to ~ 0.5 at around 0.4λ, this puts the minimum 'safe' distance for an adversary at four times the distance between the legitimate users.

Further, we conducted measurements with three time-synchronized USRP receivers: Alice, Bob and Eve, simultaneously tuned in to the same FM radio channel. Figure 4.7 illustrates the lack of correlation between the channel estimates

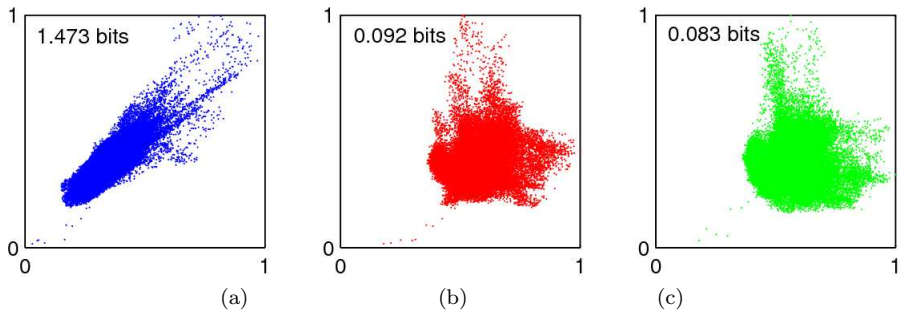


Figure 4.7: Scatterplots of the channel estimates of (a) Alice Vs. Bob, (b) Alice Vs. Eve and (c) Bob Vs. Eve, made using a three-user measurement on an FM radio channel. Eve is at a distance of $\lambda/2$ from both Alice & Bob, who are $d = 0.05\lambda$ apart. Each plot also shows an estimate of the mutual information per pair of channel estimates, computed using the algorithm in [2].

of Eve and those of Alice and Bob when Eve is at a distance of $\lambda/2$ from both Alice and Bob. Eve's low mutual information relative to Alice and Bob indicate her inability to estimate their bit sequence.

Our results for the average rate at which secret bits were extracted from a single RF source are summarized in Table 4.1 for the three types of environments we have considered, for a distance between Alice and Bob of 5 cm for the TV signal and 34 cm for the FM radio signal. We note that the rates reported here are raw key rates from list encoding prior to reconciliation. Applying the Golay (23,12,7) code on the bits obtained from the TV signal when Alice and Bob were shaken together fast (see Table 4.1), results in a final secret bit rate of 1.84 bps, with an error rate of 2.42×10^{-3}. In the remainder of this chapter, we report raw secret key rates. The results in Table 4.1 indicate that the secret bit rate does in fact increase proportionally to an increase in λ, and that physical shaking can improve the rate by a factor of $3 \sim 4$. We also see that the error rate from list encoding remains fairly constant irrespective of the frequency or the rate.

Temporal correlation

Next, we estimate the temporal correlation in the fading processes using the relationship between level crossing rate and the coherence time, given in (4.2). Coherence time is an important parameter as it tells us how often the channel can provide fresh secret random bits. While ambient scaterrers and reflectors between a distant source and the receivers contribute to the natural rate of temporal variation, it is possible to *create* an increased rate of random temporal variation in the channels by moving or shaking Alice and Bob together. To investigate how much of an increase in temporal variation physical movement can provide, we collected channel measurements at Alice and Bob while shaking them together, keeping the distance between them fixed. For each type of signal (TV and FM), we physically waved Alice's and Bob's antennas together in the air, first slowly, and then vigorously. Figures 4.6(c,d) show the coherence time for each case. We find that waving the legitimate devices together vigorously can improve the rate at which secret bits can be extracted by a factor or 2 to 2.5. An advantage of our method over the shaking-based secure pairing proposed by [61, 62], is that an onlooker cannot predict the channel estimates made by the legitimate devices.

4.5.2 Monitoring multiple sources

Our analysis thus far has only considered a single public source, Peter. However, by simultaneously monitoring multiple RF sources, Alice and Bob can increase the amount of common randomness available to them per unit time. In fact, the number of achievable secret bits per second scales linearly with the number of added sources, so long as the sources themselves are physically separated by roughly $\lambda/2$ distance. We focus on FM broadcast transmissions, since the narrow bandwidth of each FM radio station (200 KHz) allows us to use the USRP platform to receive multiple FM stations simultaneously. In such a scenario, Alice

and Bob can treat each signal source as an *independent* source of common randomness and run parallel instances of our secret bit extraction algorithm. With a bandwidth of 200 kHz per FM station, the maximum number of FM sources that Alice and Bob can receive with a single USRP setup is 40 (due to throughput limitations of USB2.0). While not all channels in the FM band carry an FM radio station with a strong signal, there are few regions in the country without adequate FM radio coverage. Using our GnuRadio/USRP platform, we sampled the channel responses for five radio stations within 1 MHz of block of spectrum (at 97.9, 98.3, 98.7, 99.1, and 99.5 MHz).

To test for independence of the fading processes from different FM stations, we examine the correlation between Alice's channel estimates at a given instant of time, across three of these FM stations. Figure 4.8 illustrates this as pair-wise scatter plots between Alice's channel estimates for the three stations along with corresponding correlation coefficients for each pair of stations. This is compared with the correlation between the estimates of Alice and Bob on a given FM station. Our results suggest that temporal variations in the channels from different FM radio stations are in fact fairly independent. We ran our list-encoding algorithm on the 5 FM channels taken as parallel, independent sources of randomness, and observed a combined average secret bit rate of 1.08 bps, with an average error rate of 0.039 when Alice and Bob were stationary and 0.1λ apart, and 4.27 bps with an average error rate of 0.042 when Alice and Bob were waved together in unison. When compared to the results of a single FM channel, it is clear that sampling the abundance of public sources provides a simple means to more rapidly establish secret bits.

4.5.3 Coping with an adversarial source

In this section, we consider an active Eve, controlling Peter's transmitted signal in an attempt to influence the key bit generated by Alice and Bob. We will assume

Environment	TV	FM
Stationary	$(0.81, 0.037)$	$(0.22, 0.029)$
Moved slowly	$(2.4, 0.038)$	$(0.60, 0.042)$
Moved fast	$(3.5, 0.041)$	$(0.83, 0.032)$

Table 4.1: (Average secret-bits per sec., average error-rate) pairs for $d = 0.1\lambda$ using the TV signal at 584.31 MHz and a single FM radio station at 88.7 MHz when employing list-encoding.

that the environment remains multipath rich. If Alice and Bob use temporal variations in the *amplitude* of the channel variations to extract secret bits, then Eve can modulate her transmit amplitude to *create* perceived fades in amplitude and thus influence the bits extracted by Alice and Bob. However, Alice and Bob have another option— changes in the channel's *phase* as the source of common randomness for extracting secret bits, which we left unexplored thus far. The phase added by the channel, and hence the phase of the signal received at Alice & Bob is not observable by Eve since she does not have control over the positions or movements of the numerous scatterrers or those of Alice and Bob.

However, instantaneous phase alone, cannot be directly used for extracting bits at Alice and Bob because the measurement of phase also depends upon the phase of the local oscillators (LO) at both the transmitter and the receiver. Since it is not pragmatic to assume phase synchronization between the LOs of Alice and Bob, we use the *change in phase* instead of the actual value of the phase. With this modification, we can still design a system in which secret bits can be extracted, even when the adversary controls the transmitter and can modulate the transmit signal arbitrarily, as we explain below.

Without loss of generality, let us assume that a public source transmits a pilot tone $\tilde{s}(t) = A\cos(2\pi f_c t)$, represented as a baseband equivalent by $s(t) = A$. That is, its bandpass representation is $\tilde{s}(t) = Re\{s(t) \cdot e^{j2\pi f_c t}\}$. Let the overall multipath fading channel be represented by a time varying phasor $h(t) = H(t) \cdot$

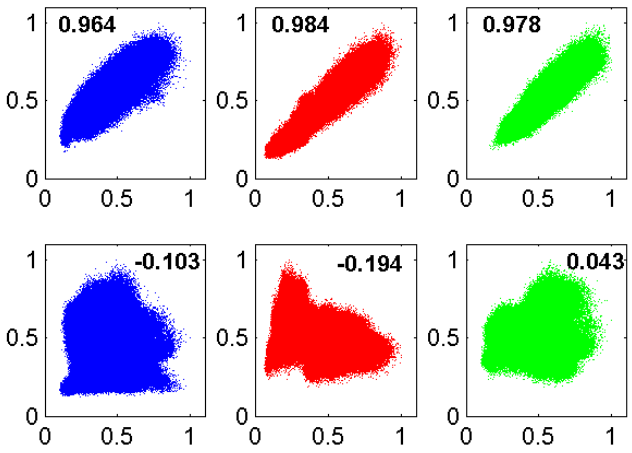


Figure 4.8: *Top row:* Channel measurements of Alice versus those of Bob on three different FM radio stations. *Bottom row:* Measurements of Alice on the three FM radio stations, taken two at a time. In each case the channel variations are normalized by the maximum value of the channel estimate during the measurement, and the linear correlation coefficient is provided.

$e^{j\theta(t)}$. Ignoring noise, the received signal is

$$r(t) = s(t) \cdot h(t) = AH(t)e^{j(2\pi f_c t + \theta(t))}. \tag{4.3}$$

The baseband equivalent of the above signal is ideally obtained by the receiver by using a sinusoid from the local oscillator at frequency f_c, giving the phasor $AH(t)e^{j\theta(t)}$ at time t. In the baseband representation, we may represent the relationship (4.3) between the received and transmit signals and the channel simply as $r = Ah$, where A is the amplitude of the transmitted sinusoid and $h = H(t)e^{\theta(t)}$ is the complex baseband channel. The channel estimate is then given simply by $\hat{h} = r/A$.

Suppose the source is controlled by Eve, who modulates both the magnitude and the phase, and thus transmits $\tilde{s}_E(t) = A(t)\cos(2\pi f_c t + \phi(t))$ with a complex representation $s_E(t) = A(t) \cdot e^{j\phi(t)}$. The receiver's channel estimate is

$$\hat{h}(t) = A(t)h(t)e^{j\phi(t)} \tag{4.4}$$

The adversary has inserted a multiplicative factor of $A(t)e^{j\phi(t)}$ in front of the true

channel state $h(t) = H(t)e^{j\theta(t)}$, causing the channel to appear to have magnitude of $A(t)H(t)$ and a phase of $\phi(t) + \theta(t)$. Observe that the product of amplitudes $A(t)H(t)$ behaves very differently from the sum of the phases $\theta(t) + \phi(t)$. In particular, while Eve has non-trivial information about the magnitude $A(t)H(t)$ (i.e. mutual information $I(A(t); A(t)H(t)) > 0$) by virtue of being able to control $A(t)$, she has no information about the received phase $\theta(t) + \phi(t)$ (i.e. $I(\phi(t); [\theta(t) + \phi(t)] \text{mod} 2\pi) \approx 0$) because the phase *wraps around* modulo 2π. Therefore, while the adversary can affect the estimated amplitude of the channel by using a smaller transmit amplitude $A(t)$ and affect the resulting bits, the phase of the received signal cannot be controlled by the adversary. Hence if each legitimate user samples the phase once per coherence time[2], the resulting phase samples are both correlated at the two users, as well as independent from one sample to the next, irrespective of the function $\phi(t)$ used by Eve. Thus, differential-phase, measured across successive coherence times, can be used to extract secret bits, even if the transmitter is controlled by the adversary.

We evaluated our differential phase method for extracting secret bits by creating our own FM source (Eve) using the USRP/GnuRadio platform. To avoid interference from real radio stations, we transmitted our fake radio station indoors at 315MHz. In order to also show that our differential phase method can be combined with physical movement, Alice and Bob were located in a different room $d = 0.04\lambda = 1.5$ inches apart and vigorously shaken. Alice's and Bob's USRP/GnuRadio platforms performed a standard frequency recovery algorithm to track the carrier frequency for baseband shifting. The signal phase was estimated and used to calculate the differential phase. We compared Alice's and Bob's differential phase against Eve's known transmit signal's differential phase.

[2]It is important that the estimate of channel coherence time be based on an a conservative and a priori knowledge of a typical coherence time, rather than based on channel estimates, as Eve can rapidly vary $A(t), \phi(t)$ to fool the users into severely underestimating the coherence time.

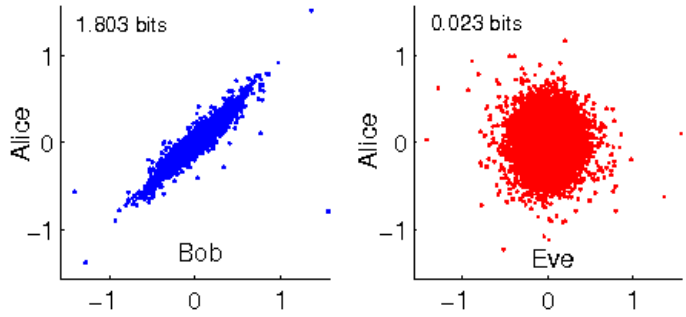


Figure 4.9: Differential phase measurements of Alice vs Bob (left), compared with those of Alice vs Eve (right). The axes represent the interval $[-\pi, \pi]$ radians. Each plot shows a mutual information estimate between the differential phases of the corresponding two users, computed using the algorithm in [2].

The change in phase from one measurement epoch (channel coherence time) to the next served as the source of common randomness between Alice and Bob.

Figure 4.9 shows a scatter diagram of the differential phase of Alice's received signal versus Bob's, and a scatter diagram of Alice's differential phase versus the differential phase in Eve's transmit signal. The plot also shows an estimate of the mutual information in bits per differential-phase measurement, for the two comparisons. These results indicate that differential phase in the received signals at Alice and Bob are highly correlated, while nearly independent of Eve's differential transmit phase. Therefore, Eve has no useful information about the differential phases at Alice and Bob.

We then extracted secret bits by quantizing the differential phases at Alice and Bob using an equiprobable one-bit quantizer, with a boundary at 0 radians. Alice and Bob thus obtain one bit per coherence time. We conservatively chose a coherence time of 200msec. The actual coherence time was computed to be 132msec, which is a consequence of using 315MHz and vigorously shaking Alice and Bob. Ultimately, Alice and Bob had a secret bit rate of 5bps, with an error rate of 0.090. Eve's corresponding best estimate of Alice's bits had a

corresponding error rate of 0.4933 relative to Alice's bits.

4.6 Security discussion

In this section we discuss the security aspects of Clique and examine limitations of the technique.

Spatial security. Beyond theoretical models, our experimental evaluation in Section 4.5.1 suggests that if an adversary is greater than 0.4λ away from both Alice and Bob, then the fraction of bits produced by the adversary that are different from those produced by Alice or Bob is very close to 0.5. For an FM radio signal, 0.4λ corresponds to $\sim 1 - 1.3$ m. Mutual information is a particularly convenient metric that helps us upper bound the amount of relevant information available to Eve about Alice's & Bob's measurements. Empirically computed estimates of the mutual information between the measurements of each pair of users (see for e.g. Figure 4.7) show that the information that a sufficiently distant Eve has about Alice's or Bob's observations is more than an order of magnitude lower than the information either legitimate user has about the other's observations. This implies that for each sequence of 100 bits generated by Alice & Bob, Eve can guess *at most* 1 bit.

Further, an increasing number of attacks today are conducted via an adversary placing a listening/reading device on or right next to a target (e.g. card swipes/RFID readers, skimmers on ATM machines). An Eve that is within the prescribed 0.4λ from Alice or Bob would be able to attack Clique easily. One exception to this is if Alice and Bob are mobile devices and can be shaken together – this induces random temporal variations into the obsevations of Alice and Bob, which cannot be matched by Eve unless she also moves in unision.

Active adversary. An active adversary may transmit a signal in an attempt to influence Alice's and Bob's estimates of their channels from Peter. If Alice and

Bob use only the magnitude of their received signal as a measure of the temporal variation of their respective channels, this results in a valid attack. Such an attack can be guarded against if Alice and Bob use differential-phase to generate a key, as explained in Section 4.5.3. This is because, Eve cannot control the phase of the resultant received signal at Alice & Bob, unless she has precise knowledge of all the reflectors and scatterrers in the environment and the various paths taken by the signal [68]. Precisely manipulating the phase characteristics of a multipath-rich environment is very hard.

Passive manipulations and modeling. If an attacker can manipulate the wireless environment by controlling the scatterrers and reflectors in the path of the signal between the public source and Alice/Bob, she can (in principle) influence the temporal variations observed by Alice and Bob. This is hard to accomplish when the public source is distant, as is likely to be the case with public broadcast sources, and would require a significantly powerful adversary. Second, the method in Section 4.5.3 based on differential-phase, makes this attack implausible for the same reason as the active attack above. What if a passive adversary that attempts to model the temporal variation in the channels between the public source and Alice/Bob by placing herself at the location of Alice/Bob, before or after the key extraction protocol? Such an adversary cannot get any useful information about the bits extracted by Alice and Bob because the wireless channel decorrelates over time, and channel models are only a statistical description of the channel, whereas Alice and Bob utilize features in a specific *instantiation* of the random processes formed by the temporal channel variations. Indeed, a model of wireless channel observed by Alice and Bob can in fact be given to the adversary a priori.

Statistical randomness of extracted bits. The generated key needs to consist of statistically independent bits in order to be used as a cryptographic key. This is ensured in Clique by having Alice wait for at least one coherence time interval after extracting a bit, before attempting the next bit extraction.

As explained in Section 4.3.1, the coherence time, is by definition, the amount of time needed for the channel state to become random after an observation, and is estimated in Clique by Alice using an empirical estimate of the level crossing rate. It is interesting to note that in order to truly estimate the per-bit entropy of a sequence of bits, a very large number of bits ($\gtrsim 10^7$) is required, so that the results of statistical tests such as those in [44, 46] can be meaningful.

4.7 Concluding remarks

This chapter shows how wireless devices in proximity can form secure associations autonomously by monitoring public sources of airwaves and using their correlated fading processes to form a shared cryptographic key. The speed with which users can securely pair depends on their physical separation, and on the rate of temporal variation in the observed fading processes. Pairing can be accelerated by monitoring multiple public sources concurrently, or by manually shaking the legitimate devices together. Using differential-phase in place of amplitude variations for extracting secret bits proves to be particularly robust against active attacks. The key geenrated by Clique can also allow devices to authenticate each others' proximity *autonomously*. By picking a public source of a suitable wavelength, a device can ascertain whether a device that it is communicating with is really within the physical distance it claims to be. We expect this will be immensely useful in preventing some types of spoofing attacks, and in situations in which devices need to communicate without human involvement.

Clique need not be restricted to two devices and can be easily extended to allow multiple devices to establish a common secure association. We believe that a number of useful optimizations of our system can improve its functionality and performance even further. For example, if Alice and Bob do not end up with identical bits after using an error correcting code, they do not need to discard

their bits and start over, but can instead employ further rounds of error correction in a manner similar to iterative reconciliation protocols used in quantum cryptography. Another direction for improvement, noted earlier, would be to explore the use of non-binary quantization for improved extraction rates.

Chapter 5

Traffic privacy in packet-size side channels

In this chapter, we explore a specific type of data privacy problem in wireless networks. By virtue of being a broadcast medium, data traffic on a wireless link can be easily overheard by eavesdroppers, often from very large distances. Even if the data being transmitted is encrypted, there are often traces of semantic information that are present in the broadcast information, that are not entirely independent of the encrypted data being transmitted. One such source of semantic information is packet size. As we wll explain, the size of a packet can often reveal quite a bit of information about the packet's contents and its context, even if the packet is completely encrypted. This is the case with ecryption algorithms that preserve the length of a packet upon encryption. Packet size information forms a type of side-channel, leaking unintended information to a passive adversary. In the following sections, we first introduce this problem in greater detail and then build a formal framework for building a mechanism that can stop information from leaking out from packet sizes. We include our results to date about the problem, and finally we list the open issues that remain to be addressed.

5.1 Introduction

The successful adoption of personal communication technologies is leading to an increase in the amount of sensitive or private information being exchanged. Sensitive information, such as a personal phone call or financial transactions, cross our networks everyday and it is very easy for adversaries on the network

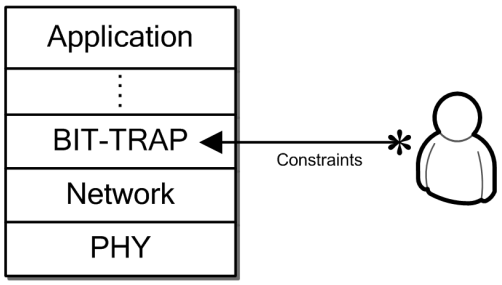


Figure 5.1: We envision a $\mathsf{Bit-Trap}$ as a separate layer that modifies data traffic handed down to it so that given a set of constraints on available resources, it optimally destroys side-channels from leaking out any information.

to monitor this traffic, especially when a wireless link is involved. Although a first line of defense to ensuring confidentiality involves encrypting the packets prior to transmission and forwarding over a network, recent evidence has shown that higher layer encryption alone is insufficient to providing the strict guarantees of privacy that most users expect for their transactions. In numerous different application scenarios, it has been shown that basic traffic analysis involving the statistical analysis of packet sizes and inter-arrival times can identify significant levels of contextual information in spite of the underlying session being *properly* encrypted.

Adversaries armed with knowledge of the application and the underlying network functions, yet no knowledge of the cryptographic material associated with security primitives, have been able to identify a surprising amount of information, including the language and specific phrases spoken in encrypted voice-over-IP calls [11, 10], or the identity of a video clip [70], or even the number of participants in an encrypted broadcast service [71]. The common confidentiality failure across all of these examples extends from the fact that secrecy is reliant on more than just protecting individual bits, but requires protecting the context of such bits. Messages typically do not occur in isolation, but instead occur together as part of a larger application. By analyzing the traffic in the context a specific application, in all cases it was possible to identify an underlying correlative structure that had

very little to do with the bits themselves, and which was able to reveal significant (though not all) information about the actual messages being exchanged.

In order to prevent such traffic analysis attacks, it is necessary to re-examine how the traffic is loaded onto the network prior to communication. Strategies, such as the introduction of *phantom* users[71], padding messages to uniform size [11], buffering[72, 73, 74] and re-encryption of packets[75] have been presented as techniques to obfuscate the true traffic pattern associated with an application. Although these methods represent a powerful set of tools for hiding the true traffic, and hence enhancing confidentiality of encrypted packet streams, there has been very little work in exploring the fundamental tradeoffs associated with such tools.

In this chapter, we cast the problem of protecting the contextual privacy associated with encrypted packet streams in an information theoretical setting, where the objective is to obfuscate the meaning of these packet streams by randomly padding packets or changing their sizes by buffering them. Starting from an assumption that packets can be broken into smaller chunks (for the sake of discussion, we'll assume the minimal indivisible unit is a bit), padded, and buffered, we seek to minimize the *mutual information* [3] between an incoming packet stream and an outgoing packet stream, while maintaining desirable constraints (e.g. delay and average bits spent on padding). We assume that there is some fixed cost (in bits per packet) for obfuscating packet sizes, perhaps in the form of protocol bits that tells the destination how to re-assemble the original packets. We do not deal with the details of this mechanism, so as to focus on the tradeoffs between the level of obfuscation achieved and the amount of resources required. We envision $\mathsf{BIT-TRAP}$ as a layer as depicted in Figure 5.1 that sanitizes packetized traffic being handed down to it, so as to remove the existence of a leaking side-channel as much as possible, given a set of user-specified constraints on the available resources that $\mathsf{BIT-TRAP}$ can spend. This chapter ofthe thesis takes

steps in this direction by quantifying the relationship between resources and the amount of leakage possible.

Through our analysis, we uncover several useful insights that can guide practical traffic analysis countermeasures:

- In a *single packet model*, random padding of packets is the best way to achieve privacy against traffic analysis, and given a fixed average padding budget, an optimal method for random padding can be quickly found via a convex program. We also prove that the space of padded packets need not be different from the space of true packet sizes and that the trade-off between the achieved obfuscation and the amount of padding allowed is a convex rate-distortion type relationship.

- For *streams of packets*, it is the correlation between successive packet sizes, rather than actual packet sizes, that is responsible for vulnerability to traffic analysis. We formulate the problem of obfuscating the information in correlated streams of packets as a queuing problem and show how a 2-dimensional Markov chain can be used to relate the buffer delay and the mutual information for various obfuscation policies.

5.2 Related Work

The subject of traffic privacy has a rich history and prior literature. These can be broadly classified into attacks and defenses dealing with: (i) making inferences about the source and/or origin of communications in a network, and (ii) gleaning contextual information about the information content itself. It is the latter with which we are concerned in this chapter of the thesis.

Privacy issues in the former model have been examined in general networks, particularly through the methods of anonymous communications. Chaum proposed a model to provide anonymity against an adversary conducting traffic

analysis[73]. His solution employs a series of intermediate systems called mixes. Each mix accepts fixed length messages from multiple sources and performs one or more transformations on them, before forwarding them in a random order. Most of the early mix related research was done on *pool mixes*[74], which wait until a certain threshold number of packets arrive before taking any mixing action. Kesdogan [76] proposed a new type of mix, *SG-Mix*, which delays an individual incoming message according to an exponential distribution before forwarding them on. Later, Danezis proved in [77] using information theory that a SG-Mix is the optimal mix strategy that maximizes anonymity. The objective of SG-Mixes, however, is to decorrelate the input-output traffic relationships at an individual node, and the methods employed do not extend to networks of queues.

Work in the latter model has focused much more heavily on attacks than on defenses. Attacks have ranged from inferring the spoken language and specific phrases in encrypted VoIP streams [11, 10] from the sizes of packets in a stream, to inferring contextual information from key strokes in encrypted SSH sessions [78] and the identity of video clips [70] from the variable timing of packets on a network connection. One reason for the greater attention given to attacks is that these works most often suggest a supposedly 'simple' guarding measure: regularize the quantity that leaks out information. In the above examples, this would translate to making all packets the same length and making intern-packet durations the same for all packets. While expending sufficient resources on the problem (extra bits, delay, etc.) can eliminate a known side-channel, the amount of extra resources that might be needed to remove such side-channels may be non-trivial. In this chapter of the thesis, we formally study the problem of side-channel leaking out information, in the context of variable packet sizes. In particular, we show that there often exist optimum solutions when resources to be expended are limited. Finally, we note that our work has some connections with the study of timing-capacity of queues [79], in which information is encoded in the intervals

between packets arriving at a queue. However, variable packet sizes and inter packet duration are separate sources of randomness - we specifically used a model with one packet per slot (including zero-sized 'packets') so as to focus on the information contained in the packet sizes and not their timing.

5.3 Notation

We will denote the sizes of packets found in a network or a packet stream by $\mathcal{A} = \{A_1, \ldots, A_M\}$, where $\mathcal{A}$ is the set of all packet sizes, and $A_1 < A_2 < \cdots A_M$ are the possible sizes of the packets. We will denote the probabilities of occurrence of the M possible packet sizes by the probability mass function $\mathcal{P} = \{p_1, \ldots, p_M\}$, where $\sum_i p_i = 1$. The packet sizes, after being modified by an obfuscator are denoted by $\mathcal{D} = \{D_1, \ldots, D_N\}$, where we will assume with loss of generality that $D_1 < D_2 \ldots < D_N$, where N need not equal M. The letters A and D are chosen to denote *arrivals* and *departures* respective, where the arrivals and departures are with respect to an obfuscator, which may add extra bits and/or delay to the packets. We will use the letters A and D to denote the random variable associated with the true size of a packet and modified size of a packet respectively. We use the notation y^+ to denote $max(y, 0)$.

We will use the terms obfuscator, obfuscation channel and obfuscation system interchangeably, to refer to the operator that we are interested in designing, responsible for preventing the leakage of information through the packet-size side channel.

With slight abuse of notation, in the section on packet streams, we will use A_i and D_i to denote the true and modified size of the i^{th} packet in a packet stream respectively, and A^n & D^n to denote the sequences $A_1, A_2, \ldots, A_n$ and $D_1, D_2, \ldots, D_n$. Lower case letters a and d denote the realization of a true packet size and modified packet size respectively. Likewise, a_i and d_i denote a specific

realization of the true size and modified size of the i^{th} packet in a sequence of packets respectively. We will use $\{G\}$ to represent the random process formed by successive realizations of a random variable G.

5.4 Single packets

Consider a sensor network which monitors several types of events. As each type of event is triggered or observed by a node in the network, it records or generates data corresponding to that event and send it in the form of a messages towards the sink. Let the set of all possible events in the network be denoted by $\mathcal{E} = \{E_1, \ldots, E_M\}$ with probabilities of occurrence $\mathcal{P} = \{p_1, \ldots, p_M\}$ and message sizes $\mathcal{A} = \{A_1, \ldots, A_M\}$. Consider an adversary observing traffic at an opportune point in the network. The adversary can attempt to infer which event has occurred simply by looking at the size of the message it intercepts over the air. We need a formal metric to quantify the amount of uncertainty in the attackers inference, given the observations available to her. If an adversary observes a packet of size s, the amount of uncertainty in the adversary's inference about the event in $\mathcal{E}$ given her observation of s is captured by Shannon's entropy function:

$$H(\mathcal{E}|s) = -\sum p(E_i|s) \log p(E_i|s)$$

A network designer's goal would be to maximize this conditional entropy by altering the sizes of packets transmitted after each event. Let these altered packet sizes be reflected as $\mathcal{D} = \{D_1, \ldots, D_n\}$. The problem of providing traffic privacy then becomes: Maximize $\sum_{j=1}^{n} H(E_j|s)$ while minimizing the expected communication overhead due to the privacy protection methods. There are two broad strategies to improve the privacy in this situation:

Constant packet size: The key idea is to attempt to make all packets in the network have the same size. It requires padding all packet to the size of the largest packet.

Randomized packet sizes: This approach relies on randomizing the size of every single packet to create an uncertainty about the type of packet and underlying event.

Let us look at these two strategies in some detail.

Fixed vs. randomized obfuscation. The suggested solution for preventing analysis of packet sizes in much of the prior work highlighting attacks, is padding packets to all have the same size. This strategy, though, is only satisfactory when the variation in sizes across packets is small. However, when the variation is large, padding packets can significantly increase the usage of bandwidth. Furthermore, for scenarios in which nodes have limited energy, such as wireless sensor networks, this increase in traffic can lead to an unacceptable increase in energy consumption. In general, we may obfuscate the size of a packet by making it larger (padding bits), or by splitting it into multiple packets. This latter approach is not beneficial for sporadic communication, when the network only infrequently transmits a single packet, since an adversary can simply add up packet sizes and subtract known header sizes to arrive at the true packet size. Hence, we believe that padding packets is the most general and promising recourse for preventing traffic analysis when dealing with single packets or a collection of a few packets rather than a long stream.

To begin, we will investigate whether it is possible to achieve obfuscation without having to pad all packet to the same size. Suppose that we have M different types of packets traversing a network with packet sizes given by $\mathcal{A}$ above, and a priori probabilities of occurrence given by $\mathcal{P}$ above. Further, suppose we have an average padding budget of B bits. Our task is to pad the packets randomly so as to achieve the greatest obfuscation of the true packet lengths subject to our padding budget. Let $A \in \mathcal{A}$ be a random variable that denotes the true packet size and $D \in \mathcal{D}$ denote the size of the packet after it has been

padded. Then $D = A + Z$, where $Z \in \mathcal{I}^+$ denotes the number of padding bits added. All packet sizes, $\mathcal{A}$ and $\mathcal{D}$ are integers. The following result declare that there exists a uniquely *optimal* obfuscation strategy:

Theorem 1: Maximizing obfuscation using randomized packet padding of single packets with a bound on the given average padding budget is a convex optimization problem.

Proof: We assume that a maximum packet size of D_{max} is allowable in the network (where $D_{max} \geq A_M$) and that the set $\mathcal{D}$ of packets that can be observed after padding are $\mathcal{D} = \{A_1, A_1 + 1, A_1 + 2, \ldots, D_{max}\}$, with D_k being the k^{th} element of this ordered array. Let P_{ij} denote the probability that a packet of size $A_i \in \mathcal{A}$ is padded to a packet of size D_j and $\mathbf{P}$ denote the $M \times |\mathcal{D}|$ matrix of these probabilities, where $|\mathcal{D}| = D_{max} - A_1 + 1$. Our objective may be stated to be simply

$$\max_{\mathbf{P(D|A)}} H(A|D) \tag{5.1}$$

where the variables of optimization are the the contents of the *transition probability matrix* (henceforth, TPM) $\mathbf{P}$ and $H(A|D)$ denotes the average uncertainty about A upon having observed D, averaged across all packets. Note that $P_{ij} = 0$ whenever $A_i > D_j$. The constraints of the problem can be formally stated as follows:

$$\text{Cost} = E_i[Z] = \sum_{i=1}^{M} \sum_{j=1}^{|\mathcal{D}|} P_{ij}(D_j - A_i)p_i \leq B \tag{5.2}$$

$$\sum_j P_{ij} = 1 \quad \text{for all} \quad i = 1, \ldots, M \tag{5.3}$$

$$P_{ij} = 0 \quad \text{for all} \quad i, j \text{ such that } D_j < A_i. \tag{5.4}$$

This can be recognized as a type of discrete memoryless channel (DMC) with input alphabet $\mathcal{A}$ and output alphabet $\mathcal{D}$. The DMC probabilistically adds bits to the input packets, transforming A to D and is represented by $\mathbf{P}$, whose elements P_{ij} are the probabilities that a packet of size A_i is converted to a packet of size D_j.

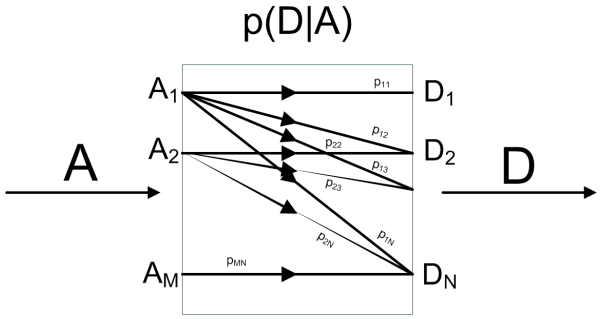


Figure 5.2: The discrete memoryless channel (DMC) [3] representing randomized packet padding. Each input packet is mapped to a packet of equal or greater size in accordance with a transition probability matrix $p(D|A)$.

We wish to determine the worst possible DMC so that an adversary observing its output learns as little as possible about the input, given an average constraint on the additive noise, $E[Z]$.

Observe that the a priori packet probabilities is fixed and that P_{ij} are the variables. Hence, maximizing $H(A|D)$ is equivalent to minimizing the mutual information $I(A;D)$. This quantity is convex in the transition probabilities P_{ij} for a discrete memoryless channel, when the input distribution is fixed (Theorem 2.7.4 of [3]). Therefore, we have a convex objective function (5.1), with constraints (5.2)-(5.4) that are linear in P_{ij}. Thus, this is a convex optimization problem with a unique solution that can be computed in time that is polynomial in the number of variables using known convex programming algorithms (e.g. gradient search, etc). ■

The optimization variables are the elements P_{ij} of the matrix $\mathbf{P}$. We note that while the size of the input alphabet $|\mathcal{A}|$ is fixed, the size of the output alphabet $|\mathcal{D}|$ is up to us as the system designer. Given that the problem above is convex, we can re-frame the problem as follows to compute the complete trade-off boundary between the bit padding budget B and the largest achievable obfuscation:

$$\text{Max}_{\mathbf{P}} H(A|D) + \lambda \left(\sum_{i=1}^{M} \sum_{j=1}^{|\mathcal{D}|} P_{ij}(D_j - A_i)p_i - B \right) \tag{5.5}$$

such that

$$\sum_j P_{ij} = 1 \text{ for all } i = 1, \ldots, N \tag{5.6}$$

$$P_{ij} = 0 \text{ for all } i, j \text{ for which } D_j < A_i. \tag{5.7}$$

Here λ is a Lagrange multiplier that controls the trade-off between the opposing objectives of maximizing entropy and minimizing padding cost. A given value of λ can be plugged into the problem and solved to compute the (entropy, cost) pair and the transition probabilities required to achieve it. At the end of this section, we show that the trade-off between obfuscation and the average bit-padding cost can be interpreted as a rate-distortion function and that obfuscation is convex and decreasing in the bit-padding budget B.

Algorithms for solving a convex optimization problem have running times that are polynomial in the number of variables. Although we have shown that the problem of finding the optimal random padding of packets is convex, the size of the set of variables, namely $\mathbf{P}$, can be very large if we allow for all integer values between A_1 and D_{max} in the output alphabet $\mathcal{D}$. We now show that in any optimal solution, the output alphabet need not be larger than the input alphabet. This result is significant because it shows that the problem is not only convex but practically solvable since it can be simplified to have only a fairly limited number of variables.

Theorem 2: The output alphabet $\mathcal{D}$ in the optimal allocation can be the same as the input alphabet $\mathcal{A}$.

Proof: The proof is based on the concavity of the entropy function and the structure of our cost function. Let D_k be any letter in $\mathcal{D}$ with non zero probability of occurrence such that $D_k \notin \mathcal{A}$ and $A_1 < A_i < D_k < A_{i+1}$, where A_1, A_i and A_{i+1} are letters in the input alphabet $\mathcal{A}$ and have corresponding same-size letters in the output alphabet $D_1 = A_1, D_i = A_i$ and $D_{i+1} = A_{i+1}$. Our objective is to

maximize the equivocation

$$H(A|D) = \sum_{d \in \mathcal{D}} H(A|D = d)p(D = d) \tag{5.8}$$

We will prove the above theorem by showing that the equivocation can only increase when any letter of the type $D_k \notin \mathcal{A}$ is eliminated, without increasing the average cost. To begin, assume that the input distribution $p(A)$ is fixed and an optimal allocation of conditional probabilities $p(D|A)$ has been found for the chosen output alphabet containing D_k. Consider the effect of eliminating the letter D_k from the output alphabet by diverting all the conditional probabilities $p(D = D_k|A)$ for all $A < D_i$ to the output letter D_i which is the next letter in $\mathcal{D}$ with packet-size smaller than D_k (i.e., if we initially allowed all integer sizes in $\mathcal{D}$, then $D_i = D_k - 1$). That is

$$p'(D_i|A_m) = p(D_i|A_m) + p(D_k|A_m) \text{ for all } m \leq i \tag{5.9}$$

where the prime denotes the new conditional probabilities. This action can only decrease the average cost

$$\sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} p(A = a)p(D = d|A = a)(d - a)$$

. Since this action only affects the terms in (5.8) pertaining to D_i and D_k, it is sufficient to prove that

$$\begin{aligned} H'(A|D = D_i) \cdot p'(D = D_i) \quad &\geq \\ H(A|D = D_i) \cdot p(D = D_i) \quad &+ \quad H(A|D = D_k) \cdot p(D = D_k) \end{aligned} \tag{5.10}$$

where the primes denote the new value of conditional entropy and probability. However, since we have $p(D = d) = \sum_{a \in \mathcal{A}} p(a)p(D = d|A = a)$, we have $p'(D = D_i) = p(D = D_i) + p(D = D_k)$. Dividing the inequality (5.10) by $p'(D = D_i)$ and denoting $\alpha = \frac{p(D=D_i)}{p'(D=D_i)}$, (5.10) can be re-written as

$$H'(A|D = D_i) \geq \alpha H(A|D = D_i) + (1 - \alpha)H(A|D = D_k) \tag{5.11}$$

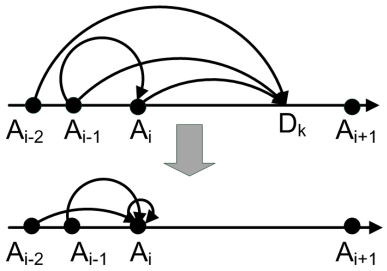


Figure 5.3: All transitions going to the output letter $y_k \notin \mathcal{A}$ are diverted to the next lower letter $y_i = x_i \in \mathcal{A}$. In the above figure transitions to $A_{i+1} = D_{i+1}$ are omitted for the sake of legibility.

The last inequality is the definition of a concave function. All that remains is to show that $p'(A|D = D_i)$ for the entropy term on the left hand side is an $(\alpha, 1-\alpha)$ linear combination of the conditional distributions $p(A|D = D_i)$ and $p(A|D = D_k)$ corresponding to the two terms on the right hand side. This can be shown using Bayes' rule:

$$p'(A = A_j|D = D_i) = \tag{5.12}$$

$$= \frac{p(A = A_j) \cdot [p(D = D_i|A = A_j) + p(D = D_k|A = A_k)]}{\sum_m p'(D = D_i|A = A_m) \cdot p(A = A_m)}$$

$$= p(A = A_j) \cdot \left(\begin{array}{c} \frac{p(D=D_i|A=A_j)}{p(D=D_i)} \cdot \alpha \\ + \frac{p(D=D_k|A=A_j)}{p(D=D_k)} \cdot (1-\alpha) \end{array} \right) \tag{5.13}$$

$$\alpha \cdot p(A = A_j|D = D_i) + (1-\alpha) \cdot p(A = A_j|D = D_k) \tag{5.14}$$

which proves that the new conditional distribution of $p'(A|D = D_i)$ after redirecting all the conditional probabilities from D_k to D_i is a $(\alpha, 1-\alpha)$ linear combination of the distributions $p(A|D = D_i)$ and $p(A|D = D_k)$. Since entropy is a concave function of the conditional probabilities, this proves (5.10). Since this procedure can be carried out for any such letter $D_k \notin \mathcal{A}$ without increasing the average cost, the result follows[1]. ■

[1]Note that any packet of the smallest packet size $A_1 \in \mathcal{A}$ must either be padded to the next larger packet $A_2 \in \mathcal{A}$ or be left unchanged, because padding to a length between a_1 and a_2 does not buy any equivocation but incurs a non-zero cost. Therefore the above proof is trivially true for any D_k in the interval $A_1 < D_k < A_2$.

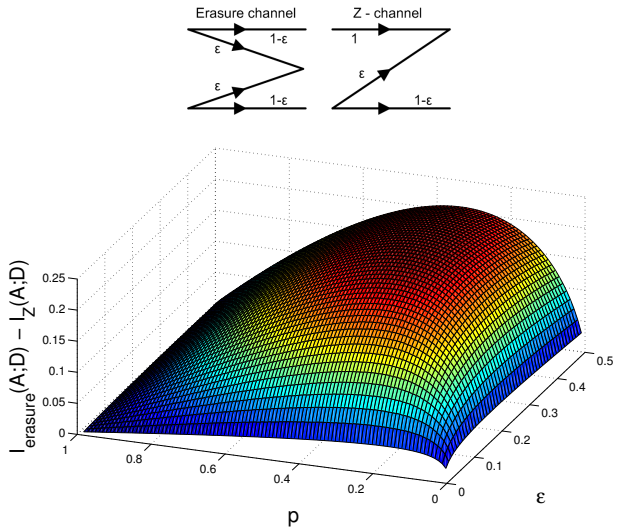


Figure 5.4: The Z-channel (top right) can be obtained from the ϵ-erasure channel (top left) by transferring the transitions to the erasure output in the erasure channel to one of the other outputs. This results in a lower mutual information across the the Z-channel for all combinations of the input distribution p and the parameter ϵ as witnessed by the plot of $I_{erasure}(A;D) - I_Z(A;D)$.

The above result is of practical significance because it allows us to use a matrix $\mathbf{P}$ of size $M \times M$, thereby limiting the number of variables to $\frac{M(M+1)}{2}$.

Corollary 1: The smallest possible mutual information is achieved if the output alphabet has size $|\mathcal{D}| = 1$.

This follows from the fact that if $|\mathcal{D}| = 1$, then output would have an entropy of $H(D) = 0$ and the mutual information, which is non-negative, is upper bounded by the output entropy. It also follows from the theorem above because the same argument as the one in the proof above can be repeatedly applied to each letter of the output alphabet, this time by transferring all conditional probabilities arriving at a given output letter to the next *larger* letter. However, this increases the padding overhead. In particular, if the constraint $E[Z] \leq B$ is not violated, the optimal solution is to pad all packets to the size of the largest packet.

The problem of determining the smallest mutual information R that the obfuscation channel can allow for a given padding budget B can be thought of as finding

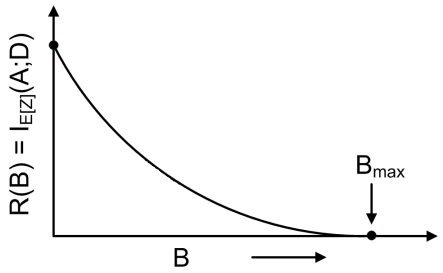


Figure 5.5: The trade-off between the amount of dummy bits and obfuscation achived is a convex rate-distortion function, with distortion being the amount of extra traffic. Zero distortion corresponds to no obfuscation, i.e. the mutual information equals the entropy of the packet sizes. At a certain value B_{max} of the padding budget, all packets can be padded to the size of the largest, providing perfect obfuscation.

the rate distortion function $R(B)$ with the allowable total average-padding B as the distortion measure and $R(B)$ being the smaller mutual information achievable with distortion of B or less. The rate distortion function is convex and goes to zero at a value of distortion B_{max} that is large enough for all packets to be padded to the size of the largest packet. The convexity of $R(B)$ can be proved as follows. Consider two conditional probability distributions, $p_1(D|A)$ and $p_2(D|A)$. For a given distribution of packet sizes, let these conditional distributions lead to distortions and rates of B_1, R_1 and B_2, R_1 respectively. Consider a conditional distribution $p_\alpha(D|A) = \alpha p_1(D|A) + (1-\alpha)p_2(D|A)$. This must lead to a distortion of $B_\alpha = \alpha B_1(y|x) + (1-\alpha)B_2(y|x)$, since the distortion $B = E[Z]$ is a linear function of the conditional distribution. Since mutual information is a convex function of the conditional distribution, $R_\alpha \leq \alpha R_1 + (1-\alpha)R_2$, which implies that the $R(B)$ curve is convex. Further, in a padding-only obfuscation channel, the only condition under which the mutual information can be brought to zero is if all packets are padded to the size of the largest packet. Otherwise, the occurrence of any packet at the output conveys the information that the input packet was of equal or smaller size.

Analytical solution to the padding problem. We will now try to solve this convex program analytically. Using the Lagrange multiplier method, we can form the Lagrange function

$$\begin{aligned}&\Lambda(P,\lambda,\underline{\gamma}) = \\ &\sum_l \sum_m p_l P_{lm} \left\{ \lambda(A_m - A_l) - \log\left[\frac{p_l P_{lm}}{\sum_k p_k P_{km}}\right]\right\} \\ &-\lambda B + \sum_i \gamma_i \left[\sum_{j\geq i} P_{ij} - 1\right]\end{aligned} \tag{5.15}$$

where P is the matrix with elements $P_{ij} = p(D = A_j | A = A_i)$. Here λ is the Lagrange multipliers associated with the equality constraints

$$\sum_i \sum_{j\geq i} p_i P_{ij}(A_m - A_l) = B$$

and γ_i are the Lagrange multipliers associated with the equality constraints

$$\sum_{j\geq i} P_{ij} = 1 \quad \text{for all} \quad j = 1, \ldots, M$$

The optimal solution can be found by using the conditions

$$\nabla_{P_{ij}} \Lambda = 0 \tag{5.16}$$

$$\nabla_{\lambda} \Lambda = 0 \tag{5.17}$$

Differentiating the Lagrange function with respect to P_{ij}, where $i \neq j$, the first condition (5.16) gives the equation

$$p_i \lambda(A_j - A_i) - p_i \log\left[\frac{p_i P_{ij}}{\sum_k p_k P_{kj}}\right] + \gamma_i = 0 \tag{5.18}$$

and differentiating with respect to P_{ii} gives

$$\gamma_i = p_i \log\left[\frac{p_i P_{ii}}{\sum_k p_k P_{ki}}\right] \tag{5.19}$$

Let us denote

$$\alpha_{ij} = \left[\frac{p_i P_{ij}}{\sum_k p_k P_{kj}}\right] = p(A = A_i | D = A_j)$$

Since α_{ij} is a conditional probability measure, therefore we have $\sum_{i \leq j} \alpha_{ij} = 1$. Eliminating γ_i using (5.19) in (5.18) gives

$$\alpha_{ij} = \alpha_{ii} \cdot e^{\lambda(A_j - A_i)} \tag{5.20}$$

Applying the fact that $\sum_{i \leq j} \alpha_{ij} = 1$ for all $j = 1, \ldots, M$ to (5.20), we get the relation

$$\alpha_{ij} = e^{\lambda(A_j - A_i)} - e^{\lambda(A_j - A_{i-1})} \tag{5.21}$$

which provides a way to compute all P_{ij} in terms of λ. We now have $\frac{M(M+1)}{2}$ equations in $\frac{M(M+1)}{2} + 1$ variables. We need only one more equation to eliminate λ and this is provided by applying the second condition (5.17) which simply gives the constraint

$$\sum_i \sum_j p_i P_{ij} (A_j - A_i) = B \tag{5.22}$$

The constraint (5.22) can be used to eliminate λ as follows. From (5.20), we have

$$\log \left\{ \frac{\alpha_{ij}}{\alpha_{ii}} \right\} = \lambda (A_j - A_i)$$

Multiplying both sides by $p_i P_{ij}$ and summing over i and j gives

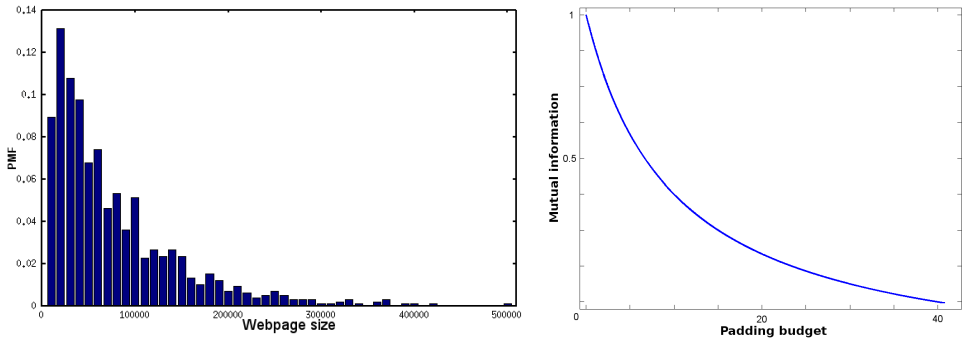
$$\begin{aligned} \sum_i \sum_j p_i P_{ij} \log \left\{ \frac{\alpha_{ij}}{\alpha_{ii}} \right\} &= \lambda \sum_i \sum_j p_i P_{ij} (A_j - A_i) \\ &= \lambda B \end{aligned}$$

Therefore,

$$\lambda = \frac{1}{B} \sum_i \sum_j p_i P_{ij} \log \left\{ \frac{\alpha_{ij}}{\alpha_{ii}} \right\}$$

We find that the solution to the problem of optimally allocating padding bits randomly to packets does not quite lend itself to a water-filling interpretation.

Example: Web-browsing traffic. As an example, we now consider the traffic produced by the HTTP protocol running on TCP/IP, while browsing webpages. A passive traffic analysis attack for identifying webpages based on the



(a) (b)

Figure 5.6: (a) The distribution of the sizes of the 1000 most visited webpages on the world wide web as of May 2010. (b) The rate-distortion function between mutual information and bit-padding budget for obfuscating the sizes of the 1000 most visited webpages on the world wide web. The x-axis is in units of 10^3 bytes, and the y-axis is normalized by 4.3 bits

number and sizes of packets sent by a web-server, has been considered in [80, 81]. When a browser requests a web-page from a webserver, the webpage's content is delivered through a sequence of packets that contain various objects in that page. For example, some of the packets may contain individual jpeg files, some may contain text and some others may contain a sound clip or a flash video clip. Even if these packets are encrypted, the sizes and number of packets of each size allow a way for a passive eavesdropper to fingerprint websites and later use this information to infer which website is being accessed by a user on a network. In some cases, the attacker may only be interested in narrowing down the set of possible webpages being accessed by a user, rather than pinpointing it. Random padding of packets, as studied in this section can make this task much harder for such an adversary.

Suppose a web-server returns IP packets of sizes $\{a_1, \ldots, a_n\}$ where multiple packets may have the same size. The order in which packets arrive cannot reliably be used as a fingerprint because packets can be re-ordered by the network itself.

This is neither a single packet, nor a long stream of packet, but rather an ensemble of packets. How can we best limit the inference that a passive adversary with a fingerprint database can make? the optimal solution involves considering the entire ensemble as a single packet, adding a random number of bytes, and re-fragmenting to produce packet of random sizes and numbers. In doing this we force the adversary to treat the total size of the ensemble as the only fingerprinting feature, because information about the sizes and numbers of packets returned by the HTTP request is lost. The problem is thus reduced to one with a single packet. As an example, we determine the total size of the HTTP response from the 1000 most popular webpages on the World Wide Web [82]. The result is shown in Figures 5.6(a). The long-tailed behavior exhibited by the PMF in Figure 5.6(a) is typical of packet sizes in networks and points to the fact that perfect obfuscation by padding all packets up to the size of the largest packet would be very expensive. We find that if web-page sizes are rounded off to the nearest $10,000$ bytes, then the sizes of these 1000 most popular webpages contains ~ 4.3 bits of entropy. Figure 5.6(b) shows the rate distortion function that gives the trade-off between the amount of extra bytes that must be added to achieve a given level of obfuscation. We find that with an average of just $7,343$ bytes of random padding, one can reduce the amount of entropy by a factor of 2, while complete obfuscation costs $42,070$ bytes on average, almost 6 times as much!

Example: Cost of obfuscating distributions. Consider a problem in which the adversary is interested in the distribution $p(A)$ of A in order to perform a classification task. What is the cost of transforming a distribution $p_1(A)$ to another distribution $p_2(A)$ by padding packets. It is easy to see that by padding packets alone, the distribution can only be changed to one whose mean packet size is larger than the mean packet size of the original distribution, and the cost is difference of the means $E_2[A] - E_1[A]$. Just like we have been obfuscating the true size of a single packet, we can obfuscate the distribution by random padding,

by randomly picking a distribution (with a larger mean) and transform the given distribution to the chosen distribution. However, it is not possible to morph any distribution $p_1(A)$ to any other distribution $p_2(A)$ by padding alone. This is because the size of each packet is either increased or stays the same. Therefore, in order to morph from $p_1(A)$ to $p_2(A)$, where $p_1(A)$ and $p_2(A)$ are vectors of the same length, $p_1(A)$ must majorize $p_2(A)$. We will see that this condition is not necessary for morphing if small amounts of delay are allowed in addition to padding.

5.5 Packet streams

We now study a stream of variable sized packets. We will find that the insights we have developed using the single packet model will prove to be useful in analyzing streams. A number of recent attacks on encrypted data streams [11, 10] have used machine classification techniques to infer useful information by employing the variability in packet sizes. To defend against such traffic analysis, we will examine padding as well as splitting packets into smaller packets and fusing packets together to form larger packets. We model a discrete time slotted stream of data packets (one packet per slot) and the manipulations carried out on it using an information-theoretic discrete channel as before. With slight abuse of notation with respect to the previous section, we will denote the stream to be obfuscated as a time ordered sequence A^N of random variables A_i, with $i = 1, \ldots, N$ representing time slots and A_i representing the size of the packet in the i^{th} slot, such that $A_i \in \mathcal{A}$. Similarly, the output D^N is a sequence of random variables $D_i, i = 1, \ldots, N$, such that $D_i \in \mathcal{D}$ for all i and $\mathcal{A}, \mathcal{D} \in \mathcal{I}^+$. We will use the notation G^N to refer to the sequence of random variables $G_1, \ldots G_N$, and $\{G\}$ to mean the random process formed by successive realizations of G.

As before, we will use the mutual information as a metric to quantify how

dissimilar the modified stream of packets $\{D\}$ is, compared to the original stream $\{A\}$. However, we will no longer be able to use the single-letter mutual information metric that we have used in Section 5.4; instead, we will develop a mutual information metric that is appropriate for streams, treating each stream as a stationary, discrete-time stochastic process.

Mutual information provides a general upper bound on the performance of any classification algorithm, and that it is one of the few metrics that can measure non-linear relationships between sequences (though there are many measures that quantify only linear relationships [83]). This is important for us as the redistribution of bits across packets can easily create a non-linear relationship between the input and output of an obfuscator with memory. This is the reason why single-letter mutual information is not appropriate for measuring the dependence between streams.

When the input stream A^N has a finite memory (i.e., there exists some k such that for all i, A_i and A_{i+k} are independent for all practical purposes), a data stream can be treated as a sequence of independent vector inputs to a discrete memoryless vector channel in a manner similar to that of Section 5.4. Obfuscating information in the packet-size side channel could then be accomplished by the following steps:

1. Collect k packets arriving sequentially and treat the sequence $\{A_1, \ldots, A_k\}$ as a single vector input to a vector discrete memoryless channel.

2. Map each input vector of length k to a vector of packets of lengths $\{D_1, \ldots, D_l\}$, where l may be different from k, such that $\sum_{i=1}^{k} A_i = \sum_{i=1}^{l} D_i$ (conservation of total information bits).

Since the above approach employs a discrete memoryless channel, it can be considered to be a straightforward extension of the method of Section 5.4 using vector inputs instead of scalars. However, it is not practical for two reasons. First, it

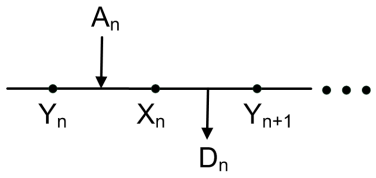


Figure 5.7: Timeline showing the relationship between the variables associated with a single slot.

requires the obfuscating vector-DMC to collect k packets from the stream before deciding the sequence of output packets, thereby introducing a fixed large delay in the stream, which may not be suitable for delay-sensitive applications. Secondly, it requires the computation of a large transition probability matrix to characterizes the vector-DMC that is of size $|\mathcal{A}|^k \times |\mathcal{D}|^j$.

An alternative approach that seeks to avoid the problems with a vector-DMC is that of a discrete channel with memory. Here, the obfuscating channel is defined in general by the conditional probability distribution $p(D_n|A^n, D^{n-1})$ wherein the output of the channel is explicitly allowed to depend upon past inputs and outputs. Since the input to the channel does not have a chance to observe the output, we say that the channel is used without feedback, i.e. therefore, $p(A_n|A^{n-1}, D^{n-1}) = p(A_n|A^{n-1})$. Our objective can be now be stated as

$$\min_{p(D_n|A^n, D^{n-1})} I\{(A\}; \{D\}) \tag{5.23}$$

where $I\{(A\}; \{D\})$ is the *mutual information rate* between the discrete time processes $\{A\}$ and $\{D\}$, defined as

$$\begin{aligned} I\{(A\}; \{D\}) &\triangleq \lim_{n\to\infty} \frac{1}{n} I(A^n; D^n) \\ &= \lim_{n\to\infty} \frac{1}{n} (H(A^n) - H(A^n|D^n)) \\ &= \lim_{n\to\infty} (H(A_n|A^{n-1}) - H(A_n|A^{n-1}, D^n)) \end{aligned}$$

The idea of splitting and fusing packets can be generalized by considering a buffer that allows us to store the remaining bits of a packet that has been split (thereby

allowing $D_n < A_n$) as a first-in-first-out queue. The queue is modeled using an internal state variable whose value Y_n at time instant n denotes the number of bits contained in the buffer just before the n^{th} arrival, A_n. Thus, delay can be used as a resource for obfuscating the stream $\{A\}$. It can now be seen why mutual information rate is an appropriate metric - the non-linearity introduced by the queue cannot be captured by a metric that only measures linear relationships. Note that in general, a packet stream (non necessarily discrete-time) may contain information in the packet sizes as well as in the inter-packet timing. This is true for discrete-time streams as well. For example, variable periods of silence between spoken words (zero-sized packets) can convey information about the language being spoken. We accommodate this phenomenon in our model by including zero-sized packets in the input and output alphabets $\mathcal{A}$ and $\mathcal{D}$, respectively.

We can augment the use of a buffer with randomized padding to create further randomization between input and output streams. While the padding resource is limited by the average number of padding bits available for random padding, the delay resource is limited by the (average) delay that is introduced into the stream. Therefore, we now have two separate constraints that have to do with average delay and average padding bits:

$$E[Y] \leq Q \tag{5.24}$$

$$E[Z] \leq B \tag{5.25}$$

where Q denotes the largest mean delay that can be tolerated by the stream, and B denotes the bit-padding budget as in Section 5.4.

In the following subsections, we will develop, step-by-step, a framework for studying the obfuscation of streams of variable sized packet as constrained optimization problems, using padding bits and delay as separate resources.

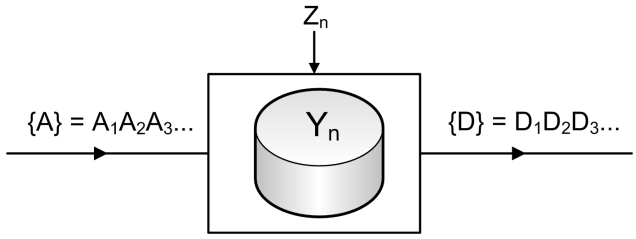


Figure 5.8: The general model for a discrete channel with memory in which packets can be split up and combined. The memory of the channel is manifest in the form of a buffer Y_n that hold bits that have not been sent out yet. In addition, a random amount of bits Z_n can also be used for padding each departing packet.

5.5.1 Obfuscation by Padding only

In this model, packets are altered only by adding padding-bits, as in Section 5.4. Successive packets are allowed to have correlated sizes. Since the output of the channel D_n at time n can, in general be arbitrarily dependent on the set of all past inputs A^n and past outputs D^{n-1}, we may write our objective in its most general form as:

$$\min_{p(D_n|A^n,D^{n-1})} I(\{A\};\{D\})$$

We will refer to the variable $p(D_n|D^{n-1},A^n)$ as the obfuscation channel. How should we design this channel? Notice that in communications engineering we are never faced with the problem of designing a channel with memory to minimize mutual information. Indeed, this is just the opposite of the usual goal of designing a coding scheme for a *given* channel with memory (e.g. a fading channel). As a first step we will consider what the output alphabet $\mathcal{D}$ should be, when padding a stream of packets with possibly correlated sizes.

Theorem 3: The output alphabet $\mathcal{D}$ in the obfuscation of a stream of packet sizes by padding only can be the same as the input alphabet of packet sizes $\mathcal{A}$.

Proof: The proof is similar to the proof of Theorem 2 and therefore we will only provide a sketch here. The proof is trivially true for an *iid* stream $\{A\}$ because such a stream can be treated as independent repeated occurrences of

a single packet, and we have already proved the result for a single packet. Let us consider the case of a correlated stream $\{A\}$. Instead of considering a single letter characterization of the obfuscation channel, $p(D|A)$ as depicted in Figure 5.2 and described in Section 5.4, one may treat the obfuscation channel as if it were operating on a sequence of packets A^n at once, and one may therefore characterize the channel as $p(D^n|A^n)$, with the understanding that there is no delay introduced and $p(D^n|A^n) = 0$ if $D_i < A_i$ for any $i = 1, \ldots, n$. With this characterization, we can now treat the obfuscation channel as a DMC with vector inputs and outputs. We wish to prove that the output of this DMC need only consist of vectors whose elements belong to $\mathcal{A}$.

Consider an output of this DMC $D^N_{(k)}$, where the superscript N denotes the length of the vector and the subscript (k) simply denotes the index of this vector in a lexicographic ordering of the output vectors. Let $D^N_{(k)}$ contain elements that do not belong to $\mathcal{A}$. Let $D^N_{(i)}$ denote the output vector in which each element belongs to $\mathcal{A}$ and is such that each element of $D^N_{(k)}$ that is not in $\mathcal{A}$ is replaced by the largest element in $\mathcal{A}$ smaller than it. With respect to the proof of theorem 2, $D^N_{(k)}$ plays the role of D_k and $D^N_{(i)}$ plays the role of D_i. Similarly, $p(D^N = D^N_{(k)}), p(D^N = D^N_{(i)})$ and $p'(D^N = D^N_{(i)}) = p(D^N = D^N_{(k)}) + p(D^N = D^N_{(i)})$ play roles analogous to $p(D = D_k), p(D = D_i)$ and $p'(D = D_i) = p(D = D_i) + p(D = D_k)$ respectively. With this equivalence, it can be shown, using the set of arguments identical to the ones in the proof of theorem 2, that

$$\begin{aligned} H'\left[A^N|D^N_{(i)}\right] \cdot p'\left[D^N = D^N_{(i)}\right] \quad &\geq \\ H\left[A^N|D^N_{(i)}\right] \cdot p\left[D^N = D^N_{(i)}\right] \quad &+ \\ H\left[A^N|D^N_{(k)}\right] \cdot p\left[D^N = D^N_{(k)}\right] & \end{aligned} \tag{5.26}$$

which in turn, using an argument based on the concavity of the the entropy function, identical to the one used in the proof of Theorem 2 proves that the output need only consists of elements from the input alphabet $\mathcal{A}$. ■

Can the construction of the padding-only obfuscator be simplified from the cumbersome form $p(D_n|A^n, D^{n-1})$? Yes, it can be reduced to the form $p(D_n|A^n)$; that is, the output D_n at time n need not depend upon the previous outputs given the previous inputs. This can further be reduced to the form $p(D_n|A_{n-k}^n)$ if the input stream $\{A\}$ is known to have a memory of k or less packets. We will now show how this simplification in the form of the padding-only obfuscator is possible.

Lemma 1: The following is a Markov Chain:

$$A_n \leftrightarrow A^{n-1} \leftrightarrow D^{n-1}$$

Proof: This can be proved by a simple constructive proof that utilizes the vector-DMC characterization above. Let us assume that A^{n-1} is the input to a vector-DMC and D^{n-1} is the corresponding output. The DMC is therefore characterized by the conditional distribution $p(D^{n-1}|A^{n-1})$. The key observation is that even though in reality our obfuscation channel does not observe the entire vector A^{n-1} before producing D^{n-1}, we can still construct the conditional distribution $p(D^{n-1}|A^{n-1})$ because each D_i only depends upon past inputs A^i and outputs D^{i-i}. Similarly, consider another DMC which takes A^{n-1} as input and produces A^n as output. We now have two DMCs, both taking A^{n-1} as input and producing D^{n-1} and A^n as their respective outputs. From this construction, it follows that $A_n \leftrightarrow A^{n-1} \leftrightarrow D^{n-1}$ is a Markov Chain. ■

Let us now focus on the objective function:

$$\begin{aligned}
& \min_{p(D_n|A^n,D^{n-1})} I(\{A\};\{D\}) \\
= & \min_{p(D_n|A^n,D^{n-1})} H(\{A\}) - H(\{A\}|\{D\}) \\
= & H(\{A\}) - \max_{p(D_n|A^n,D^{n-1})} H(\{A\}|\{D\}) \\
= & H(\{A\}) - \max_{p(D_n|A^n,D^{n-1})} \lim_{n\to\infty} H(A_n|A^{n-1},D^n) \\
= & H(\{A\}) - \max_{p(D_n|A^n,D^{n-1})} \lim_{n\to\infty} H(A_n|A^{n-1},D_n) && (5.27) \\
= & \min_{p(D_n|A^n,D^{n-1})} \lim_{n\to\infty} I(A_n;D_n|A^{n-1}) && (5.28) \\
= & \min_{p(D_n|A^n,D^{n-1})} \lim_{n\to\infty} H(D_n|A^{n-1}) - H(D_n|A^n) && (5.29) \\
= & \min_{p(D_n|A^n)} \lim_{n\to\infty} H(D_n|A^{n-1}) - H(D_n|A^n) && (5.30)
\end{aligned}$$

where (5.27) follows from Lemma 1 and (5.30) follows from (5.29) because the argument of (5.29) does not require D_n to depend upon D^{n-1} but only upon A^n. We have therefore shown that the general obfuscator $p(D_n|A^n, D^{n-1})$ can be simplified to $p(D_n|A^n)$. The obfuscator can be further simplified if it is known that the input stream $\{A\}$ has a known finite memory or is Markovian.

Corollary 2: If the input stream is Markovian such that $p(A_n|A^{n-1}) = p(A_n|A_{n-k}^{n-1})$ then the obfuscator can be simplified from $p(D_n|A^n)$ to $p(D_n|A_{n-k}^n)$.

Proof: If the input stream is k^{th} order Markovian, then $A_n \leftrightarrow A_{n-k}^{n-1} \leftarrow A_1^{n-k-1}$ is a Markov chain. Therefore, we have

$$\begin{aligned}
& \min_{p(D_n|A^n)} \lim_{n\to\infty} H(A_n|A^{n-1}) - H(A_n|A^{n-1},D^n) \\
= & \min_{p(D_n|A^n)} \lim_{n\to\infty} H(A_n|A_{n-k}^{n-1}) - H(A_n|A_{n-k}^{n-1},D_{n-k}^n) \\
= & \min_{p(D_n|A^n)} \lim_{n\to\infty} H(A_n|A_{n-k}^{n-1}) - H(A_n|A_{n-k}^{n-1},D_n) \\
= & \min_{p(D_n|A^n)} \lim_{n\to\infty} I(A_n;D_n|A_{n-k}^{n-1}) \\
= & \min_{p(D_n|A^n)} \lim_{n\to\infty} H(D_n|A_{n-k}^{n-1}) - H(D_n|A_{n-k}^n) \\
= & \min_{p(D_n|A_{n-k}^n)} \lim_{n\to\infty} H(D_n|A_{n-k}^{n-1}) - H(D_n|A_{n-k}^n) && (5.31)
\end{aligned}$$

where (5.31) follows because the argument to be maximized in the last equation does not require D_n to depend upon inputs prior to A_{n-k}; therefore the obfuscator can be simplified to $p(D_n|A_{n-k}^n)$. ■

Note that the simplified objective function

$$\max_{p(D_n|A^n)} H(A_n|A^{n-1}, D^n)$$

is convex in the variable of optimization $p(D_n|A^n)$. This can be seen as follows. $H(A_n|A^{n-1}, D_n)$ is concave in the distribution $p(A^{n-1}, D_n|A_n)$, which in turn can be expressed as $p(D_n|A^n) \times p(A^{n-1}|A_n)$, implying that $p(A^{n-1}, D_n|A_n)$ is affine in $p(D_n|A^n)$. Therefore, $H(A_n|A^{n-1}, D_n)$ must be concave in $p(D_n|A^n)$. Further, if we use a memoryless channel, i.e. we use the simplified variable $p(D_n|A_n)$ instead of the more cumbersome $p(D_n|A^n)$, the objective function continues to be a convex function of $p(D_n|A_n)$, although the optimal value may be smaller. This can be seen as follows. $H(A_n|A^{n-1}, D_n)$ is concave in the distribution $p(A^{n-1}, D_n|A_n)$. Now, if we use a memoryless channel $p(D_n|A_n)$, then D_n is independent of A^{n-1} given A_n and therefore, the following is a Markov Chain: $D_n \leftrightarrow A_n \leftrightarrow A^{n-1}$. Using this fact, $p(A^{n-1}, D_n|A_n)$ simplifies to the product form $p(A^{n-1}|A_n) \times p(D_n|A_n)$. Since the first term in this product is a constant, we find that $p(A^{n-1}, D_n|A_n)$ is affine in $p(D_n|A_n)$, and therefore, $H(A_n|A^{n-1}, D_n)$ must be concave in $p(D_n|A_n)$.

Finally, it must be noted that the rate at which information leaks out through the obfuscation channel - in bits per packet - is lower for a correlated stream $\{A\}$ than for an *iid* stream, as one would intuitively expect. This is true even if one picks the possibly suboptimal padding rule $p(D_n|A_n)$ – that is, a rule where the output is conditionally independent of previous packet sizes, given the present packet size. This can be shown as follows, starting with the objective function in

(5.28):

$$\begin{aligned} & \min_{p(D_n|A^n)} I(A_n; D_n|A^{n-1}) \\ \leq & \inf_{p(D_n|A_n)} I(A_n; D_n|A^{n-1}) && (5.32) \\ = & \min_{p(D_n|A_n)} I(A_n; D_n|A^{n-1}) && (5.33) \\ \leq & \min_{p(D_n|A_n)} I(A_n; D_n) && (5.34) \end{aligned}$$

where (5.32) follows from the fact that using an obfuscator $p(D_n|A_n)$ where the output of the obfuscator is conditionally independent of past inputs given the most recent input D_n can only worsen the optimal mutual information achieved by the more general obfuscator $p(D_n|A^n)$. The equality in (5.33) follows from the fact that inf can be replaced by min because replacing $p(D_n|A^n)$ by $p(D_n|A_n)$ as the variables of optimization preserves the convexity of the variable set and hence $I(\{A\};\{D\})$ remains convex in the new variable. The inequality in (5.34) follows from the fact that when the obfuscator is $p(D_n|A_n)$ rather than $p(D_n|A^n)$, then

$$D_n \leftrightarrow A_n \leftrightarrow A^{n-1}$$

is a Markov Chain and this implies that

$$I(A_n; D_n) \geq I(A_n; D_n|A^{n-1})$$

(see for e.g. [3], Section 2.8).

5.5.2 Obfuscation by buffering only

In this subsection, we will develop a mechanism to obfuscate a stream of packets using buffering only. We will find that this problem translates to the analysis of a queue as an information-theoretic channel, which proves to be a hard problem. However, the insights we develop in this section serve us greatly in understanding how an obfuscation system that combines buffering and padding will work.

When using buffering alone, the obfuscator is a queue of bits. In the i^{th} slot, a packet of size A_i arrives at the queue and a packet of size D_i departs from the head of the queue. There are no padding bits added to the stream as a whole. After the $(n-1)^{th}$ departure D_{n-1}, the buffer contains the set of bits $Y_n = \sum_{j=1}^{n-1}(A_j - D_j)$ that have arrived at the queue but have not departed yet. We use $X_n = Y_n + A_n$ to denote the buffer contents just after the n^{th} arrival A_n (see Figure 5.7). Let us begin by stating the objective in its most general form:

$$\min_{p(D_n|A^n,D^{n-1})} I(\{A\};\{D\})$$

Without any bound on the delay, the above optimization problem might cause an infinite delay (for eg. corresponding to buffering all packets and then releasing them in arbitrary sizes). In practice we cannot tolerate large delays and therefore need to model this as a constraint on some measure of delay. Some possible measures of delay we might consider are:

- Average queue length
- Maximum delay per bit
- Buffer overflow probability

The intuition is that the larger the queuing delay that we are willing to tolerate (say, average delay $\leq Q$), the more unrelated the output of the obfuscation channel can be to the sequence of arrivals at the queue. If the queue is long enough, the output of the channel D_n at time n can be chosen to be completely independent of the sequence of arrivals A^n. However, given, say a bounded mean queue length, the queue would reduce in size sometimes and may even be empty at times, invariable forcing D_n to have some dependence on the size of the queue and hence on the arrival process. In such a situation it is not possible to choose the output of the obfuscation channel independent of the queue size. We are

interested in studying the trade-off between the mutual information between $\{A\}$ and $\{D\}$ and an appropriate measure of delay.

It is difficult to attack the problem in its complete generality, for all possible types of channels with memory and all types of correlation in $\{A\}$. Therefore, using the intuition given above, we will restrict our attention to the class of queues for which $p(D_n|D^{n-1}, A^n) = p(D_n|X_n)$. That is, the class in which the departure D_n in any slot depends only on the size of the queue $X_n = Y_n + A_n$ at that instant, i.e. after the arrival A_n. This will allow the construction and analysis of simple queue control strategies which can make decisions based only on the present state of the buffer. The great advantage of a simplification to $p(D_n|X_n)$ is the vastly reduced search space for the control variables $p(D_n|D^{n-1}, A^n)$. A number of interesting models can be formed using this class of queues, comprising both stochastic and deterministic control policies:

1. $D_i = \min(X_i, S_i)$ where S is a random variable, independent of $\{A\}$ with a fixed distribution $p(S)$, chosen by us.

2. $D_i = \min(X_i, c)$ for some constant c..

3. $D_i = c$ if $X_i \geq C$ and 0 otherwise, for a constant c.

4. $D_i = c$ if $X_i \geq c$ and X_i otherwise, for a constant c.

Model 1 above is a random queue control policy, while the other are deterministic queue controllers (i.e. D is deterministic given X). We will use model 1 above in the sequel. Model 2 is a specific case of Model 1, when the distribution $p(S)$ converges to a delta function. Further, we will stick to considering average delay[2]) as a measure of the queuing delay. Throughout, we will consider a discrete random variable S, with pmf $p(S = i)$, $i \in \mathcal{S}$ to be the variables of our problem, assuming

[2]By an application of Little's law, mean queue length (in bits) is proportional to the mean delay in the discrete-time queue we are using

that a support $\mathcal{S}$ for S has been fixed. Finally, we will assume that the input process $\{A\}$ has a first order Markov correlation, i.e $p(A_n|A^{n-1}) = p(A_n|A_{n-1})$. The justification for this comes from a result in [84] Chapter 6, which explains that a source with arbitrary correlation structure can be approximated by an M^{th}-order Markov chain with arbitrary precision. For analytical ease, we have chosen $M = 1$ - however, it is possible to model higher order Markov chains as a first order Markov chain, by a simple re-definition of the state to include multiple outputs.

The average queue length (or average delay) is convex with respect to the distribution $p(S)$. To establish this, we will make use of a result from [85]. First, observe that $\mu = E[S]$ is linear in $p(S)$; therefore, if the average queue length is convex in μ, then it must be convex in $p(S)$. [85] studies the trade-off between average queue length and the minimum required $\mu = E[S]$ as a rate distortion problem using average queue length as a measure of distortion. It has been shown that the minimum required μ needed to keep the mean queue length under a fixed constant Q, is decreasing and convex in Q. We use this result by observing that it implies that given a μ, the average queue length must be decreasing and convex in μ. Now, $E[S] = \sum_{i \in \mathcal{S}} i \cdot p(S = i)$ is a linear function of the distribution $p(S)$, and by a convexity preserving argument [86], if a function $f(\cdot) : \mathbb{R} \to \mathbb{R}$ is convex and non-increasing and another function $g(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is concave, then the composite function $f(g(x))$ is convex in $x \in \mathbb{R}^n$. Here, the distribution $p(S)$ plays the role of $x \in \mathbb{R}^n$.

Conjecture 1: The mutual information rate $I(\{A\}; \{D\})$ is convex in the distribution $p(s)$.

While we have been unable to conclusively prove this conjecture analytically, our conjecture is based on the observation that mutual information $I(A^N; D^N)$ is convex in the conditional distribution $p(D^N|A^N)$, which can be expressed in terms of $p(S)$ via $p(D|X)$. Therefore, controlling $p(S)$ provides a way to simply

control $p(D^N|A^N)$, and thus it is intuitive that mutual information $I(A^N; D^N)$ would be a convex function of $p(S)$. We believe the hardness of proving this result conclusively arises from the infinite memory in the queue that forms the obfuscation channel. In the following subsection, we will find that the intractability of analyzing the use of buffering as a means for obfuscation can be alleviated if small amounts of padding are also allowed.

In order to fully formulate (and solve, using a computer program) a convex optimization problem for the minimization of mutual information with the pmf $p(S)$ as the set of variables, we need to specify a way of computing the value of the objective function and constraints, for every feasible value of the variables. The usual method of specifying this relationship is via a closed form expression. However, closed form expression for the mutual information and average queue length are very hard to find. Indeed, the analysis of queues with correlated arrivals is cumbersome and usually involves finding the zeros of polynomials in a transform domain [87]. However, as we show below, it is possible to compute both these quantities in terms of known quantities and relations using a 2-dimensional imbedded Markov chain for the queue. In the rest of this subsection, we will derive computable expressions for mutual information and mean queue length, with the 2-D Markov chain as the link between the two.

2-D Markov chain

If the input process $\{A\}$ were iid, then it easy to see that the queue length in successive slots, $\{X\}$ forms an imbedded Markov chain - given the buffer contents at time n, its distribution at time $(n+1)$ is fully determined because we know $p(D|X)$ and $p(A)$. However, when $\{A\}$ is correlated and is itself Markovian, then $\{X\}$ is no longer a Markov chain as can be seen from the relation:

$$X_{k+1} = (X_k - S_k)^+ + A_k \tag{5.35}$$

Both X_k and A_k depend upon A_{k-1}. We get around this problem by the observation that the combination (X_k, A_k), of the queue length just after the $(k)^{th}$ arrival, and size of the $(k)^th$ arrival has the Markov property, and therefore form a 2-dimensional imbedded Markov chain. Indeed, the knowledge of A_k is sufficient to characterize the distribution of A_{k+1}, while A_{k+1} and X_k together determine the probability distribution of X_{k+1}.

Buffer Analysis

We now derive the steady state distribution of the 2-dimensional Markov chain formed by (X_k, A_k). We will assume that we have a finite buffer of size L bytes. If a packet arrives when the buffer is almost full and there is no room for the packet, the entire packet is dropped by the queue. This obviously incurs some loss in the data stream and we must model this loss somehow. We will avoid modeling this loss by choosing the buffer size L to be a very large number compared to the average queue size that we wish to operate under ($L >> Q$). This will make the loss of packets an event rare enough to be neglected. The advantage analyzing the buffer using a finite size buffer is that it allows us to create a finite-sized transition probability matrix for the 2-dimensional Markov chain, in turn allowing us to compute the steady state distribution of this Markov chain. Let $q[n, l|i, j] = Prob[X_{k+1} = n, A_{k+1} = l|X_k = i, A_k = j]$ denote the one-step transition probabilities for the 2-D Markov chain formed by (X_k, A_k). Note that $n \geq l, i \geq j$ because the queue X_k cannot contain less bits than the most recent arrival A_k. If we can specify each of the probabilities in the chain, then the first left eigenvector of this matrix can be computed to yield the steady state distribution. We will now show how the transition probabilities can be computed for the general class of queue control policies of the type $D = min(X, S)$ where

S is a random number with a chosen support $\mathcal{S}$ and distribution $p(S)$. We have:

$$q[n,l|i,j] = \begin{pmatrix} P[A_{k+1}=l|A_k=j]\times \\ P[X_{k+1}=n|X_k=i,A_k=j,A_{k+1}=l] \end{pmatrix}$$

We only need to compute the second term in the product as the source transition probabilities $P[A_{k+1}=l|A_k=j]$ are known. However, since we have a random-sized departure, we now have $X_{k+1} =$

$$\begin{cases} (X_k-S_k)^+ + A_{k+1}, & \text{for } (X_k-S_k)^+ + A_{k+1} \le L \\ (X_k-S_k)^+, & \text{for } (X_k-S_k)^+ + A_{k+1} > L \end{cases}$$

and therefore

$$\begin{aligned} & P(X_{k+1}=n|X_k=i,A_k=j,A_{k+1}=l) \\ = & \begin{cases} P(n=i-D+l|i,l)=p_1, & \text{if } i-D+l \le L \\ P(n=i-D|i,l)=p_2, & \text{if } i-D+l > L \end{cases} \end{aligned}$$

Let us further denote $p_3 = P(i-D+l \le L|i,l)$. Then the required probability $P(X_{k+1}=n|X_k=i,A_k=j,A_{k+1}=l)$ can be written as $p_1p_3 + p_2(1-p_3)$. The probabilities p_1, p_2 and p_3 can be written as $p_1 = P(D=i+l-n|i,l), p_2 = p_2 = P(D=i-n|i,l)$ and $p_3 = P(D \ge i+l-L|i,l)$ and can all be computed by substituting the appropriate values in the conditional distribution:

$$P(D=\beta|X=\gamma) = \begin{cases} \sum_{t=\gamma}^{\infty} P(S=t), & \text{for } \beta=\gamma \\ P(S=\beta), & \text{for } \beta<\gamma \end{cases}$$

Therefore, we have all the information necessary to specify the transition probability matrix P_{2D} for the 2-D Markov chain formed by (X_k, A_k), from which the steady state distribution $\underline{x}$ of the chain's states can be found by solving a set of linear equations of the form $\underline{x} \cdot P_{2D} = \underline{x}$ or by finding the left eigenvector of P_{2D} for the eigenvalue 1. Finally, the steady state queue length can be computed from the steady state distribution of (Y_k, A_{k-1}) by summing over A_{k-1}, allowing us to compute the mean queue length.

Computing Mutual Information

We will show that the mutual information rate between the streams $\{A\}$ and $\{D\}$ can be computed for any obfuscation model, provided we have the following information:

- The transition probabilities for the source
- The obfuscation function (e.g. $D_i = min(X_i, S_i)$ with iid S_i and given $p(S)$)
- The steady state distribution of the two-dimensional Markov chain (Y_{k+1}, A_k).

Note that the steady state probabilities of (Y_{k+1}, A_k) can be easily found from those of the (X_k, A_k) chain, by noting that $Y_{k+1} = X_k - D_k$. Therefore, $p(Y_{k+1} = i, A_k = j) =$

$$\begin{aligned} &= \sum_{n,l} \begin{pmatrix} p(Y_{k+1} = i, A_k = j | X_k = n, A_k = l) \\ \times p(X_k = n, A_k = l) \end{pmatrix} \\ &= \sum_{n} p(D_k = X_k - i | X_k = n) \cdot p(X_k = n, A_k = j) \end{aligned}$$

where the quantity $p(D_k = X_k - i | X_k = n)$ can be computed from the known conditional distribution $p(D|X)$ which is specified by the obfuscation model chosen. We write

$$\begin{aligned} I(\{A\}; \{D\}) &= H(\{A\}) - H(\{A\}|\{D\}) \\ &= H(A_n|A_{n-1}) - \lim_{n\to\infty} H(A_n|A^{n-1}, D^n) \end{aligned}$$

The entropy $H(A_n|A_{n-1})$ can be easily computed using the known transition probabilities of the Markov process $\{A\}$. We will focus on computing the second term $\lim_{n\to\infty} H(A_n|A^{n-1}, D^n)$. Assuming that steady state has been reached, the limit can be dropped. Using the fact that $\{A\}$ is first-order Markov and that $p(D_j|D^{j-i}, A^j) = p(D_j|X_j)$ for any j, we have the Markov chain

$$D^n, A^{n-1} \leftrightarrow A_{n-1}, Y_n, D_n \leftrightarrow A_n$$

where $Y_j = \sum_{i=1}^{j-1}(A_i - D_i)$. This simplifies $H(A_n|A^{n-1}, D^n)$ to $H(A_n|A_{n-1}, Y_n, D_n)$. An important point to note here is that the queue length Y_k is not Markovian, as it would have been, had the input stream been iid. We can now expand $H(A_n|Y_n, A_{n-1}, D_n)$ as:

$$\sum_{i,j,k} \begin{pmatrix} H(A_n|Y_n = i, A_{n-1} = j, D_n = k) \\ \times p(Y_n = i, A_{n-1} = j, D_n = k) \end{pmatrix}$$

where $p(Y_n = i, A_{n-1} = j, D_n = k)$ can be evaluated as follows

$$\begin{aligned} & p(Y_n = i, A_{n-1} = j, D_n = k) \\ = & \sum_l p(Y_n = i, A_{n-1} = j, A_n = l, D_n = k) \end{aligned}$$

and to evaluate $p(Y_n = i, A_{n-1} = j, A_n = l, D_n = k)$, we note that we have the Markov chain

$$Y_n, A_{n-1} \leftrightarrow Y_n, A_n \leftrightarrow D_n,$$

allowing us to expand $p(Y_n = i, A_{n-1} = j, A_n = l, D_n = m)$ as:

$$\begin{pmatrix} p(y_n = i, A_{n-1} = j) \times p(A_n = l|A_{n-1} = j) \\ \times p(D_n = m|A_n = l, Y_n = i) \end{pmatrix} \tag{5.36}$$

which in turn allows us to express $p(Y_n = i, A_{n-1} = j, D_n = m)$ as:

$$\begin{aligned} & p(y_n = i, A_{n-1} = j) \times \\ & \sum_l p(A_n = l|A_{n-1} = j) \cdot p(D_n = m|A_n = l, Y_n = i) \end{aligned} \tag{5.37}$$

which can be computed by a computer for all values of i, j, m if it is provided with the stationary distribution for (Y_n, A_{n-1}), and the queue control strategy (relationship between D and $X = Y + A$). Further, the conditional entropy $H(A_n|Y_n = i, A_{n-1} = j, D_n = k)$ is simply given by $-\sum_q p(r|i, j, k) \log(p(r|i, j, k))$, where $p(r|i, j, k) = p(A_n = r|Y_n = i, A_{n-1} = j, D_n = k)$. These conditional probabilities can be computed as follows:

$$= \frac{p(A_n = r, Y_n = i, A_{n-1} = j, D_n = k)}{p(Y_n = i, A_{n-1} = j, D_n = k)}$$

Both, the numerator and the denominator of the above expression have been computed in (5.36)-(5.37). Therefore, we have shown that given the three quantities listed above, the mutual information between the input and output streams can be easily computed.

5.5.3 Using a combination of buffering & padding

The use of buffering and padding to obfuscate packet streams is complementary and amenable to much simpler analysis than the preceding case of a purely buffering-based obfuscation system.

First, let us focus on the use of buffering and padding to achieve perfect obfuscation, i.e. $I(\{A\};\{D\}) = 0$. In this case, the obfuscated stream $\{D\}$ gives absolutely no information about the true stream $\{A\}$. This is achieved by making the size of each departing packet D_i, completely unrelated to the sequence A^i.

Theorem 4: Given the class of queue control strategies $D_n = min(X_n, S_n)$ the trade-off between delay and padding in the $I(\{A\};\{D\}) = 0$ plane for different choices of $p(S)$ is obtained by using an average padding of $E[S] - E[A] = \mu - \lambda_A$.

Proof: The critical observations are that (i) $\{D\}$ is completely independent of $\{A\}$ if $D_i = S_i$, and (ii) utilizing padding bits when the buffer is not empty is counter-productive. Therefore, padding bits only need to be added when $S_i > X_i$, in the amount of $S_i - X_i$. The departing packet is made up of bits from the buffer, and if needed, some padding bits, but the size of the departing packet is always indepedent of the arrival process. Therefore, we have

$$\begin{aligned} D_i &= \min(X_i, S_i) + Z_i \\ &= \min(X_i, S_i) + (S_i - X_i, 0)^+ \\ &= S_i \end{aligned}$$

In the $I(\{A\};\{D\}) = 0$ plane, the output sequence is $\{D\} = \{S\}$ and therefore $E[D] = \mu = E[S] > \lambda$. Since the queue is stable (i.e. queue length does not blow

up to infinity), by conservation of flow, we must have $E[A] + E[Z] = E[S]$, and therefore, the rate of addition of padding bits to be $E[Z] = E[S] - E[A] = \mu - \lambda_A$.

■

Corollary 3: The trade-off curve between delay and average bit padding in the $I(\{A\}; \{D\}) = 0$ plane is convex.

Proof: The amount of average bit padding used is a linear function of the distribution $p(S)$ because by Theorem 4, it depends only on $\mu = E[S]$. Refer to figure 5.9 which attempts to construct the tradeoff between the amount of padding and the amount of mean delay in the $I = 0$ plane. Now, consider two service distributions $p_1(S)$ and $p_2(S)$. Let $p_\alpha(S) = \alpha p_1(S) + (1-\alpha) p_(S)$ be a linear combination of $p_1(S)$ and $p_2(S)$ for some $\alpha \in (0, 1)$. The linearity of the amount of average bit padding in $p(S)$ implies that $Z_\alpha = \alpha Z_1 + (1 - \alpha) Z_2 \in (Z_1, Z_2)$ where Z_1 and Z_2 correspond to the average bit padding used when $p_1(S)$ and $p_2(S)$ are used, respectively and Z_α is the amount of bit-padding used when $p_\alpha(S)$ is the distribution of S used. Since mean queue length, Q, is a convex function of $p(S)$, therefore we have $Q_\alpha \leq \alpha Q_1 + (1 - \alpha) Q_2$. Combining the facts that $Z_\alpha = \alpha Z_1 + (1 - \alpha) Z_2 \in (Z_1, Z_2)$ and $Q_\alpha \leq \alpha Q_1 + (1 - \alpha) Q_2$, we get the desired result, namely that the delay - padding curve (Q vs. Z curve) is convex. Note that the convexity of the Q vs Z curve in the $I = 0$ plane does not depend upon the convexity of mutual information with respect to $P(S)$. ■

Now, let us analyze the general problem of using padding and buffering to obfuscate the stream $\{A\}$.

Theorem 5: The obfuscation of a stream of packet sizes with constraints on the average bit-padding (5.25) and the average-delay (5.24) is a convex optimization problem.

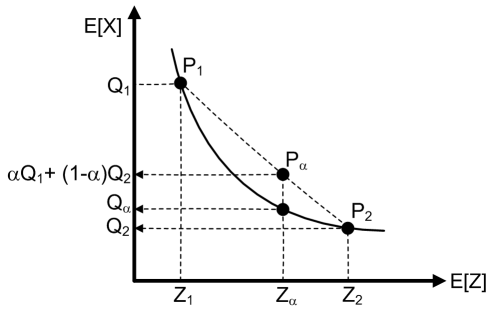


Figure 5.9: The relationship between the amount of average bit padding needed and the amount of average buffering delay is a convex relationship, for any given level of obfuscation (i.e. in the $I(\{A\};\{D\}) = \text{constant}$ plane).

Proof: We wish to prove that the following problem is a convex optimization problem.

$$\min_{\mathbf{P}} I(\{A\};\{D\}) \tag{5.38}$$

such that

$$\sum_j P_{ij} = 1 \ \forall \ i = 1, \ldots, N \tag{5.39}$$

$$E[X] \leq Q \tag{5.40}$$

$$E[Z] \leq B \tag{5.41}$$

$$P_{ij} \in [0,1] \tag{5.42}$$

Let us construct a discrete channel, with input alphabet $\mathcal{A}$ and output alphabet $\mathcal{D} = \mathcal{A}$, such that the channel is defined by the transition probability matrix P, whose $(i,j)^{th}$ element $P_{ij} = pr(D = A_j | A = A_i)$. Unlike the channel defined in Section 5.4 however, here P_{ij} need not be zero for $j < i$. We will allow a packet of size A_i to be transformed to a smaller packet $A_j < A_i$ by keeping the left over bits $A_i - A_j$ stored in the buffer. The number of bits in the buffer is therefore updated as

$$X_{i+1} = X_i + (A_i - A_j)$$

When a packet A_i is transformed to a larger packet $A_j > A_i$, the extra $A_j - A_i$ bits are obtained from the bits held by the buffer at that time, and if the bits in the buffer are not sufficient to make a packet of size A_j, then the required number of dummy bits are padded. Therefore for both the cases $A_j > A_i$ and $A_j < A_i$, we may write

$$Z_i = ((A_j - A_i) - X_i)^+$$

and

$$X_{i+1} = (X_i + (A_i - A_j))^+$$

The proof is based on the fact that an obfuscator that uses a combination of padding and buffering delay can in fact be modeled as a discrete memoryless channel with a queue inside it (see Figure 5.10), rather than a pure queue that we have dealt with in the previous subsection, which is a channel with infinite memory. In order to show that the problem is a convex optimization problem, we need to show that the quantities $E[Z], E[X]$ and $I(\{A\};\{D\})$ are convex in the variable of optimization, namely P. From the flow conservation argument used in the proof of Theorem 4, it can be seen that we have the relationship

$$\begin{aligned} E[Z] &= E[D] - E[A] \\ &= \sum_i \sum_j p_i P_{ij} (A_i - A_j), \end{aligned}$$

which makes $E[Z]$ a linear function of P. Further, the fact that $E[X]$ is a convex function of $p(S)$ in the previous model can be used to show that it is also a convex function of P. This can be shown as follows. In the previous model, the random variable S represented the amount of bits requested from the buffer in order to form a departing packet in each time slot. In the present model, the number of bits requested from the buffer in each time slot is given by the output (with alphabet $\mathcal{A}$) of the discrete channel we have constructed. The distribution of this quantity is given by $p(D = A_j) = \sum_i p_i P_{ij}$, or in vector form,

$$p_D = p^T P$$

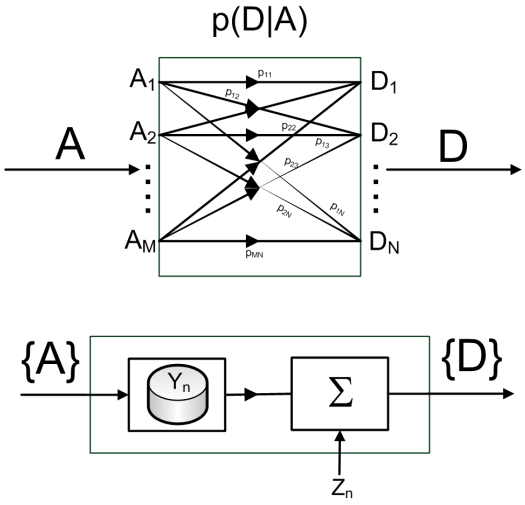


Figure 5.10: The joint use of padding and delay for obfuscation can be modeled as a discrete memoryless channel, even though it has a queue inside it, because the output of the channel need not depend upon the infinite memory in the queue.

Therefore, the distribution of the quantity requested from the buffer is linear in the variable P. Since $E[X]$ is convex in this quantity, $E[X]$ must therefore also be convex in P. Finally, $I(\{A\};\{D\})$ is also convex in P. Mutual information rate may be expanded as

$$I(\{A\};\{D\}) \quad = \quad \lim_{n\to\infty} H(A_n|A^{n-1}) - H(A_n|A^{n-1}, D^n)$$

and therefore we simply need to show that $H(A_n|A^{n-1}, D^n)$ is concave in P. But this has already been shown in Section 5.4, both for a channel with memory, when P is $p(D_n|A^n)$ as well as for a DMC, when P is $p(D_n|A_n)$.

■

If we plot every single combination of padding-budget B, mean buffering delay budget Q and the mutual information corresponding to this pair, we would get a surface such as the one in Figure 5.11. Various points on this surface correspond to different choices of P.

Corollary 4: The surface formed by mutual-information rate corresponding to various combinations of padding and buffering delay (Figure 5.11) is a convex

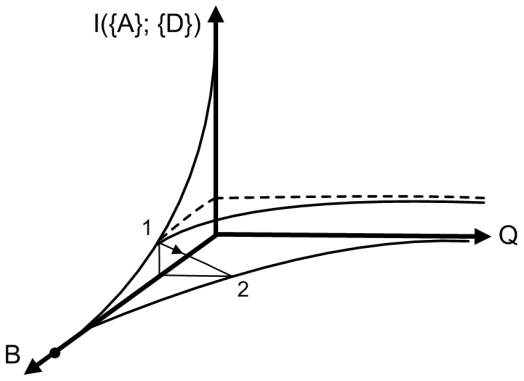


Figure 5.11: The surface formed by the mutual information rate achieved for any combination of average bit-padding and average buffering delay is a convex surface. Therefore, adding a small amount of delay to a bit-padding budget, allows for much better obfuscation (point 1 → point 2 in the figure). All points under this surface are not achievable.

surface. Consequently, the mutual-information rate is a convex function of the amount of padding when delay is fixed, and it is a convex function of the mean delay when the amount of padding is fixed.

Proof: This is simply a consequence of the fact that mutual information rate $I(\{A\};\{D\})$ is convex in P. Consider two different matrices P_1 and P_2 and let these correspond to the points (B_1, Q_1, I_1) and (B_2, Q_2, I_1) in Figure 5.11. From the convexity of $I(\{A\};\{D\})$ with respect to P, we know that the mutual information rate I_α corresponding to $P_\alpha = \alpha P + (1-\alpha)P$ is such that $I_\alpha \leq \alpha I_1 + (1-\alpha)I_2$. This proves that the three-dimensional surface in Figure 5.11 is convex. Points in the region under the surface cannot be achieved, while points on the surface and above it, can. The achievable region is therefore an epigraph of the convex function.

Further since $I(B,Q) : \mathbb{R}^2 \mapsto \mathbb{R}$ is a convex of (B,Q), it must be a convex function in each dimension B and Q separately. This follows from the fact that a convex function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is convex if and only if it is convex along any line padding through $\mathbb{R}^n$ [86]. ■

An important interpretation of the convexity of the surface is the observation that obfuscating using one of the resources (delay or padding) becomes much more effective when some amount of the other resource is thrown in. This is depicted in Figure 5.11.

5.6 Conclusions

Although streams of packets may be encrypted, such mechanisms are not sufficient to ensuring the secrecy of the content contained within the packets. Recent evidence has shown that traffic analysis of encrypted packets can reveal significant information, and consequently in order to achieve heightened levels of confidentiality, it is necessary to employ appropriate countermeasures to prevent traffic analysis. In this chapter, we have examined the problem of padding and delaying packets of a communication flow as a defense against traffic analysis. We started by examining the problem of protecting the size of a single packet transmission, where we sought to minimize mutual information between the original and outgoing packet size. We showed that maximizing the obfuscation level with a bound on the padding budget is a convex optimization problem and further that the output sizes should be chosen from the same set as the original packet sizes. We then examined the more general case involving a stream of packets, where the objective becomes minimizing the mutual information rate. For the packet stream case, we show how it is possible to split and fuse packets by using a buffer that stores the remaining bits of a packet that has been split earlier. We separately analyze the strategies of only padding packets, and only delaying packets before presenting a general traffic-obfuscation queue control strategy that jointly combines delay and padding. As in the single packet case, we show that there is an underlying tradeoff between delay and padding, and ultimately culminate in the observation that delay and padding are synergistic: a strategy involving small

amounts of delay and padding can create far more obfuscation that a strategy involving only delay or only padding.

Appendix A

Key reconciliation using an error correcting code

During reconciliation, Alice and Bob utilize a publicly known (n, k) error correcting linear block code $\mathcal{C}$. Let $\mathsf{Enc}(\cdot) : \{0,1\}^k \mapsto \{0,1\}^n$ be the function that maps a given k-bit string to an n-bit codeword in the code $\mathcal{C}$ and $\mathsf{Dec}(\cdot) : \{0,1\}^n \mapsto \{0,1\}^n$ be the function that maps a given n bit string to the codeword that it is decoded to. Let us assume that the code $\mathcal{C}$ is capable of correcting upto t errors.

Alice first computes the *helper* string $P = w - \mathsf{Dec}(w) \pmod 2$. She then sends P in the clear to Bob. Bob computes $c' = w' - P$ and then computes $\mathsf{Dec}(c')$. If w and w' have a Hamming distance of $d(w, w') < t$, then $\mathsf{Dec}(c') = \mathsf{Dec}(w)$, giving both parties the value $\mathsf{Dec}(w)$ which is a codeword from the (n, k) code. Since there are only 2^k such codewords, the per bit entropy of each codeword is only $k/n < 1$ bit. Therefore, in order to obtain a bit sequence with completely random bits, each user computes $\mathsf{Enc}^{-1}(\mathsf{Dec}(w))$ to arrive at a k-bit key, about which Eve has no information.

It is useful to visualize above set of steps geometrically (see Figure A.1). P can be thought of as a vector pointing from $\mathsf{Dec}(w)$ to w. P leaks partial information about w to Eve because knowledge of P narrows down the possible values of w to all those n-bit strings which are of the form $c + P$, where c is any codeword $\in \mathcal{C}$. Suppose $C = \mathsf{Dec}(w)$ is the codeword corresponding to w. While P doesn't reveal any information to Eve about C, it does helps Bob, who possesses w' to deduce C. Also, it is easy to see using the geometric interpretation, that if $d(w, w') < t$, then $d(w' - P, \mathsf{Dec}(w)) < t$.

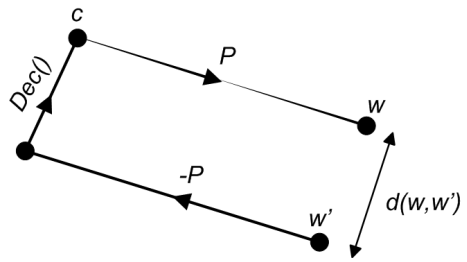


Figure A.1: A geometric visualization of Hamming space when the helper string P is the error between w and the codeword it decodes to $\mathsf{Dec}(w) \triangleq c$. Here, $d(w, w')$ is the Hamming distance between w and w'.

The constructions above assumes that the helper string P is communicated by Alice to Bob without suffering any errors or modifications, which is possible through proper modulation and coding.

Appendix B

List encoding vs. a purely-code based construction

In a purely code-based construction, Alice and Bob obtain bit sequences w and w' respectively, at the rate of 1 bit per coherence time, which on average differ at a fraction ϵ of the bits. This is equivalent to Alice sending her bits to Bob through a Binary Symmetric Channel (BSC1) of capacity $C = 1 - H_b(\epsilon)$. Using list encoding, Alice and Bob obtain $x < 1$ bit per coherence time, and Alice's and Bob's bits differ, on average, at a fraction ϵ_2 of the bit positions. This situation can also be modeled as a BSC (BSC2) with a cross-over probability $\epsilon_2 < \epsilon$.

If it is possible to use a rate x code on BSC1 to lower the error rate from ϵ to a value $\leq \epsilon_2$, (allowing the use of infinitely long blocklengths), then BSC1 can achieve the same combination of rate and error-ratio as BSC2. From Shannon's coding theorem, if $x > C$ then the probability of error cannot be lowered to zero even if the codeword length tends to infinity. Using a well-known result from rate-distortion theory (see for e.g. section 10.4 in [88]), the lowest the probability of error can go (allowing blocklength to approach infinity) when the rate x is larger than the capacity C, is $H_b^{-1}(1 - \frac{C}{x})$. Therefore, if we have

$$x(1 - H_b(\epsilon_2)) > C, \tag{A-1}$$

then BSC1 cannot achieve BSC2's performance of x bits per coherence time at an error rate of ϵ_2, and hence list-encoding is better in this regime. We note that the bound above does not invoke the requirement that the block length must be less than a maximum tolerable N. When this requirement is invoked, then even

if $x < C$, there are a range of values of x and ϵ_2 for which BSC1 cannot achieve the performance of BSC2 for any given finite blocklength $\leq N$. For this range of values also, list encoding is better. Thus the bound expressed above is loose as it allows blocklengths to approach infinity. The exact range of values of x and ϵ_2 that can be achieved by BSC1 for a given maximum block length can be determined using a lower bound on the probability of block decoding error for linear block codes, such as the sphere packing bound. We do not develop this bound further here.

Example. For the purely code-base construction, we observe an error rate of $\epsilon \sim 0.15$ at a distance of $d = 0.1\lambda$ between Alice and Bob. This induces a BSC1 of capacity $C = 1 - H_b(0.15) \sim 0.27$ bits per coherence time. For the same distance between Alice and Bob, list encoding extracts bits at the rate of $x = 0.55$ bits per coherence time with an error rate of $\epsilon_2 = 0.034$. Plugging these numbers into (A-1), we see that list-encoding performs better than the purely code-based method.

References

[1] Robert Gallagher. *Low Density Parity Check Codes*. PhD Dissertation, MIT, 1963.

[2] Q. Wang, S. R. Kulkarni, and S. Verdu. A nearest-neighbor approach to estimating divergence between continuous random vectors. In *Int. Symp. on Inform. Theory*, pages 242–246, 2006.

[3] T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley and Sons, 1991.

[4] Andrea Bittau, Mark Handley, and Joshua Lackey. The final nail in wep's coffin. In *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 386–400, Washington, DC, USA, 2006. IEEE Computer Society.

[5] Erik Tews and Martin Beck. Practical attacks against wep and wpa. In *WiSec '09: Proceedings of the second ACM conference on Wireless network security*, pages 79–86, New York, NY, USA, 2009. ACM.

[6] Vladimir Brik, Suman Banerjee, Marco Gruteser, and Sangho Oh. Wireless device identification with radiometric signatures. In *MobiCom '08: Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 116–127, New York, NY, USA, 2008. ACM.

[7] Loh Chin Choong Desmond, Cho Chia Yuan, Tan Chung Pheng, and Ri Seng Lee. Identifying unique devices through wireless fingerprinting. In *WiSec '08: Proceedings of the first ACM conference on Wireless network security*, pages 46–55, New York, NY, USA, 2008. ACM.

[8] Jeffrey Pang, Ben Greenstein, Ramakrishna Gummadi, Srinivasan Seshan, and David Wetherall. 802.11 user fingerprinting. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 99–110, New York, NY, USA, 2007. ACM.

[9] Marco Gruteser and Dirk Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *MobiSys '03: Proceedings of the 1st international conference on Mobile systems, applications and services*, pages 31–42, New York, NY, USA, 2003. ACM.

[10] C. Wright, L. Ballard, F. Monrose, and G. Masson. Language identification of encrypted voip traffic: Alejandra y roberto or alice and bob? In *Proceedings of the 16th USENIX Security Symposium*, 2007.

[11] C. Wright, L. Ballard, S. Coulls, F. Monrose, and G. Masson. Spot me if you can: recovering spoken phrases in encrypted voip conversations. In *Proceedings of IEEE Symposium on Security and Privacy*, 2008.

[12] Simon Singh. *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*. Anchor, 2000.

[13] Claude Shannon. *Communication theory of secrecy systems*. The Bell System Technical Journal, Vol. 28, No. 4 (October 1949), pp. 656-715, 1949.

[14] Gilbert S. Vernam. Cipher printing telegraph systems for secret wire and radio telegraphic communications. *Journal of the IEEE*, 55, 1926.

[15] J. Wright. *Dispelling Common Bluetooth Misconceptions*. SANS Technology Institute Security Laboratory, 2007.

[16] Andreas Molisch. *Wireless Communications*. Wiley-IEEE Press, 2005.

[17] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR., 2001.

[18] M. Rudolf and R. P. Mukherjee. *Method and system for deriving an excryption key using joint randomness not shared by others*. US Patent Application Publication US2007/0058808A1, 2007.

[19] U.M. Maurer. Secret key agreement by public discussion from common information. *IEEE Transactions on Information Theory*, 39(4):733–742, 1993.

[20] Rudolph Ahlswede and Imre Csiszar. Common randomness in information theory and cryptography – Part I: Secret sharing. *IEEE Transactions on Information Theory*, 39(4):1121–1132, 1993.

[21] J. Cardinal and G. Van Assche. Construction of a shared secret key using continuous variables. *Info. Theory Workshop*, 2003.

[22] G. Brassard and L. Salvail. Secret key reconciliation by public discussion. *Advances in Crytology Proc. - Eurocrypt '93, Lecture Notes in Computer Science*, 765:410–423, 1994.

[23] C. Ye, A. Reznik, and Y. Shah. Extracting secrecy from jointly Gaussian random variables. In *Proceedings of IEEE Int. Symp on Info. Theory*, pages 2593 – 2597, Jul 2006.

[24] C. Cachin and U. M. Maurer. Linking information reconciliation and privacy amplification. *Journal of Cryptology: the journal of the International Association for Cryptologic Research*, 10(2):97–110, Spring 1997.

[25] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley, 1991.

[26] Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM J. Comput.*, 17(2):210–229, 1988.

[27] W. T. Buttler, S. K. Lamoreaux, J. R. Torgerson, G. H. Nickel, C. H. Donahue, and C. G. Peterson. Fast, efficient error reconciliation for quantum cryptography. *Phys. Rev. A*, 67:052303.1–052303.8, 2003.

[28] Gilles Van Assche. *Quantum Cryptography and Secret Key Distillation*. Cambridge University Press, 2006.

[29] Ueli Maurer and Stefan Wolf. Secret key agreement over a non-authenticated channel -Part II: The simulatability condition. *IEEE Transactions on Information Theory*, 49(4):832–838, April 2003.

[30] Zang Li, Wenyuan Xu, Rob Miller, and Wade Trappe. Securing wireless systems via lower layer enforcements. In *WiSe '06: Proceedings of the 5th ACM workshop on Wireless security*, pages 33–42, 2006.

[31] Neal Patwari and Sneha K. Kasera. Robust location distinction using temporal link signatures. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 111–122, 2007.

[32] L. Xiao, L. Greenstein, N.B. Mandayam, and W. Trappe. Fingerprints in the ether: Using the physical layer for wireless authentication. In *Proceedings of the IEEE Int.. Conf. on Comm.*, pages 4646 – 4651, 2007.

[33] Babak Azimi-Sadjadi, Aggelos Kiayias, Alejandra Mercado, and Bulent Yener. Robust key generation from signal envelopes in wireless networks. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 401–410, 2007.

[34] R. Wilson, D. Tse, and R. Scholtz. Channel identification: Secret sharing using reciprocity in UWB channels. *IEEE Transactions on Information Forensics and Security*, 2(3):364–375, 2007.

[35] T. Aono, K. Higuchi, T. Ohira, B. Komiyama, and H. Sasaoka. Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels. *IEEE Transactions on Antennas and Propagation*, 53(11):3776–3784, Nov 2005.

[36] A. Hassan, W. Stark, J. Hershey, and S. Chennakeshu. Cryptographic key agreement for mobile radio. *Digital Signal Processing*, 6:207–212, 1996.

[37] Havish Koorapaty, Amer Hassan, and Sandeep Chennakeshu. Secure information transmission for mobile radio. *IEEE Communication Letters*, 4(2), Feb 2000.

[38] Andrea Goldsmith. *Wireless Communications*. Cambridge University Press, New York, NY, USA, 2005.

[39] J. K. Tugnait, Lang Tong, and Zhi Ding. Single-user channel estimation and equalization. *IEEE Signal Processing Magazine*, 17:16–28, 2000.

[40] W. C. Jakes. *Microwave Mobile Communiations*. Wiley, 1974.

[41] T. Moore. IEEE 802.11-01/610r02: 802.1x and 802.11 key interactions. *Microsoft Research*, 2001.

[42] S.J. Fortune, D. M. Gay, B.W. Kernighan, O. Landron, R. A. Valenzuela, and M.H. Wright. Wise design of indoor wireless systems: practical computation andoptimization. *Computational Science and Engineering, IEEE*, 2(1):58–68, April 1995.

[43] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[44] U. M. Maurer. A universal statistical test for random bit generators. *Journal of Cryptology*, 5:89–105, 1992.

[45] http://csrc.nist.gov/groups/st/toolkit/rng/.

[46] NIST. A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications, 2001.

[47] IEEE standard 802.11a: Part 11 wireless LAN medium access control (MAC) and physical layer (PHY) specifications: High-speed physical layer in the 5 GHz band.

[48] http://www.atheros.com/.

[49] http://www.madwifi.org/.

[50] http://www.tcpdump.org.

[51] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. on Information Theory*, 22(6):644–654, 1976.

[52] B. O'Higgins, W. Diffie, L. Strawczynski, and R. do Hoog. Encryption and ISDN - a natural fit. In *in Proceedings of the International Switching Symposium (ISS87)*, pages 863 – 869, 1987.

[53] C. H. Bennett and G. Brassard. Public-key distribution and coin tossing. In *Proceedings of the IEEE Interantonal Conference on Computers, Systems and Signal Processing*, pages 175–179, 1984.

[54] F. Forsshans, G. Van Assche, J. Wenger, R. Brouri, N.J. Cerf, and P. Grangier. Quantum key distributions using Gaussian modulation coherent states. *Nature*, 421, 2003.

[55] G. Van Assche, J. Cardinal, and N.J. Cerf. Reconciliation of a quantum-distributed gaussian key. *IEEE Trans. on Information Theory*, 50:394–400, 2004.

[56] Amer Hassan, Wayne Stark, John Hershey, and Sandeep Chennakeshu. Cryptographic key agreement for mobile radio. *Digital Signal Processing*, 6:207–212, 1996.

[57] Suhas Mathur, Wade Trappe, Narayan Mandayam, Chunxuan Ye, and Alex Reznik. Radio-telepathy: Extracting a secret key from an unauthenticated wireless channel. In *Proc. of the 14th annual conference on mobile computing and systems (MobiCom 2008)*, San Francisco, CA, Sept 2008.

[58] Chunxuan Ye, Suhas Mathur, Alex Reznik, Yogendra Shah, Wade Trappe, and Narayan Mandayam. Information-theoretically secret key generation for fading wireless channels. *Accepted for publication at the IEEE Trans. on Information Forensics and Security, June 2010, preprint available at http://arxiv.org/abs/0910.5027*, 2010.

[59] Suman Jana, Sriram Nandha Premnath, Mike Clark, Sneha K. Kasera, Neal Patwari, and Srikanth V. Krishnamurthy. On the effectiveness of secret key extraction from wireless signal strength in real environments. In *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking (MobiCom '09)*, pages 321–332, September 2009.

[60] C. Kuo, M. Luk, R. Negi, and Adrian Perrig. Message-in-a-bottle: User-friendly and secure key deployment for sensor nodes. In *In Proceedings of the ACM Conference on Embedded Networked Sensor System (SenSys 2007)*, pages 233 – 246, 2007.

[61] R. Mayrhofer and H. Gellersen. Shake well before use: Intuitive and secure pairing of mobile devices. *IEEE Transactions on Mobile Computing*, 8(6):792–806, 2009.

[62] D. Bichleri, G. Stromberg, and M. Huemer. Key generation based on acceleration data of shaking processes. *Ubiquitous Computing 2007*, pages 304–317, 2007.

[63] A. Scannell, A. Varshavsky, A. LaMarca, and E. Lara. Proximity based authentication of mobile devices. *Int. J. Secur. Netw.*, 4(1/2), 2009.

[64] A. Varshavsky, A. Scannell, A. LaMarca, and E. de Lara. Amigo: Proximity-based authentication of mobile devices. In *In Proceedings of UbiComp 2007: Ubiquitous Computing*, page 253270, 2007.

[65] Near field communications forum, http://www.nfc-forum.org/.

[66] D. Klinc, J. Ha, S.W. McLaughlin, J. Barros, and B.J. Kwak. Ldpc codes for physical layer security. In *to appear in Proceedings of the IEEE Globecom*, 2009.

[67] E. Haselsteiner and K. Breitfuss. Security in near field communication (NFC). *Workshop on RFID Security RFIDSec*.

[68] Andreas Molisch. *Wireless Communications*. Wiley-IEEE Press, 2005.

[69] http://gnuradio.org/trac.

[70] T. Scott Saponas, Jonathan Lester, Carl Hartung, Sameer Agarwal, and Tadayoshi Kohno. Devices that tell on you: privacy trends in consumer ubiquitous computing. In *SS'07: Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, pages 1–16, Berkeley, CA, USA, 2007. USENIX Association.

[71] Y. Sun and K.J.R. Liu. Analysis and protection of dynamic membership information for group key distribution schemes. *IEEE Transactions on Information Forensics and Security*, 2:213–226, 2007.

[72] P. Kamat, W. Xu, W. Trappe, and Yanyong Zhang. Temporal privacy in wireless sensor networks. In *ICDCS '07: Proceedings of the 27th IEEE International Conference on Distributed Computing Systems (ICDCS'07)*, pages 23–31, 2007.

[73] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24:84–88, 1981.

[74] Claudia Diaz and Bart Preneel. Taxonomy of mixes and dummy traffic. In *3rd Working Conference on Privacy and Anonymity in Networked and Distributed Systems*, 2004.

[75] M.Reed, P. Syverson, and D. Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 16:482–494, May 1998.

[76] Dogan Kesdogan, Jan Egner, and Roland Buschkes. Stop-and-go-mixes providing probabilistic anonymity in an open system. In *Proceedings of the Second International Workshop on Information Hiding*, pages 83–98, London, UK, 1998. Springer-Verlag.

[77] G. Danezis. The traffic analysis of continuous-time mixes. In David Martin and Andrei Serjantov, editors, *Privacy Enhancing Technologies (PET 2004)*, May 2004.

[78] Dawn Xiaodong Song, David Wagner, and Xuqing Tian. Timing analysis of keystrokes and timing attacks on ssh. In *SSYM'01: Proceedings of the 10th conference on USENIX Security Symposium*, pages 25–25, Berkeley, CA, USA, 2001. USENIX Association.

[79] Venkat Anantharam and Sergio Verdu. Bits through queues. *IEEE Transactions on Information Theory*, 42, 1996.

[80] Qixiang Sun, Daniel R. Simon, Yi-Min Wang, Wilf Russell, Venkat N. Padmanabhan, and Lili Qiu. Statistical identification of encrypted web browsing traffic. In *IEEE Symposium on Security and Privacy*. Society Press, 2002.

[81] George Dean Bissias, Marc Liberatore, David Jensen, and Brian Neil Levine. Privacy vulnerabilities in encrypted http streams. In *In Proceedings of Privacy Enhancing Technologies Workshop (PET 2005*, pages 1–11, 2005.

[82] Google Doubleclick Ad-Planner. The 1000 most-visited sites on the worldwide web. http://www.google.com/adplanner/static/top1000/.

[83] Rui Menezesb Andreia Dionisioa and Diana A. Mendesb. Mutual information: a measure of dependency for nonlinear time series. *Physica A: Statistical Mechanics and its Applications, Applications of Physics in Financial Analysis 4 (APFA4)*, 344, 2004.

[84] Robert Ash. *Information Theory*. Dover Publications, 1965.

[85] C.S. Chang and Joy Thomas. Rate distortion functions and effective bandwidth of queueing processes. In *Proceedings of the 1995 IEEE International Symposium on Information Theory*, pages 103–103, Piscataway, NJ, 1995. IEEE.

[86] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.

[87] M.L. Chaudhry and John G. C. Templeton. *A First Course in Bulk Queues*. John Wiley, 1983.

[88] David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA, 2003.

Vita

Suhas Mathur

Education

- B.Tech, Electrical Engineering, 07/2004
  Indian Institute of Technology Madras, India
- M.S., Electrical & Computer Engineering, 01/2007
  WINLAB, Rutgers University, NJ, USA
- Ph.D., Electrical & Computer Engineering, 10/2010
  WINLAB, Rutgers University, NJ, USA

Experience

- Teaching Assistant, 09/2004 - 06/2005
  Electrical & Computer Engineering Dept., Rutgers University, NJ, USA
- Systems Engineering Intern, Corporate R&D, 06/2006 - 08/2006
  Qualcomm Inc., San Diego, CA, USA
- Intern, Chief Technology Office, 05/2008 - 08/2008
  InterDigital Inc., King of Prussia, PA, USA
- Graduate Research Assistant, 09/2005 - 08/2010
  WINLAB, Rutgers University, NJ, USA

Publications

- Mobility Emulation Through Spatial Switching on a Wireless Grid (Demo), Kishore Ramachandran, Sanjit Kaul, Suhas Mathur, Marco Gruteser, and Ivan Seskar, International Conference on Mobile Systems, Applications and Services (ACM MobiSys), 06/2005
- Towards Large-Scale Mobile Network Emulation Through Spatial Switching on a Wireless Grid, K. Ramachandran, Sanjit Kaul, Suhas Mathur and Marco Gruteser, Workshop on experimental approaches to wireless network design & Analysis, SIGCOMM, 08/2005.

- Coalitional Games in Receiver Cooperation for Spectrum Sharing, Suhas Mathur, Lalitha Sankar and Narayan Mandayam, Conference on Information Sciences and Systems (CISS), 03/2006
- Coalitional games in Gaussian Interference Channels, Suhas Mathur, Lalitha Sankar and Narayan Mandayam, International Symposium on Information Theory (ISIT) 06/2006
- Coalitional Games in Cooperative Networks, Suhas Mathur, Lalitha Sankar and Narayan Mandayam, Asilomar Conference on Signals, Systems and Computers, 09/2006
- Coalitions in Cooperative Wireless Networks, Suhas Mathur, Lalitha Sankar and Narayan Mandayam, IEEE Journal on Selected Areas in Communications, 09/2008
- Radio-telepathy: Extracting a Cryptographic Key from an Unauthenticated Wireless Channel, Suhas Mathur, W. Trappe, N.B. Mandayam, A. Reznik and C. Ye, The 16^{th} Annual International Conference on Mobile Computing and Networking, (ACM MobiCom), San Francisco, CA, 09/2008
- ParkNet: Harvesting Real-Time Vehicular Parking Information Using a Mobile Sensor Network, Suhas Mathur, Sanjit Kaul, Marco Gruteser and Wade Trappe, The S3 Workshop at The International Symposium on Mobile Ad Hoc Networking and Computing, ACM MobiHoc, 05/2009.
- Secret Key Extraction from Level Crossings over Unauthenticated Wireless Channels, Suhas Mathur, W. Trappe, Narayan Mandayam, Chunxuan Ye, and A. Reznik, Book chapter, Securing Wireless Communications at the Physical Layer, Springer Verlag, 2009
- Information-theoretic Key Generation from Wireless Channels, Chunxuan Ye, Suhas Mathur, A. Reznik, W. Trappe and Narayan Mandayam, IEEE Transactions on Information Forensics and Security, June 2010.
- ParkNet: Drive-by Sensing of Road-side Parking Statistics , Suhas Mathur, Tong Jin, Nikhil Kasturirangan, Janani Chandrasekharan, Wenzhi Xue, Marco Gruteser, Wade Trappe, The 8^{th} International Conference on Mobile Systems, Applications, and Services (ACM Mobisys) [Best Paper Award], 06/2010
- Exploiting the Physical Layer for Enhanced Security, Suhas Mathur, A. Reznik, Rajat Mukherjee, Akbar Rahman, Yogendra Shah, W. Trappe and Narayan Mandayam, IEEE Wireless Communications Magazine, special issue on Security & Privacy, 10/2010