2018

Secret key establishment from common randomness represented as complex correlated random processes: Practical algorithms and theoretical limits

Mohammad R. Khalili Shoja
Iowa State University

Secret key establishment from common randomness represented as complex correlated random processes: Practical algorithms and theoretical limits

by

Mohammad R Khalili Shoja

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Electrical and Computer Engineering (Secure and Reliable Computing)

Program of Study Committee:
George T. Amariucai, Co-major Professor
Zhengdao Wang, Co-major Professor
Yong Guan
Daji Qiao
Doug Jacobson
Vivekananda Roy

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation. The Graduate College will ensure this dissertation is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2018

DEDICATION

I dedicate this thesis to my amazing wife, Zahra for her unconditional love, sacrificial care and support during the years of my Ph.D., and to our loving parents, to whom we owe everything and whose prayers kept me going.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

ACKNOWLEDGEMENTS

This thesis was completed under supervision of my Ph.D. advisors, Professors George Traian Amariucai and Zhengdao Wang. I would like to thank Professor Amariucai for his guidance and support throughout my Ph.D. studies. His insight and advice have always guided me in my personal and professional life. I am also thankful to Professor Zhengdao Wang for his insightful discussions. I would also like to thank my Ph.D. committee members, Professors Doug Jacobson, Daji Qiao, Yong Guan, and Vivekananda Roy for their helpful suggestions.

I would like to thank my friends and labmates at Iowa State University for making my stay in Ames, Iowa memorable.

ABSTRACT

Establishing secret common randomness between two or multiple devices in a network resides at the root of communication security. In its most frequent form of key establishment, the problem is traditionally decomposed into a randomness generation stage (randomness purity is subject to employing often costly true random number generators) and an information-exchange agreement stage, which relies either on public-key infrastructure or on symmetric encryption (key wrapping).

This dissertation has been divided into two main parts. In the first part, an algorithm called KERMAN is proposed to establish secret-common-randomness for ad-hoc networks, which works by harvesting randomness directly from the network routing metadata, thus achieving both pure randomness generation and (implicitly) secret-key agreement. This algorithm relies on the route discovery phase of an ad-hoc network employing the Dynamic Source Routing protocol, is lightweight, and requires relatively little communication overhead. The algorithm is evaluated for various network parameters, and different levels of complexity, in OPNET network simulator. The results show that, in just ten minutes, thousands of secret random bits can be generated network-wide, between different pairs in a network of fifty users.

The proposed algorithm described in this first part of this research study has inspired study of the problem of generating a secret key based on a more practical model to be explored in the second part of this dissertation. Indeed, secret key establishment from common randomness has been traditionally investigated under certain limiting assumptions, of which the most ubiquitous appears to be that the information available to all parties comes in the form of independent and identically distributed (i.i.d.) samples of some correlated random variables. Unfortunately, models employing the i.i.d assumption are often not accurate representations of real scenarios. A more capable model would represent the available information as correlated hidden Markov models (HMMs), based on the same underlying Markov chain. Such a model accurately reflects the scenario where all parties have access to imperfect observations of the same source random process, exhibiting a certain time dependency. In the second part of the dissertation , a computationally-efficient asymptotic bounds for the secret key capacity of the correlated-HMM scenario has been derived. The main obstacle, not only for this model, but also for other non-i.i.d cases, is the computational complexity. This problem has been addressed by converting the initial bound to a product of Markov random matrices, and using recent results regarding its convergence to a Lyapunov exponent.

CHAPTER 1. INTRODUCTION TO THE INFORMATION-THEORETIC CRYPTOGRAPHY

1.1 Introduction

Information-theoretic methods are among the most important approaches in the field of security whose merit is that they require no assumptions about computational capabilities of adversaries. The introduction of these methods goes backs to 1949, when Shannon initially studied the secrecy based on the model in Figure 1.1.

In this model, plain text message (M) is encrypted into ciphertext (C) by applying a shared key (K) at the transmitter, with the decryption process accomplished using the same key at the receiver. The main goal is to transmit M in secrecy. Shannon defined the system to be perfectly secure if knowledge of the plaintext message cannot be achieved just by knowing the ciphertext [1].

$$I(M;C) = 0 \tag{1.1}$$

Moreover, the receiver must have capability for decrypting the plaintext by knowing both the ciphertext and the key.

$$H(M|C,K) = 0 \tag{1.2}$$

By considering such conditions, Shannon [1], using a combinatorial proof, showed that $H(K) \geq H(M)$. In other words, the length of the key should be larger than the length

[1] The preliminaries of information theory have been reviewed in Appendix A

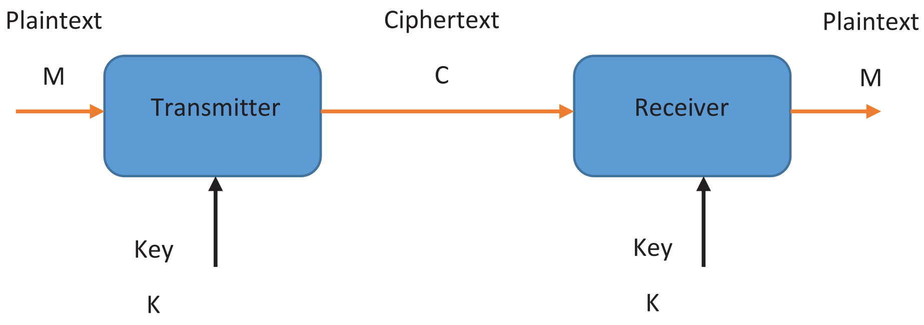

Figure 1.1 Shannon's Model to Study the Secrecy

of the plaintext, a proposition known as "Shannon's pessimistic result". Yeung [2] used a technique based on an information diagram to prove the same result.

Wyner [3] in 1975 introduced a wiretap channel and studied this models secrecy capacity. Wyner's model consists of two channels; the first channel is located between legitimate users (main channel) and second channel is located between a sender and an attacker (attacker's channel). Wyner assumed the main channel to be better than the attacker's channel and also assumed that the attacker is a passive attacker and cannot send any message into the channels. The main goal in the wiretap model is to transmit message M in such a way that the attacker can obtain no information about it. Wyner showed that in the wiretap model there is no need for a pre-existing shared secret key between legitimate sender and receiver. Csiszár and Körner [4] generalized Wyner's model by assuming that the attacker's channel is not inferior to the main channel, and studied the secrecy capacity of the new model, depicted in Figure 1.2.

An important problem in the field of security is generating a secret key, the basic tenet of cryptography and secure communication. Generating a secret key based on information theoretic methods was initially studied by Maurer [5] and Ahlswede and Csiszár [6].

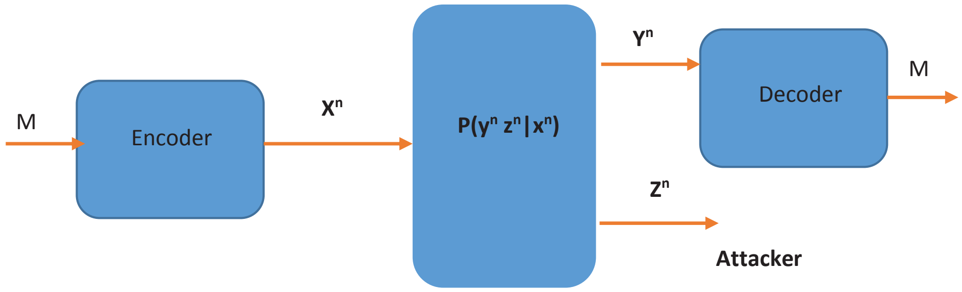

Figure 1.2 Wiretap Model

In this chapter, the problem of generating a secret key between two users for the case of independent and identically distributed (i.i.d) random variables is defined, followed by the exploration of prevalent concepts and tools employed in information-theoretic methods.

1.2 Generating Secret Key in the case of i.i.d Random Variables

Assume i.i.d repetition of (X, Y, Z) with joint probability distribution P_{XYZ}. In the process of generating a secret key, Alice, who can observe X^n, wants to produce secret common randomness with Bob, who has access to Y^n, by exchanging message F over a public channel. This key should be hidden from the perspective of Eve who has side information Z^n and can also overhear the public communication.

The process of generating a secret key is composed of two phases: information reconciliation and privacy amplification. The aim of the former is to generate (with high probability) the same common information between the two parties. Since, in the information reconciliation phase, two parties using public communication reveal some information to a potential eavesdropper, the goal of privacy amplification is to boost the security of the generated key by extracting a (shorter) secret that is uniformly distributed over its space, given the adversary's knowledge. At the end of the protocol, Alice and Bob have random variables K_A

and K_B for the secret key (Figure 1.5). This model for generating secret key is called the source-type model. The two random variables must with high probability be the same and, given Eve's knowledge (F and Z^n), their probability distribution must be uniform.

The techniques employed in i.i.d models are based on important theorems, most related to typical sequences, packing lemma, Slepian-Wolf theorem, etc. (we have provided an introduction to these concepts in Appendix A). In this section we will review methods for generating a secret key in i.i.d models.

For simplicity, assume that, while Eve has no side information (there is no any Z), she can overhear the public communication between Alice and Bob (F). At the end of the process of generating a secret key, K_A and K_B should with high probability be the same and their distribution must be uniform. The key should also be independent of public communication. In some literature, this condition is stated as follows.

$$I(F;K) \leq \epsilon \tag{1.3}$$

We will present an accurate definition of secret key in Chapter 3 of this thesis.

It can be proved that converse bound of the secret key length per observation in the i.i.d framework is $R_K < I(X;Y)$.

The achievability proof for this problem has been traditionally solved using random binning and the Slepian-Wolf theorem. Intuitively, based on Slepian-Wolf theorem, if Alice sends $nH(X|Y)$ bits to Bob, Bob can reconstruct X^n (this phase is called information reconciliation phase) so that both Alice and Bob have $nH(X)$ bits at their disposal while $nH(X|Y)$ of these bits have been released. The secret key length per observation would therefore be

$$\frac{1}{n}(nH(X) - nH(X|Y)) = H(X) - H(X|Y) = I(X;Y) \tag{1.4}$$

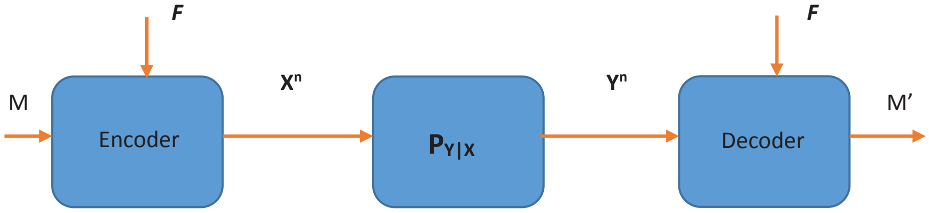

Figure 1.3 Discrete Memoryless Channel Model

As mentioned above, the exact proof of achievability is traditionally based on random binning. Yassaee, et. al, [7] have proposed a new approach for the proof of achievability. This proposed method, called output statistics of random binning (OSRB), can be considered as a general framework for obtaining achievability results in network information theory problems. In other words, it can be employed as an alternative approach instead of using traditional packing and covering lemmas. The main idea in OSRB is exploiting duality between problems. For example, they show that by observing the duality existing between channel coding and secret key generation, the secret-key achievability rate can be derived. Consider a DMC with message M and shared codebook F that can be considered as shared randomness. By considering the model depicted in Figure 1.3 as a Bayesian network, the joint probability distribution of this model can be written as follows.

$$\begin{aligned} P_M^{uni} P_F P_{X^n|MF} P_{Y^n|X^n} P_{\hat{M}|FY^n} &= \\ P_{X^n MF} P_{Y^n|X^n} P_{\hat{M}|FY^n} &= \\ P_{X^n} P_{MF|X^n} P_{Y^n|X^n} P_{\hat{M}|FY^n} &= \\ P_{X^n Y^n} P_{MF|X^n} P_{Y^n|X^n} P_{\hat{M}|FY^n} & \end{aligned} \tag{1.5}$$

The last expression states the joint probability distribution for the generating secret key model illustrated in Figure 1.4. Hence, by choosing $R < I(X;Y)$, the error probability of

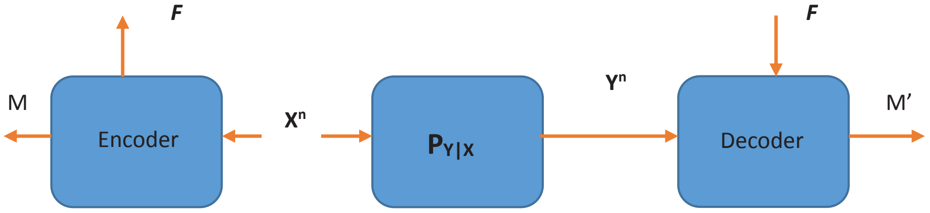

Figure 1.4 Source Secret Key Generation Model

these two models are the same and F and M are independent of one another, meeting all the required conditions for generating a secret key.

1.2.1 OSRB Framework

As discussed earlier, the OSRB framework can be considered as an alternative method for packing and covering lemmas, and in this section, after reviewing a simple form of two important theorems in OSRB, we will employ them for solving the two earlier-mentioned important problems.

Theorem 1. *Consider (X_1^n, X_2^n, Y^n) as an i.i.d repetition of distributed sources with joint probability distribution $P_{X_1^n X_2^n Y^n}$. Then, by assuming random binning ($\mathcal{B}_1$ and $\mathcal{B}_2$) as follows:*

$$\mathcal{B}_i : \mathcal{X}_i^n \rightarrow [1 : 2^{nR_i}], \qquad i = 1, 2 \tag{1.6}$$

if

$$\begin{aligned} R_1 &\geq H(X_1|X_2Y) \\ R_2 &\geq H(X_2|X_1Y) \\ R_1 + R_2 &\geq H(X_1X_2|Y) \end{aligned} \tag{1.7}$$

then

$$E_{\mathcal{B}_1\mathcal{B}_2}(\|\dot{P}(x_1^n, x_2^n, y^n, \hat{x}_1^n, \hat{x}_2^n) - P(x_1^n, x_2^n, y^n)\mathbb{1}[x_1^n = \hat{x}_1^n, x_2^n = \hat{x}_2^n]\|) \to 0 \tag{1.8}$$

as n goes to ∞.

Note that $\dot{P}$ is the random probability distribution induced by two random binnings. Theorem 1 is an equivalent form of the Slepian-Wolf theorem. [7]

Theorem 2. *Consider (X_1^n, X_2^n, X_3^n) as an i.i.d repetition of distributed sources with joint probability distribution $P_{X_1X_2X_3}$. Then, by assuming random binning ($\mathcal{B}_1$ and $\mathcal{B}_2$) as follows:*

$$\mathcal{B}_i : \mathcal{X}_i^n \to [1 : 2^{nR_i}], \qquad i = 1, 2 \tag{1.9}$$

and if

$$\begin{aligned} R_1 &\leq H(X_1|X_3) \\ R_2 &\leq H(X_2|X_3) \\ R_1 + R_2 &\leq H(X_1X_2|X_3) \end{aligned} \tag{1.10}$$

then

$$E_{\mathcal{B}_1\mathcal{B}_2}(\|\dot{P}(b_1(x_1^n), b_2(x_2^n), x_3^n, \hat{x}_1^n, \hat{x}_2^n) - P(x_3^n)P^{uni}(b_1(x_1^n))P^{uni}(b_2(x_2^n))\|) \to 0 \tag{1.11}$$

as n goes to ∞.

In this theorem b_i is the realization of $\mathcal{B}_i$.

To apply the OSRB framework to solving the earlier-mentioned problem of generating a secret key mentioned, let us assume in theorem 1 that if there is no X_2^n and $X_1^n = X^n$, the result is that if $R_F = R \geq H(X|Y)$, then Bob can reconstruct X^n. As discussed earlier, the

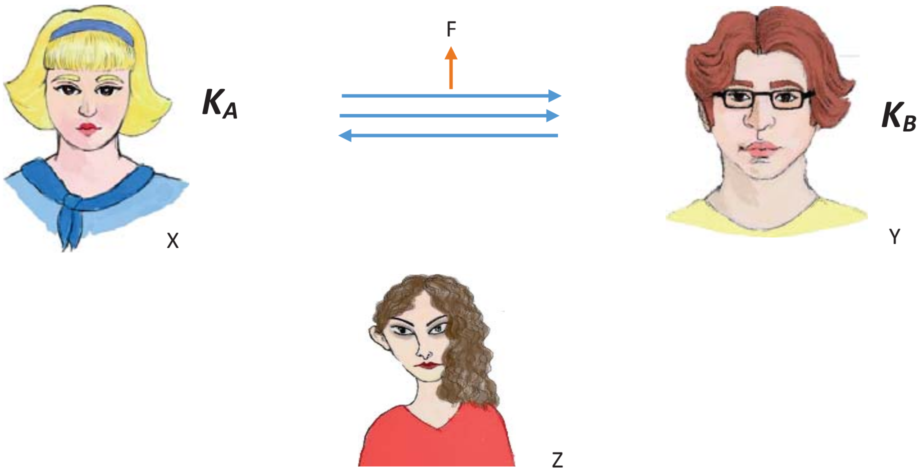

Figure 1.5 The Model for Generating Secret Key

result is the equivalent of the Slepian-Wolf theorem. If Alice and Bob then apply another random binning with rate R_k on the X_n and if $R_f + R_K < H(X)$, the output of this process is uniformly distributed and independent of the output of the first random binning. To achieve this result we have used theorem 2 assuming $X_1 = X_2 = X$ and no X_3.

1.3 Preliminary Concepts in Generating Secret Keys for non-i.i.d Cases

As mentioned earlier, the results of generating a secret key in the i.i.d cases are proved using typicality arguments, and therefore cannot be employed in non-i.i.d models. In this part the main tools to study non-i.i.d models have been explored.

1.3.1 Information Spectrum and Smooth Entropies

Let X be defined as a general random variable with probability distribution of P_X. The information spectrum of the random variable X is the probability distribution of the random variable $\log \frac{1}{P_X}$ (this latter random variable is usually called the *self-information* of X). Two points of the information spectrum are of importance in this regard: $H_{max}(P_X) = \max \log \frac{1}{P_X}$, and $H_{min}(P_X) = \min \log \frac{1}{P_X}$ (see Figure 1.7). Loosely speaking, $H_{max}(P_X)$ is related to the number of bits required to reconstruct the random variable X. As an example, if the minimum probability of a discrete random variable is $\frac{1}{16}$, in the worst-case scenario, X will have 16 realizations each with probability $\frac{1}{16}$. So, the random variable can be defined with 4 bits. In the same vein, $H_{max}(P_{XY}|P_Y)$ is the number of required bits to reconstruct X while having perfect information of Y.

On the other hand, $H_{min}(P_X)$ is roughly related to the number of intrinsic secure bits that can be extracted from random variable X. As discussed in Definition 7 and 8, a secret key has to have a uniform distribution. So for example, when a random variable X is distributed uniformly over its four realizations, two secure bits can be extracted. Let X be distributed over ten realizations. Except one realization with probability $\frac{1}{4}$, the probability of the rest for each realization is $\frac{1}{12}$. Although X is not uniform, a uniform random variable can easily be produced from X by bundling some realizations of X, as depicted in Figure 1.6 – in this case, each mass point of the probability distribution of the new variable equals the maximum of $p(X)$. Even when bundling cannot produce a completely uniform random variable, the result of bundling, over multiple samples, can be made close to the uniform distribution. This closeness is usually measured by the statistical distance and will be discussed more in 1.3.2. As intuition suggests, a form of $H_{max}(P_{XY}|P_Y)$ is a fundamental term in the information reconciliation phase where Bob (with access to Y) needs to reconstruct Alice's signal (X). Also, we will show that $H_{min}(P_{XZ}|P_Z)$ appears in the privacy amplification phase where Alice and Bob need to extract random bits form X while Eve has side information represented by (Z).

More random bits can be generated in the privacy amplification phase, and fewer bits can be sent in the reconciliation phase, if a small amount of error can be tolerated. This leads to considering probability distributions that are not identical, but statistically close to

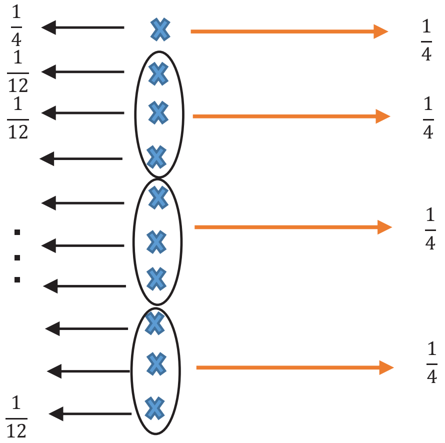

Figure 1.6 Converting random variable X to a uniform random variable

the true distribution. This is the core idea of smooth entropies [8], [9], [10]. As an example, by ignoring the smallest probabilities of a random variable, the new H_{max} can move slightly to the left of the initial H_{max} of the true distribution depicted in Figure 1.7. The new H_{max} is called the *smooth maximum entropy.* Formally, conditional maximum entropy and conditional smooth maximum entropy are defined as follows.

Definition 1.

$$H_{max}(P_{XY}|P_Y) = -\log \min_{x\in\mathcal{X}y\in\mathcal{Y}} \frac{P_{XY}}{P_Y} \tag{1.12}$$

$$H^{\epsilon}_{max}(P_{XY}|P_Y) = \min_{Q_{XY}\in\mathcal{B}^{\epsilon}(P_{XY})} H_{max}(Q_{XY}|Q_Y) \tag{1.13}$$

where $\mathcal{B}^{\epsilon}(P_{XZ})$ is the set of all non negative functions over $\mathcal{X}\times\mathcal{Y}$ such that $Q_{XY} \leq P_{XY}$ and $d(Q_{XY}, P_{XY}) \leq \epsilon$, where d is the statistical distance.

Similarly, smooth minimum entropy is defined by cutting down the largest probabilities of a random variable [8]. With this process the new H_{min} will be placed on the right side of the H_{min} of the true distribution. Heuristically more random bits can be generated by smooth minimum entropy, at the penalty that, with some small probability ϵ, the randomness extraction process will completely fail. The formal definitions for minimum entropy and conditional smooth minimum entropy are as follws.

Definition 2.

$$H_{min}(P_{XZ}|P_Z) = -\log \max_{x\in\mathcal{X} z\in\mathcal{Z}} \frac{P_{XZ}}{P_Z} \tag{1.14}$$

$$H^{\epsilon}_{min}(P_{XZ}|P_Z) = \max_{Q_{XZ}\in\mathcal{B}^{\epsilon}(P_{XZ})} H_{min}(Q_{XZ}|Q_Z) \tag{1.15}$$

Not only are information spectrum methods useful in the single-shot scenario, but we can extend the techniques to a sequence of length n. To study asymptotic behavior of a general sequence W_n, two fundamental probabilistic operations, i.e, limit superior and limit inferior (Figure 1.8) are defined:

$$P - \limsup_{n\to\infty} W_n \equiv \inf\{\alpha| \lim_{n\to\infty} Pr\{W_n > \alpha\} = 0\} \tag{1.16}$$

$$P - \liminf_{n\to\infty} W_n \equiv \sup\{\beta| \lim_{n\to\infty} Pr\{W_n < \beta\} = 0\} \tag{1.17}$$

1.3.2 Randomness Extractors in Privacy Amplification

Randomness extractors are used in the privacy amplification phase. They take as input the common sequence shared between Alice and Bob at the end of the information reconciliation phase, and output a uniformly random (from the eavesdropper's perspective) and usually shorter sequence to be used as a secret key. There are two types of extractors: deterministic and seeded. In the former type, a fixed function is employed to extract secret bits from the known random variable. It can be observed that it is not possible to have a deterministic extractor for general sources, as the following lemma, proved in [11] shows.

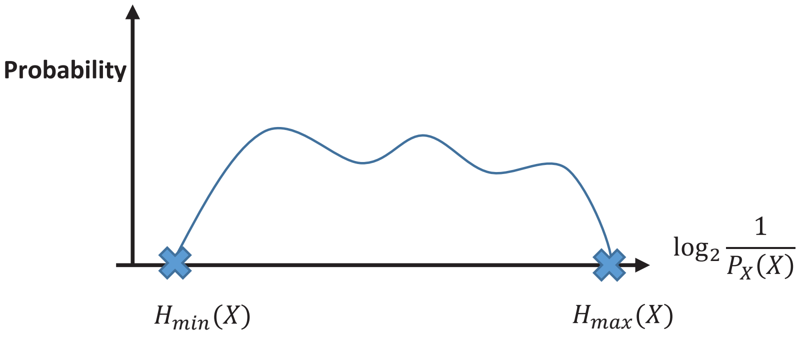

Figure 1.7 H_{min} and H_{max} for random variable X

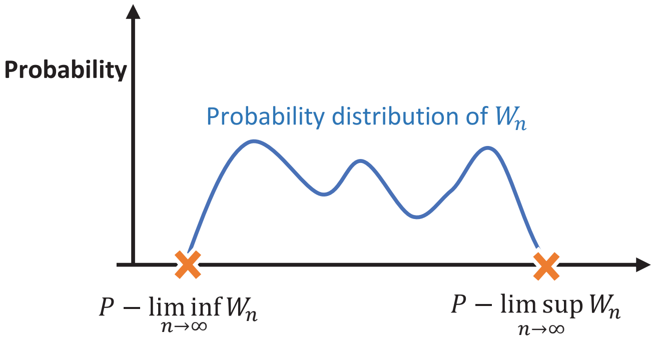
Probability
Probability distribution of W_n
$P - \liminf_{n\to\infty} W_n$
$P - \limsup_{n\to\infty} W_n$

Figure 1.8 limsup and liminf in probability

Lemma 1. *For any deterministic extractor $E : \{0,1\}^n \to \{0,1\}$, there exists a random variable X with $H_{min}(X) \geq n-1$ such that $E(X)$ is constant.*

To overcome this problem, seeded extractors are used instead. A little extra randomness called *seed* is employed in this type of extractor.

Definition 3. Strong seeded extractor

A function E: $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^u$ is a (k,ϵ)-strong seeded extractor if for every random variable X defined on $\{0,1\}^n$ with $H_{min}(P_{XZ}|P_Z) \geq k$ and seed random variable R which is uniformly distributed over $\{0,1\}^d$, the statistical distance between $P_{E(X,R)RZ}$ and

$P^u_{unif}P_{RZ}$ is less than ϵ, where P^u_{unif} is the uniform probability distribution over $\{0,1\}^u$ and Z represents the eavesdropper's side information.

It was shown in [12] that such a function E can be constructed by two-universal hash functions.

Definition 4. Two-universal hash functions *[13]*

A family of functions $\mathcal{F} = \{f : \mathcal{X} \to \{0,1\}^u\}$ is two-universal if

$$\forall x \neq x\prime \qquad P_{f\in\mathcal{F}}[f(x) = f(x\prime)] \leq 2^{-u}, \tag{1.18}$$

where the probability is with respect to the uniformly random choice of f from the family $\mathcal{F}$.

The leftover hash lemma establishes a connection between strong seeded extractors and hash functions.

Lemma 2. Leftover hash lemma [13]:

There exists a function chosen uniformly by seed R from two-universal family $\mathcal{F}$ which can be considered as a $(k, \frac{1}{2}2^{\frac{u-k}{2}})$-strong seeded extractor for random variable X with $H_{min}(P_{XZ}|P_Z) \geq k$.

The left over hash lemma directly results in

$$d(P_{f_R(X)RZ}, P^u_{unif}P_ZP_R) \leq \frac{1}{2}\sqrt{2^{u-H_{min}(P_{XZ}|P_Z)}} \tag{1.19}$$

As intuition suggests and as discussed earlier, the minimum entropy plays an important role in the privacy amplification phase.

1.3.3 Simple Binary Hypothesis Testing

The goal of a binary hypothesis test is to map an observation into either H_0 (null hypothesis) or H_1 (alternative hypothesis). This test specifies a rejection region $\mathcal{C}$ where the decision is made to reject the null hypothesis. If the distribution of a vector variable is of interest, the following binary hypothesis test can be defined;

$$\begin{aligned} H_0 &: X \sim P_\theta,\ \theta \in \Theta_0 \\ H_1 &: X \sim P_\theta,\ \theta \in \Theta_1 \end{aligned} \tag{1.20}$$

If the distribution is defined completely with the hypothesis, $(\Theta_0 = \{\theta_0\}\Theta_1 = \{\theta_1\})$, it is called simple, otherwise it is composite. So, in the simple binary hypothesis test P_{θ_0} is the distribution of X under the null hypothesis and P_{θ_1} is the distribution of X under the alternative hypothesis. A test function $\Phi(x) \in \{0, 1\}$ can be defined such that if its value is 0, the null hypothesis has been decided which means $x \in \mathcal{C}^c$, otherwise, if $\Phi(x) = 1$, the alternative hypothesis has been accepted, $x \in \mathcal{C}$.

Two types of error can be explored in a hypothesis test: type-I and type-II. Type-I error (or false alarm) occurs when H_0 is true but the test chooses H_1. So, in the simple hypothesis test

$$P_{FA} = P_{\theta_0}(\Phi(X) = 1) = \sum_x P_{\theta_0}\Phi(x) = E_{P_{\theta_0}}[\Phi(X)] \tag{1.21}$$

Type-II error (or missed detection) happens when H_1 is true but H_0 is decided by the test. So, in the simple hypothesis test

$$\begin{aligned} P_{MD} = P_{\theta_1}(\Phi(X) = 0) &= \sum_x P_{\theta_1}(1 - \Phi(x)) \\ &= E_{P_{\theta_1}}[1 - \Phi(X)] \end{aligned} \tag{1.22}$$

Although finding a region $\mathcal{C}$ with $P_{FA} = 0$ and $P_{MD} = 0$ is desirable, such a region does not exist unless P_{θ_0} is singular with respect to P_{θ_1}. In other words, by reducing the probability of missed detection in a simple hypothesis test, the probability of false alarm would increase. Hence, with a fixed tolerable amount of probability of false alarm, ϵ, a test with minimum probability of missed detection is desirable. Such a test is called *the most* *powerful test of size* ϵ and the infimum of the probability of missed detection is denoted by $\beta_\epsilon(P_{\theta_0}, P_{\theta_1})$. The Neyman Pearson lemma shows that such a test exists and it is given by the following rejection region

$$\mathcal{C} = \{x|\Phi(x) = 1\} = \{x|\frac{L(\theta_0)}{L(\theta_1)} \leq K\} \tag{1.23}$$

where L denotes likelihood function and K is a constant that can be calculated from $P_{FA} = \epsilon$.

CHAPTER 2. SECRET COMMON RANDOMNESS FROM ROUTING METADATA IN AD-HOC NETWORKS

2.1 Introduction

Automatic key establishment between two devices in a network is generally performed either by public-key-based algorithms (like Diffie-Hellman [14]), or by encrypting the newly-generated key with a special *key-wrapping key* [15]. However, in addition to the well-established, well-investigated keying information exchange, one additional aspect of key establishment is often understated: to ensure the security of the application it serves, the newly generated secret key has to be truly random. While minimum standards for software-based randomness quality are generally being enforced [16], many applications rely on often costly hardware-based *true random generators* [17]. Sources of randomness employed by true random number generators vary from wireless receivers and simple resistors to ring oscillators and SRAM memory.

In this chapter, we build upon the observation that a readily-available source of randomness is usually neglected: the network dynamics. Indeed, by their very nature, communication networks are highly dynamic and largely unpredictable. Their randomness is usually evident in easily-accessible networking metadata such as traffic loads, packet delays or dropped-packet rates. However, as the main focus of our work is on mobile ad-hoc networks (MANETs), the source of randomness we shall discuss here is one that is specific to infrastructure-less networks: the routing information itself. Another interesting feature of the routing information, in addition to its randomness, is that it can easily be made available to the devices that took part in the routing process, but it is usually unavailable to those devices that were not part of the route. This idea opens the door to a whole new class of applications: with the proper routing protocol, the routing information could be used for establishing *secret common randomness* between any two devices in a mobile ad-hoc network. This common randomness could then be further processed into true common randomness, and used as secret keys.

Common randomness was pioneered in [5, 6, 18], where it is shown that if two parties, Alice and Bob, have access to two correlated random variables (RVs) X' and Y' respectively, (in either the source or the channel models), a secret key can be established between them through public discussions and random-binning-like (e.g. hashing) operations. The key should remain secret from an adversary eavesdropper (Eve) who overhears the public discussions, and possesses side information (in the form of a third RV Z) correlated with that available at Alice and Bob. Common-randomness-based key establishment generally consists of three phases. First, Alice and Bob have to agree on two other RVs X and Y, such that $H(X|Y) < H(X|Z)$ and $H(Y|X) < H(Y|Z)$, where $H(\cdot)$ is the standard Shannon entropy. This part is sometimes called *advantage distillation*. Next, Alice and Bob (and also Eve) sample their respective random variables a large number of times, producing sequences of values. Then Alice and Bob exchange further messages (over a public channel) to agree on the same single sequence of values – this phase is the *information reconciliation*. Finally, because the agreed-upon sequence is not completely unknown to Eve (Eve can sample her variable Z synchronously with Alice and Bob), Alice and Bob run a randomness extractor on it, to produce a secret key (a shorter sequence) which, from Eve's perspective, is uniformly distributed over its space – this is the *privacy amplification phase*. The ideas of [5, 6] have been recently applied to secret key generation in wireless systems, where secure *common randomness* is attained by exploiting reciprocal properties of wireless channels or other auxiliary random sources in the physical layer [19, 20, 21, 22, 23, 24, 25, 26, 27]. One noteworthy observation is that, while the work of [5, 6, 18] considers an information-theoretic approach, in practice Alice and Bob do not usually have access to large numbers of values drawn from their random variables, but rather to only one or a few values. To address this issue, [9] shows that for such single-shot scenarios, the smooth minimum entropy provides tight upper and lower bounds on the achievable size of the secret key.

In MANETs, the lack of infrastructure, the nodes' mobility and the fact that packets are routed by nodes, instead of fixed devices, have resulted in the need for specialized routing protocols, like the ad-hoc on-demand distance vector AODV routing, or the dynamic source routing (DSR) [28]. For our secret-common-randomness-extraction purposes, DSR appears to be a good candidate, and will be the object of this work. Indeed, for generating secret common randomness between two separated nodes in the network, they must have some shared and extractable information. Among other routing protocols in ad hoc networks, DSR has this primary feature. Namely, DSR contains two main mechanisms – Route Discovery and Route Maintenance – which work together to establish and maintain routes from senders to receivers. The protocol works with the use of explicit *source routing*, which means that the ordered list of nodes through which a packet will pass is included in the packet header. It is sets of these routing lists that we shall show how to process into secret keys shared between pairs of nodes.

Our contributions can be summarized as follows:

1. We show that the randomness inherent in an ad-hoc network can be harvested and used for establishing secret keys between pairs of nodes that participate in the routing process.

2. We provide a very practical algorithm for establishing such secret common randomness, based on the DSR protocol, and we calculate a lower bound and an upper bound on the achievable number of shared secret bits, using an adversary's beliefs.

3. We simulate a realistic ad-hoc network in OPNET Modeler, and show that within only ten minutes, thousands of secret bits can be shared between different node pairs.

The rest of this chapter is organized as follows. Those parts of the DSR protocol that are essential for understanding our algorithm are examined in Section 2.2. In Section 2.3, we describe the system model and state our assumptions. Section 2.4 describes our proposed key establishment algorithm. Simulation results obtained with OPNET Modeler are presented and discussed in Section 3.7.

2.2 Dynamic Source Routing

Dynamic source routing (DSR) [28] is one of the well-established routing algorithms for ad-hoc networks. Under this protocol, when a user (the sender) decides to send a data packet to a destination, the sender must insert the *source route* in a special position of the packet's header, called the *DSR source route option*. The *source route* is an ordered list of nodes that will help relay the packet from its source to its destination. The sender transmits the packet to the first node in the *source route*. If a node receives a packet for which it is not the final destination, the node will transmit the packet to the next hop indicated by the *source route*, and this process will continue until the packet reaches its destination.

To obtain a suitable source route toward the destination, a sender first searches its own *route cache*. The *route cache* is updated every time a node learns a new valid path through the network (whether or not the node is the source or the destination for that path). If no route is found after searching the route cache, the sender initiates the *route discovery* protocol. During the route discovery, the source and destination become the *initiator* and *target*, respectively.

As a concrete example, suppose node 1 in Figure 2.1 wants to send packets to node 5. Initially, node 1 does not have any route toward node 5, and thus node 1 initiates a route

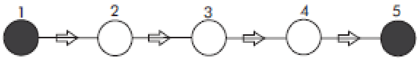

Figure 2.1 Communication among node 1 and 5

discovery by transmitting a single special local broadcast packet called *route request*. The *route request option* is inserted in the packet's header, following the IP header. To send the route request, the source address of the IP header must be set to the address of the initiator (node 1), while the destination address of IP header must be set to the IP limited broadcast address. These fields must not be changed by the intermediate nodes processing the route request. A node initiating a new route request generates a new identification value for the route request, and places it in the ID field of the route request header. The route request header also contains the address of the initiator and that of the target. The route request ID is meant to differentiate between different requests with the same initiator and target – it should be noted here that the same request may reach an intermediate or destination node twice, over different paths. Each route request header also contains a record listing the address of each intermediate node through which this particular copy of the route request has been forwarded. In our example, the route record initially lists only the address of the initiator node 1. As the packet reaches node 2, this node inserts its own address in the packet's route record, and broadcasts it further, and so on, until the packet reaches the target node 5, at which point its route record contains a valid route (1-2-3-4-5) for transmitting data from node 1 to node 5.

As a general rule, recent route requests received at a node should be recorded in the node's *route request table* – the sufficient information for identifying each request is the tuple (initiator address, target address, route request ID). When a node receives a route request packet, several scenarios can occur. First, if the node is the target, it sends a *route* *reply* packet to the initiator, and saves a copy of the route (extracted from the route request route record) in a table called the *route cache*. Second, if the node has recently seen another route request message from this same initiator, carrying the same id and target address, or if the node's own address already exists in the route record section of the route request packet (the same request reached the node a second time), this node discards the route request. Third, if the request is new, but the node is not the target, the node inserts its address in the packet's route record, and broadcasts the modified packet. Fourth, if a route exists to the target address in the node's route cache, the node sends the route reply.

In our example in Figure 2.1, node 5 constructs a *route reply* packet and transmits it to the initiator of the route request (node 1). The source address in the IP header of the route reply packet is set to the IP address of the sender of the route reply (node 5). In our example, node 5 is also the target. But this need not occur. Under the DSR protocol, it is possible that an intermediate node (who is not the target of the route request) already has a path to the target in its route cache. Then it is this node that transmits the route reply back to the initiator, and it is its IP address that gets inserted in the source IP address part of the route reply packet's header. The route reply packet header also contains a *route record*. This route record starts with the address of the first hop after the initiator and ends with the address of the target node (regardless of whether the node that issues the route reply is the target or not). In our example, the route record contained in the route reply packet is (2, 3, 4, 5). Including the address of the initiator node 1 in the route record would be redundant, as the address of node 1 is already included as the destination address in the IP header of the route reply packet. The combination of the route record and destination address in the IP header is the *source route* which the initiator will use for reaching its target. It is also noteworthy that network routes are not always bidirectional. That is, it may not always be possible for node 5 to send its route reply to node 1 using a route obtained by simply inverting the source route. In the more general case, node 5 has to search its own route cache for a route back to node 1. If no such path is found, node 5 should perform its own route discovery for finding a *source route* to node 1.

2.3 System Model

Mobile ad-hoc networks (MANETs) consist of mobile nodes communicating wirelessly with each other, without any pre-existing infrastructure. We consider a *bidirectional* MANET employing dynamic source routing (DSR), in which the nodes (corresponding to the mobile devices of the network's users) are moving in a random fashion in a pre-defined area. The bidirectional network assumption is usually a practical one, especially when all the nodes in the network belong to the same class of devices (e.g. smart phones)[1].

According to the route discovery protocol outlined in Section 2.2, every single node in the network is assumed equally likely to be the initiator of a route request packet, at any given time. Furthermore, we assume that the target of any route request is uniformly distributed among the remaining nodes. Any route discovery instance will return a path through the network (the source route), of a given length. The length of a returned path is distributed according to a probability distribution that depends on all the parameters of the network. Deriving a model for this probability distribution, based on the network parameters, is outside the scope of this work. Hence, in the remainder of this chapter, we shall assume that all nodes have access to such an (empirically-derived) probability distribution over the path lengths. That is, if we denote the random variable describing the length of some path r by L_r, then we assume that all the nodes have access to the prior $p(L_r = l)$, for $l = 2, 3, \ldots$. For our experiments, we run our simulation for a long time, and derive $p(L_r = l)$ by counting the paths of equal length. We also assume that *all paths of the same length are equally probable.*

[1] It should be noted that our algorithm should work (albeit with some reduction in performance) even if the network is not bidirectional. In this case, the route request ID needs to be inserted in the route reply packet. The reduction in performance for this scenario follows from the security considerations – namely, more nodes are involved in the routing mechanism, and hence have access to the source route.

To express this notion, denote the random variable that samples a path (or a partial path) by R. Then we can write $p(R = r|L_r = l) = \frac{1}{N_l}$ if the length of path r is l (otherwise the probability is zero), where N_l is the total number of paths of length l. This leads to $p(R = r) = \frac{1}{N_{l_r}} p(L_r = l_r)$, where l_r is the length of path r.

Our protocol, called *KERMAN* runs by making each node collect in a table all the source routes that it is part of – recall that since the network is assumed to be bidirectional, a node can extract the route request ID, the initiator and the target from the route request packet, save them in a temporary table, and then, if a route reply packet carrying a source route with the same initiator and target is observed within a pre-determined time interval, the node can associate the source route with the route request ID, and save both in a long-term table.

This mechanism brings about our security model. Since the common randomness established between two nodes by our algorithm consists of the source routes, it should be clear that several other nodes can be privy to this information. For instance, all the nodes included in a particular source route have full knowledge of this route. Moreover, it is likely that the route reply packet carrying a source route can be overheard by malicious eavesdroppers that are not part of the source route at all. Therefore, to achieve a level of security, two nodes will have to gather a large collection of source routes, such that *none of the other nodes that appear in any of the source routes in this collection has access to all the routes in the collection.* Unfortunately this is not enough, because it is still possible that one of the nodes, most likely a node that is part of many – though not all – routes in the collection, eavesdropped on all the remaining routes that it is not part of.

We deal with this problem by making an additional assumption: we assume that any two source routes are exchanged under independent and uniformly distributed network arrangements. That is, for the exchange (route discovery) of each source route, all the nodes in the network are distributed uniformly, and independently of other exchanges, in their pre-defined

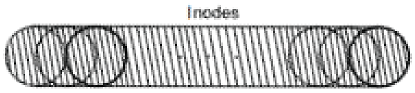

Figure 2.2 The area covered by l nodes

area. Moreover, the network remains the same for the entire duration of the route discovery and the associated data transmission. These assumptions are realistic for moderate network loads, and imply that the network nodes move around fast relative to the time between two different route discovery phases, but slow relative to the duration of a single communication session. This means that for any source route, the probability that any node which is *not* itself part of the route overhears the route (by overhearing a route reply or a data packet) is only a function of the network parameters. In the remainder of this section, we show how to compute the probability that an eavesdropper Eve knows a source route of which it is not part.

Denote the binary random variable encoding whether an eavesdropper Eve overhears a source route r by $K_{Eve}(r)$. Then $p(K_{Eve}(r) = 1)$ depends on: (a) Eve's reception radius, (b) the total area of the network (all the places where Eve could be during the communication session corresponding to source route r), and (c) the length of the path. The computation is described in Figure 2.2, where it can be observed that the worst-case scenario for a path of length l is when all the l nodes are arranged in a straight line. In this case, we can use the following worst-case approximation (obtained by first calculating the area of a circular segment):

$$p(K_{Eve}(r) = 1 | L_r = l) = \frac{\text{Shaded area in Figure 2.2, where circles have radius } d_e}{\text{Total network area}}$$
$$= \frac{l\pi d_e^2 - 2(l-1)d_e^2(\frac{\pi}{3} - \frac{\sqrt{3}}{4})}{S_{total}} = \frac{d_e^2(1.91 \cdot l + 1.23)}{S_{total}}, \qquad (2.1)$$

where d_e is the maximum eavesdropping range (the radius of the circles in Figure 2.2), which is assumed the same for each of the nodes (all nodes transmit with the same power, using isotropic antennas), and S_{total} is the total area of the pre-defined location where the nodes can move.

Finally, for brevity of presentation in the current version of this work, two additional assumptions are made: the attackers are purely passive eavesdroppers (as attackers – otherwise, they are allowed to initiate well-behaved communication, just like any other node), and they do not collude. Dealing with active and colluding attackers is the subject of future work.

2.4 Proposed Algorithm

In this section we introduce KERMAN, a Key-Establishment algorithm based on Randomness harvested from the source routes in a MANET employing the DSR algorithm. To establish secret common randomness between two nodes in the MANET, KERMAN uses the standard sequence of three steps outlined in Section 2.1: advantage distillation, information reconciliation and privacy amplification.

2.4.1 Advantage Distillation

To accomplish advantage distillation, every node in the network has to maintain a new table called the *Selected Route Table*, or SRT. The SRT contains those source routes that include that node's address, and *for which the route's destination and route-reply sender do not coincide*. To demonstrate how the SRT is built, we consider the following example. Take the scenario in Figure 2.3, in which node 1 and 6 are the source and the destination, respectively. Since node 1 does not have any route to node 6, it generates and broadcasts a route request packet. Assume that the id of this packet is 14, which means that this is the fourteenth attempt that node 1 makes to reach node 6. Further assume that the route request first reaches node 5 over the path 1-2-3-4-5. As seen in Figure 2.3, node 5 will generate the route reply from its own route cache (because we assumed that node 5 already knows how to reach node 6). The transmission path of the route reply from node 5 to node 1 is the upper path in Figure 2.3 (that is, 5-4-3-2-1), and is consistent with a bidirectional network. Each intermediate node that receives this route reply inserts the source route in their own SRT. The SRT has three columns dubbed *RID*, *partial route* and *full route* respectively. RID is a tuple that consists *(Source IP, Destination IP, route request ID, route-reply-sender IP).* In our scenario, nodes 1, 2, 3, 4 and 5 will all record an entry in their respective SRTs, with the RID 1-6-14-5. The intermediate nodes (2, 3 and 4) can obtain the route request ID by searching their own *route request tables* as discussed in Section 2.2. The *partial route* field of the SRT entry identifies those other nodes that are supposed to have this particular route in their SRT – in this case, nodes 1, 2, 3, 4 and 5. The *full route* field is the entire route from source to destination, which will be used for data transmission (1,2,3,4,5,6 in this case). The SRTs of the nodes 1, 2, 3, 4 and 5 have the same following entry:

RID	Partial Route	Full Route
1-6-14-5	1-2-3-4-5	1-2-3-4-5-6

It should be noted that, because node 6 did not directly hear the route request from node 1, it has no way of determining the route request ID in the RID, and this is why it cannot store this entry in its SRT, although it will most likely learn the source route from the received data packets that follow the route discovery phase. Thus, although node 6 will not use this specific route for establishing a secret key with one of its peers, when discussing the security of the established secret common randomness between two other peers sharing this route, node 6 will be considered a possible eavesdropper (i.e. node 6 will be assumed to have full knowledge of the *full route*). Each full route in a nodes' SRT is only available to a limited number of nodes in the network, i.e., those nodes which are included in in the source (full)

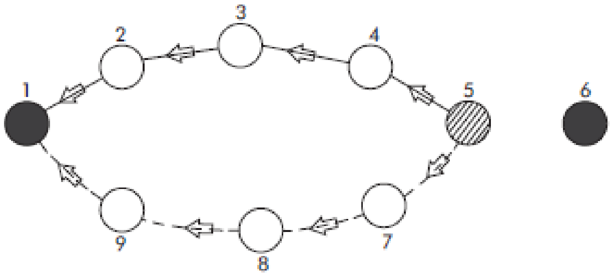

Figure 2.3 Example for proposed algorithm

route, along with some nodes who are not part of the source route but happen to overhear the route request and route reply exchange. The following proposition states that SRT entries are unique in the whole network.

Proposition 1. *If two nodes have the same RID in their own SRTs, then the full routes associated with this RID in two SRTs are exactly the same.*

Proof. Based on the DSR protocol [28], in the phase of processing a received route request, several steps must be performed in a well-defined order. The step consisting of the search in the route request table is done before the phase of sending route reply from the route cache. But if, while searching the route request table, a node finds that it has received this route request before, the node must discard the route request packet. Hence, an intermediate node can initiate the route reply only in response to the first route request, and will ignore all subsequent route requests with the same ID, source and destination. Since the SRT only contains routes in which the destination is different than the route reply sender, it is not possible that multiple route replies originate from the same node in response to the same route request, even if the route request was received multiple times, via different paths. Now, although two different route replies in response to the same route request can originate at different nodes, (for example, in Figure 2.3 node 7 also knows a path to node 6 and initiates a route reply), the RIDs corresponding to these route replies contain the IP of the route reply sender, and hence are different. Note that if the SRT contained routes for which the destination and the route-reply sender coincide, multiple routes could be associated with the same RID – this is an undesirable effect, and needs to be prevented by properly constructing the SRT. □

2.4.2 Information Reconciliation

Information reconciliation is usually a complex process, involving techniques from channel or source coding, and displaying very restrictive lower bounds on the amount of information that needs to be transmitted over a public channel [9] – these bounds can often leave very little uncertainty for an eavesdropper. Fortunately, KERMAN is particularly well-suited for information reconciliation, and only requires minimal communication overhead. This is due to the fact that in KERMAN the common randomness is based on full routes, and each full route is uniquely identified, at both parties, by its RID, thus making reconciliation simpler.

Let us assume that two nodes –call them Alice and Bob for simplicity – realize that they share a large number of routes in their SRTs. For instance, Alice could first notice that Bob is part of a large number of partial routes in her SRT, and could ask Bob to perform information reconciliation, with the purpose of eventually generating a shared secret key. Upon Bob's acceptance, Alice sends him the list of RIDs corresponding to the partial routes in Alice's SRT that include the address of Bob. Bob can then verify whether he already has the received RIDs in his SRT, and can send back to Alice only those RIDs that he could not locate. The information reconciliation is now complete. Alice and Bob share a set of full routes, which constitute their common randomness.

There is but one caveat. As mentioned in Section 2.4.1, the RIDs consist of the tuples (Source IP, Destination IP, route request ID, route-reply-sender IP) corresponding to each route request/ route reply pair. Moreover, it is possible that Alice and Bob are neither the source nor the destination, nor the route-reply sender. Thus, transmitting an RID in the clear, over a public channel, may expose up to five nodes of the route (source, destination, route-reply sender, Alice and Bob) to an eavesdropping adversary. Many practical solutions can be employed to limit the amount of information that the reconciliation leaks to potential eavesdroppers. As a starting point, several solutions are provided in [29].

But such solutions are outside the scope of this work. Instead, we take a different approach, and provide a lower bound and an upper bound on the total number of secret bits achievable by KERMAN, network-wide. For the lower bound, we consider the case when the RIDs are indeed transmitted in the clear, while for the upper bound, we consider the case where the RIDs are transmitted while being completely protected (by some hypothetical encryption mechanism) from any potential eavesdroppers. In both scenarios, however, we assume that every node in the network can see that Alice and Bob exchange RIDs – and thus any eavesdropper knows that the identities of Alice and Bob are part of the full routes used for secret key generation.

2.4.2.1 The lower bound: RIDs transmitted in the clear

Some information about the full routes is known to leak from the corresponding RIDs. But exactly how much information leaks is subject to the properties of the (Alice, Bob, route, RID) tuple. More precisely, these tuples can be divided into seven types, which can then be grouped into three different groups, according to their information-leakage behavior, as shown in Table 2.1. Group 1 consists of the cases in which the RID reveals information about a single node, in addition to Alice and Bob. Groups 2 and 3 include the cases in which the RIDs leak information about two and three nodes, respectively, in addition to Alice and Bob. In Table 2.1, A and B stand for Alice and Bob (and are interchangeable), while X and Y represent two nodes other than A and B. For example, in Group 2, type 4 , Alice is the source but destination and route replier are two distinct nodes other than Bob.

Table 2.1 Different groups and types when we send RID in clear

Group	Type	Source	Destination	RREP Sender
1	1	A	B	X
	2	A	X	B
	3	X	A	B
2	4	A	X	Y
	5	X	A	Y
	6	X	Y	A
3	7	X	Y	Z

2.4.2.2 The upper bound: RIDs completely protected

In this case, the only information that leaks to an eavesdropper in the process of information reconciliation is that the identities of Alice and Bob have to appear in every one of the full routes, the RIDs of which are being exchanged between Alice and Bob.

2.4.3 Privacy Amplification

For the purposes of this section we shall represent the full routes as sets of node identifiers, or addresses. Alice and Bob share a list of common full routes. Now Alice and Bob can construct the set $\mathcal{M} = \{m_1, m_2, \ldots, m_h\}$ where m_i (we'll call it a *trimmed route*) is produced from the full route r_i, by removing the addresses of Alice and Bob. At this point, full routes and trimmed routes are in a one-to-one correspondence. However, it is essential that the reader remembers the difference between a full route and a trimmed route.

In the next step, Alice partitions the set of trimmed routes $\mathcal{M}$ into several *disjoint* subsets $\mathcal{M}_k \subset \mathcal{M}$ of various sizes h_k, such that, for any $\mathcal{M}_k = \{m_{1,k}, m_{2,k}, \ldots, m_{h_k,k}\}$, the probability that any node in the network has knowledge of all the h_k trimmed routes is less than a small security parameter ϵ_1. This means that, with probability larger than $1 - \epsilon_1$, there exists at least one trimmed route in $\mathcal{H}$ that Eve knows nothing about – note that this is true for any identity that Eve may take (except, of course Eve cannot be Alice or Bob). It is the full route corresponding to this trimmed route (different from any node's perspective) that constitutes the randomness of the generated secret.

To extract a secret from each of the sets $\mathcal{M}_k$, Alice first represents all the *full routes* by binary strings of the same length (according to a mapping previously agreed upon by all the nodes in the network). The length of the strings is determined as the logarithm to base two of the total number of possible full routes, in a practical scenario. For example, from our simulations, we noticed that full routes are limited to 15 nodes, which means that trimmed routes are limited to 13 nodes. In a network of 50 nodes, there are thus $\binom{48}{1}3! + \binom{48}{2}4! + \ldots + \binom{48}{13}15!$ possible full routes involving Alice and Bob, where the factorial terms account for all the possible arrangements. For example, there are $\binom{48}{1}$ trimmed routes of length 1, and their corresponding full routes have length 3 (this includes the node that defines the trimmed route, Alice and Bob), and there are $3! = 6$ possible arrangements of these three nodes. This total number of possible full routes amounts to representing each full route on 78 bits. The binary sequences representing the *full routes* corresponding to the trimmed routes in $\mathcal{M}_k$ are then XORed together.

The result is inserted into a (k, ϵ_2)-randomness extractor (defined in 3), which outputs a shorter bit string s_k – the secret. The secret s_k should satisfy the (ϵ_1, ϵ_2)-security defined below.

Definition 5. *In the context of a MANET, a piece of secret common randomness s_k established between two nodes Alice and Bob is called (ϵ_1, ϵ_2)-secure if, with probability larger than $1 - \epsilon_1$, the secret s_k is ϵ_2-close to uniform from the perspective of any node in the network, except Alice and Bob.*

It has been shown in [9] that the number of completely random bits that can be extracted from a bit sequence should be upper bounded by, but very close to, the smooth min-entropy of the sequence. Thus, for the purposes of this chapter, we shall only focus on the (smooth) minimum entropy of a full route, viewed from the perspective of an eavesdropper. This minimum entropy is a good indication of the number of secret random bits that can be extracted from each set $\mathcal{M}_k$, and can be calculated according to Definition 1, where the probability distribution is that which characterizes Eve's belief about the full route. Eve's belief depends on whether the RID is sent in clear or perfectly protected.

2.4.3.1 The lower bound: RIDs transmitted in the clear

When the RIDs are communicated between Alice and Bob in the clear, Eve will be able to infer some information about the corresponding full routes agreed on by Alice and Bob. In addition, the very fact that Eve did not overhear the full route can also leak some information: longer routes are more likely to have been overheard by Eve. Thus, we are primarily concerned with the probability distribution $p(r|K_{Eve}(r) = 0, RID(r)))$, where K_{Eve} is the binary random variable encoding whether Eve knows the full route ($K_{Eve} = 1$) or not ($K_{Eve} = 0$), and $RID(r)$ is the RID corresponding to the route r. Since we already saw that the information leaked to Eve from the RID depends on the group corresponding to the tuple (Alice, Bob, route, RID) – see Table 2.1 – and since for a specific group all routes of the same length are equally probable from Eve's perspective , we can write:

$$p(r|K_{Eve}(r) = 0, RID(r))) = \begin{cases} \frac{p(L_r=l_r|K_{Eve}(r)=0,group=1)}{\binom{N-3}{l_r-3}(l_r-2)!}, & group = 1 \\ \frac{p(L_r=l_r|K_{Eve}(r)=0,group=2)}{\binom{N-4}{l_r-4}(l_r-2)!}, & group = 2 \\ \frac{p(L_r=l_r|K_{Eve}(r)=0,group=3)}{\binom{N-5}{l_r-5}(l_r-2)!}, & group = 3 \end{cases} \tag{2.2}$$

where N is the total number of nodes in the network, the random variable L_r represents the length of the full route (l_r is the actual length of route r), and the denominators stand for the possible number of routes of length l_r, and belonging to group i, with $i \in \{1, 2, 3\}$. For example in the case of group 1, the number of full routes with length l_r in which Eve already knows the identities of three nodes (see Table 2.1) is equal to $\binom{N-3}{l_r-3}(l_r-2)!$. This is because the unknown l_r-3 nodes can be picked in $\binom{N-3}{l_r-3}$ ways, and then all the nodes, except source and destination, can be arranged in $(l_r-2)!$ ways.

It now remains to compute $p(L_r = l_r | K_{Eve}(r) = 0, group = 1)$. We can write:

$$p(L_r = l_r | K_{Eve}(r) = 0, group = i) = \\ = \frac{p(L_r = l_r | group = i) p(K_{Eve}(r) = 0 | L_r = l_r, group = i)}{\sum_l p(L_r = l | group = i) p(K_{Eve}(r) = 0 | L_r = l, group = i)}, \tag{2.3}$$

where

$$p(L_r = l_r | group = i) = \frac{p(L_r = l_r) p(group = i | L_r = l_r)}{\sum_l p(L_r = l) p(group = i | L_r = l)}. \tag{2.4}$$

Now $p(L_r = l)$ is derived empirically from our simulation results, as explained in Section 2.3, while $p(group = i | L_r = l)$ can be written as:

$$p(group = i | L_r = l) = \begin{cases} \frac{6}{(l_r)} \frac{1}{(l_r-1)}, & i = 1 \\ \frac{6}{(l_r)} \frac{(l_r-3)}{(l_r-1)}, & i = 2 \\ \frac{2(l_r-3)}{(l_r)} \frac{(l_r-4)}{(l_r-1)}, & i = 3. \end{cases} \tag{2.5}$$

To explain (2.5) consider, for example, $p(group = 2 | L_r = l_r) = p(type = 4 | L_r = l_r) + p(type = 5 | L_r = l_r) + p(type = 6 | L_r = l_r)$ (see Table 2.1). The three probabilities on the right hand side are all equal. Let's now look at $p(type = 4 | L_r = l_r)$. Consider a given route of length l_r, where the component nodes are indexed as 1 (source), $\ldots, l_r$ (destination), and imagine that Alice, Bob and the route-reply node (RR) pick uniformly randomly amongst these indices, with the caveat that Alice cannot be equal to Bob. Then $p(type = 4 | L_r = l_r) = p(Alice = 1) p(Bob \neq RR \wedge Bob \in \{2, \ldots, l_r - 1\}) + p(Bob = 1) p(Alice \neq RR \wedge Alice \in \{2, \ldots, l_r - 1\}) = 2\frac{1}{l_r}\frac{l_r-3}{l_r-1}$.

Finally, whether Eve has eavesdropped a certain route or not does not depend on the roles of Alice and Bob in the path, nor on the identity of the route-reply sender. So we can write the last remaining term of (2.3) as $p(K_{Eve}(r) = 0 | L_r = l_r, group = i) = p(K_{Eve}(r) = 0 | L_r = l_r) = 1 - p(K_{Eve}(r) = 1 | L_r = l_r)$, which can be computed from (2.1).

2.4.3.2 The upper bound: RIDs completely protected

When the RID is perfectly protected, the probability of a certain route, from Eve's perspective, depends solely on its length.

Since all unknown routes of a given length are equally probable from Eve's perspective, we can write

$$p(r|K_{Eve}(r)=0)) = \frac{p(L_r=l_r|K_{Eve}(r)=0)}{\binom{N-2}{l_r-2}l_r!}, \tag{2.6}$$

where the denominator represents the number of all possible routes of length l_r that contain Alice and Bob (similarly to (2.2)). Now we can write the probability on the right-hand side as:

$$\begin{aligned} &p(L_r=l_r|K_{Eve}(r)=0) = \\ &= \frac{p(L_r=l_r)p(K_{Eve}(r)=0|L_r=l_r)}{\sum_l p(L_r=l)p(K_{Eve}(r)=0|L_r=l)}. \end{aligned} \tag{2.7}$$

In the right-hand side of (2.7), $p(L_r=l_r)$ is the empirically-derived probability distribution discussed in Section 2.3, while $p(K_{Eve}(r)=0|L_r=l_r) = 1 - p(K_{Eve}(r)=1|L_r=l_r)$ can be computed from (2.1).

2.4.3.3 The partitioning algorithm

Now the remaining question is how many subsets $\mathcal{M}_k$ we can form. To solve this problem, for any pair of nodes we organize the full set of all trimmed routes $\mathcal{M}$ as a *selection matrix*. In the *selection matrix*, a row corresponds to one of the trimmed routes in $\mathcal{M}$. A column corresponds to a node's address.

There are 48 columns (one for each node in the MANET, except Alice and Bob). Each entry in the matrix is the probability that the node in the respective column knows the full route corresponding to the respective row. The selection matrix can be represented as follows:

$$\begin{array}{c}
 \\ m_1 \\ m_2 \\ \vdots \\ m_h
\end{array}
\begin{array}{c}
\begin{array}{cccc} \textit{node } 1 & \textit{node } 2 & \dots & \textit{node } t \end{array} \\
\left(\begin{array}{cccc}
a_{11} & a_{12} & \dots & a_{1t} \\
a_{21} & a_{22} & \dots & a_{2t} \\
\vdots & \vdots & \ddots & \vdots \\
a_{n1} & a_{n2} & \dots & a_{nt}
\end{array}\right)
\end{array}$$

where a_{ij} is the probability that node j knows full route i. For example, when node j is a part of the full route corresponding to the trimmed route i, then $a_{ij} = 1$. Otherwise, $a_{ij} = p(K_j(i) = 1|L_i = l_i)$, where l_r is the length of route i. The partitioning algorithm consists of constructing *distinct* sub-matrices $\mathcal{M}_k$, each consisting of h_k rows of $\mathcal{M}$, such that the product of the entries in each column of $\mathcal{M}_k$ be less than ϵ_1. We shall informally call this property ϵ_1-security, and we shall use the terms *subset* and *sub-matrix* interchangeably. An optimal partition maximizes the number of sub-matrices $\mathcal{M}_k$ with the ϵ_1-security property. Here we propose a naïve partitioning algorithm.

For the upper-bound scenario (perfectly protected RIDs), we build $\mathcal{M}_1$ by selecting the first row in the selection matrix, and adding the next row in the selection matrix, until the column-wise product condition holds. Then we move to the next row, and start building $\mathcal{M}_2$, and so on, until we run out of rows in $\mathcal{M}$.

For the lower-bound scenario (RIDs sent in the clear), we perform one more step: we append to each row of selection matrix a number which indicates the group of the corresponding RID. Since min-entropy for each group is different, and the number of extractable random bits is related to the min entropy, before applying the naïve algorithm, Alice and Bob should sort their routes based on the group number. That is, routes whose RIDs place them in groups with higher min-entropy come first. Note that in a subset with routes from different groups, Alice and Bob have to consider the worst-case scenario. As a concrete example, if a subset contains routes from groups 3,3,3,2,1 and group 1 has the least min

Table 2.2 Number of subsets, obtained by the naïve algorithm with $\epsilon_1 = .001$, for RIDs sent in the clear. Total network-wide achievable number of shared secret bits, in last column.

No. of Subsets	1			2			3			4			
Group	1	2	3	1	2	3	1	2	3	1	2	3	B_{total}
No. of Pairs-naïve	203	125	3	31	16	1	4	2	0	3	1	0	862

Table 2.3 Number of subsets, obtained by the naïve algorithm with $\epsilon_1 = .001$, for protected RIDs. Total network-wide achievable number of shared secret bits, in last column.

No. of Subsets	1	2	3	4	5	6	7	B_{total}
No. of Pairs-naïve	215	75	22	6	1	0	1	$4.98 \cdot 10^3$

entropy, the group can only produce a number of random bits equal to the min entropy of group 1. This is due to the fact that the worst-case scenario is when an eavesdropper knows all routes, except that belonging to group 1 (recall that Alice and bob do not know who the eavesdropper might be).

As an alternative to the naïve algorithm, In [30] we provided a better-performing (but more complex) heuristic algorithm for calculating upper bound, that goes as follows. Starting with the original selection matrix, we inspect all sub-matrices of two rows, and check whether any of them satisfies the ϵ_1-security property. If any such disjoint sub-matrices are found, we count the corresponding subsets of rows $\mathcal{M}_k$, we update the selection matrix by removing these rows from the original selection matrix, and we go on to inspect all the sub-matrices consisting of three rows of the updated selection matrix.

So far, the algorithm seems to perform optimally. However, the main problem that prevents the algorithm from being optimal arises because in general several partially-overlapping sub-matrices can be formed at each step. For example, consider a scenario where two sub-matrices of two rows have been found to satisfy ϵ_1-security: say these sub-matrices are the one consisting of rows 2 and 6 of the selection matrix, and the one consisting of rows 2 and

8. Clearly, only one of them can be considered for privacy amplification, lest we compromise the entropy of the secret key. We now have to decide which of the two choices results in an updated selection matrix that is more likely to perform better in future partitions. For our example, if row 6 is less than row 8 (i.e. component i or row 6 is less than component i of row 8, for all i), then we should select the sub-matrix containing rows 2 and 8, because row 6 might prove more useful in the future. But because such an ordering of matrices is usually not clear-cut, we proceed to define our own *partial order*, which is essentially sub-optimal, and responsible for the sub-optimality of our heuristic algorithm.

Definition 6. Average-Column-Product Sub-Optimal Partial Order (ACP-PO): *For any two partially-overlapping sub-matrices* $\mathcal{M}_i$ *and* $\mathcal{M}_j$ *of the selection matrix, with* $\mathcal{M}_i \bigcap \mathcal{M}_j = \mathcal{M}_{ij} \neq \emptyset$*, we say that* $\mathcal{M}_i$ *is better than* $\mathcal{M}_j$ *in the ACP-PO sense, and write* $\mathcal{M}_i \prec \mathcal{M}_j$ *if the mean of the column-wise products of elements of* $\mathcal{M}_i$ *is less than mean of the column-wise products of elements of* $\mathcal{M}_j$*. We say that* $\mathcal{M}_i$ *is at least as good as* $\mathcal{M}_j$ *in the ACP-PO sense, and write* $\mathcal{M}_i \preceq \mathcal{M}_j$ *if the mean of the column-wise products of elements of* $\mathcal{M}_i$ *is less than or equal to the mean of the column-wise products of elements of* $\mathcal{M}_j$*.*

Our algorithm is illustrated by the pseudo-code fragment of Algorithm 1. The algorithm starts by checking whether at least one sub-matrix verifying the ϵ_1-security condition can be found – that is, whether the whole original selection matrix satisfies ϵ_1-security. The algorithm then finds all the sub-matrices of $SubsetSize$ rows of M that satisfy ϵ_1-security, and orders them according to the ACP-PO defined above. Recall that this partial order is only meaningful for two sub-matrices that have at least one row in common, but our algorithm orders the whole list of subsets anyway. After sorting all sub-matrices in descending ACP-PO we pick and process the first sub-matrix. We then make sure that the rows we already picked are not going to be considered again, by updating the ordered list and the selection matrix M. The algorithm will then continue to select sub-matrices from the remaining list, and when the list becomes empty, it switches the search to sub-matrices with more rows.

Algorithm 1 Heuristic Algorithm

```
$M$=Selection Matrix;
$SubsetSize = 2$;
$NumberSubsets = 0$;
$L$ =Number of rows in $M$;
while $Subsetsize \leq L$ AND $M$ satisfies $\epsilon_1$-security do
    for All combinations $M_k$ of $Subsetsize$ rows do
        if $M_k$ satisfies $\epsilon_1$-security then
            Calculate average of column-wise products;
    Sort subsets based on the average of column-wise products;
    while Not End of List do
        Select and process first subset in the ordered list and increment $NumberSubsets$;
        Delete from the list all subsets that share rows with the selected subset;
        Update $M$ by deleting all the rows in the selected subset;
    Increment $SubsetSize$;
    Update $L$;
```

To gain more insight into the algorithm's complexity, consider a case in which the initial selection matrix has h_1 rows. In the first stage, the algorithm examines $\binom{h_1}{2}$ partitions (sub-matrices), and if it finds any that satisfy ϵ_1 security, it updates the selection matrix, which will end up with $h_2 \leq h_1$ rows. The second stage inspects $\binom{h_2}{3}$ partitions, and so forth. All in all the heuristic algorithm should examine $\binom{h_1}{2} + \binom{h_2}{3} + \binom{h_3}{4} + ...$ partitions, which for most cases should be a lot less than 2^{h_1}. However, if there is no reduction in the number of rows in the first stages, the algorithm has to explore all 2^{h_1} partitions. Several simplifying solutions can be considered to avoid this situation: (1) if it is observed that over a pre-determined period of time the algorithm produces only sub-matrices with at least S_0 rows, then the algorithm can start with $SubsetSize = S_0$ rather than $SubsetSize = 2$; (2) the algorithm can test whether at least two sub-matrices are even possible, by testing whether the whole (updated) selection matrix M satisfies ϵ_1^2-security. If it does not, then the algorithm can stop searching for sub-matrices, and can process the whole selection matrix as a single sub-matrix.

2.5 Simulation Results

2.5.1 Secret Length and The Secret Bit Rate

The proposed protocol has been simulated in OPNET, using the parameters indicated in Table 2.4. This choice of parameters results in a maximum eavesdropping range of $d_e = 12$m. Each node sends packets to four random destinations. The number of full routes vs the full route length is shown in Figure 2.4, and the empirically-derived prior $p(L_r = l_r)$ looks similar.

Table 2.4 Simulation Parameters

Simulation Parameters	Value
Network Size	100m*100m
Number of Nodes	50
Simulation Duration	600(sec)
Transmit Power(w)	.005
Packet Reception-Power Threshold(dBm)	-55
Speed(meters/seconds)	uniform(.5,1)
Packet Inter-Arrival Time(seconds)	exponential(1)

As we discussed earlier, the probability distribution of the unknown full route, used in calculating the min-entropy, can be obtained from (2.2) (for the lower bound) or from (2.6) (for the upper bound).These probability distributions are given in Table 2.5.

It can be easily seen that when RID is sent in clear we have $H_{min}(r|K_{Eve}(r) = 0, group = 1) = -\log_2(0.428) = 1.22$, $H_{min}(r|K_{Eve}(r) = 0, group = 2) = -\log_2(0.13462) = 2.893$ and $H_{min}(r|K_{Eve}(r) = 0, group = 3) = -\log_2(0.0261) = 5.257$, while if the RID is perfectly protected we get $H_{min}(r|K_{Eve}(r) = 0)) = -\log_2(0.00062) = 10.66$.

In Figure 2.5 we show the number of pairs of nodes that share selection matrices, versus the number of rows in these shared matrices. Clearly, the larger the number of rows in the shared selection matrix, the higher the potential for generating more shared secret bits.

The number of subsets produced by the naïve partition algorithm for the whole network is shown in Tables 2.6 and 2.2, for $\epsilon_1 = 10^{-3}$. We also calculate the maximum achievable total

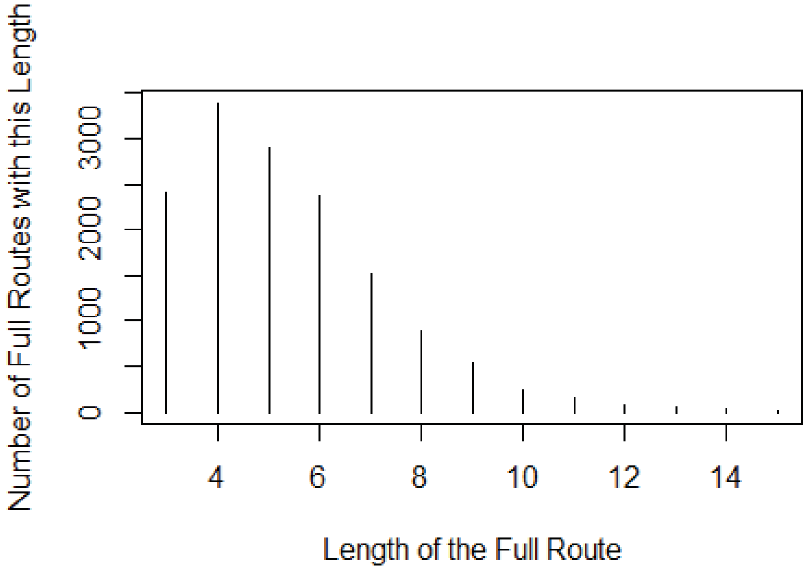

Figure 2.4 Number Of Full Routes vs. Full Route Length

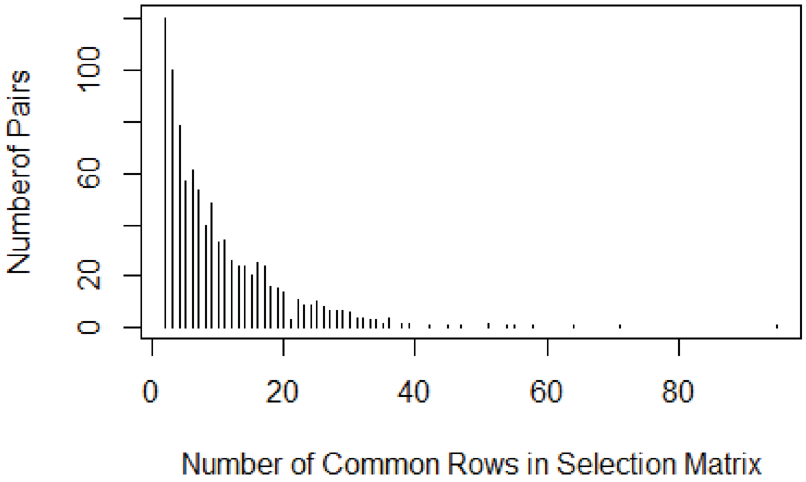

Figure 2.5 Number Of Pairs vs. Number of Rows in their shared Selection matrix

Table 2.5 Probability Distribution of an Unknown Full Route, from Eve's Perspective based on Sending RID Type

Route Length	3			4		
probability (Clear)	6.2E-4			9.0E-06		
Group (Protected)	1	2	3	1	2	3
probability (Protected)	.428	0	0	.003	0.1346	0
Route Length	5			6		
probability (Clear)	9.7E-08			1.1E-09		
Group (Protected)	1	2	3	1	2	3
probability (Protected)	2.2E-05	.001	.026	1.9E-07	8.4E-06	2E-04
Route Length	7			8		
probability (Clear)	1.1E-11			1.1E-13		
Group (Protected)	1	2	3	1	2	3
probability (Protected)	1.5E-09	6E-08	1.8E-06	1.2E-11	4.5.5E-10	1.5E-8
Route Length	9			10		
probability (Clear)	1.2E-15			9.9E-18		
Group (Protected)	1	2	3	1	2	3
probability (Protected)	1E-13	5.1E-12	1E-10	8.5E-16	3.7E-14	1E-12
Route Length	11			12		
probability (Clear)	1E-19			1E-21		
Group (Protected)	1	2	3	1	2	3
probability (Protected)	9.5E-18	4.12E-16	1.1E-14	9E-20	3.8E-18	1.05E-16
Route Length	13			14		
probability (Clear	1.6E-23			2E-25		
Group (Protected)	1	2	3	1	2	3
probability (Protected)	1.05E-21	4.5E-20	1.23E-18	1.27E-23	5.4E-22	1.48E-20
Route Length	15					
probability (Clear)	2E-27					
Group (Protected)	1	2	3			
probability (Protected)	E-25	5.7E-24	1.57E-22			

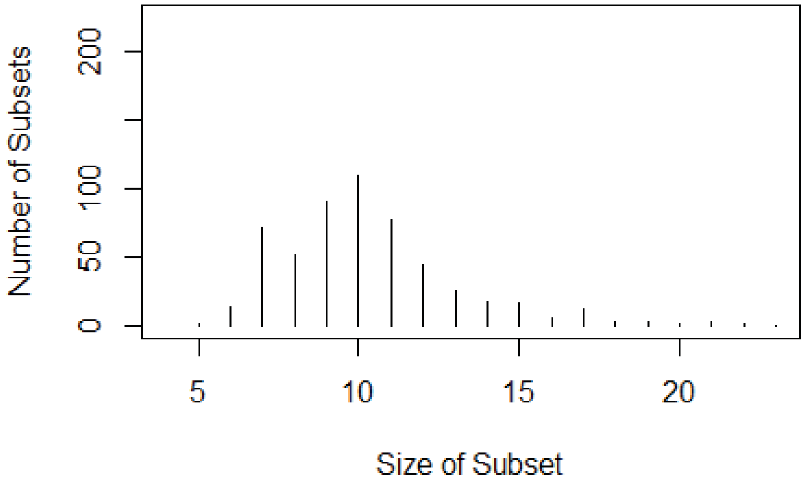

Figure 2.6 Number Of subsets of a given size (number of rows), vs. Subset size, for the naïve algorithm (Clear RID) – network-wide results.

network-wide number of shared random bits (between all the possible pairs in the network), B_{total} – this is shown in the last columns of Tables 2.6 and 2.2. For example, for $\epsilon_1 = 10^{-3}$ we have an upper bound of $B_{total} = 10.66 \cdot (215 \cdot 1 + 75 \cdot 2 + 22 \cdot 3 + 6 \cdot 4 + 1 \cdot 5 + 1 \cdot 7)$. Additionally, the numbers of subsets with a given size (number of rows) are shown for the whole network in Figure 2.6 and Figure 2.7, for the lower-bound and upper-bound scenarios, respectively.

To compare naïve and heuristic algorithms for $\epsilon_1 = 10^{-3}$, the number of sub-matrices satisfying ϵ_1-security, produced by two algorithms for the whole network in the case of protected RID, is shown in table 2.6. We have brought the simulation results for heuristic algorithm in [30] and the rest of this section has been devoted to the naïve algorithm, unless otherwise stated.

Additionally, we evaluate the *secret bit rate, relative to transmission overhead.* Since the routing information we use for the generation of secret bits comes *free* (and is normally discarded), we normalize the number of secret bits by the number of bits transmitted for the

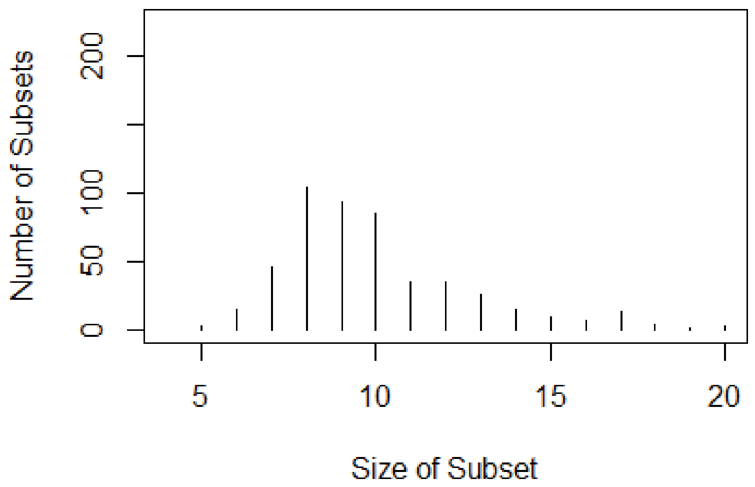

Figure 2.7 Number Of subsets of a given size (number of rows), vs. Subset size, for the naïve algorithm (Protected RID) – network-wide results.

Table 2.6 Number of subsets, obtained by the naïve and heuristic algorithms with $\epsilon_1 = .001$, when considering all full routes of length at least 3 and in the case of protected RID.

No. of Subsets	1	2	3	4	5	6	7	8	9
No. of Pairs-naïve	215	75	22	6	1	0	1	0	0
No. of Pairs-heuristic	171	70	39	25	9	2	1	2	1

purposes of information reconciliation, as in Section 2.4.2. Recall that for the reconciliation of each full route, an RID is transmitted, consisting of three node addresses and a *route-request ID*. For a network of 50 nodes, and noticing that in our simulations the route-request ID does not exceed the value of 500, the RID can be encoded on $3 \cdot \lceil \log_2(50) \rceil + 9 = 26$ bits. The additional packet header overhead is ignored here, because it is easily amortized – we could transmit many such RIDs in a single packet. The average subset size for the naïve algorithm in the case of unprotected RIDs is 9.53, and for protected RIDs it is 9.89. This implies an overhead transmission of $9.53 \cdot 26 = 238.25$ and $9.89 \cdot 26 = 257.14$ bits per subset, respectively. The *secret bit rate, relative to transmission overhead* is thus given by $1.87/238.25 = .00786$ (lower bound) and $10.66/257.14 = .0414$ (upper bound) secret bits per bit of overhead.

2.5.2 The Effects of Speed and Transmission Range

2.5.2.1 The effects of node speed

To see the effect of the nodes' speed in the number of achieved random bits, we have simulated two additional networks, with the same parameters as those in Table 2.4, except with node speeds distributed uniformly over $(1, 1.5)m/s$ and over $(1.5, 2)m/s$, respectively. Based on our simulation results, the numbers of full routes of any length in the whole network, for speeds chosen as $uniform(0.5, 1)$ (the original network), $uniform(1, 1.5)$ and $uniform(1.5, 2)$ were respectively 14544, 18768 and 19900. For fully-protected RIDs, the minimum entropies (or the numbers of secret bits that can be extracted from a full route unknown by the eavesdropper), are 10.66, 10.61 and 10.67, respectively. For the case when the RIDs are sent in the clear, the min entropies corresponding to different groups are given in Table 2.7.

Table 2.7 Size of min-entropy based on 3 different speeds in the case of clear RID. Speed 1, speed 2 and speed 3 are uniform (.5,1), uniform (1,1.5) and uniform (1.5,2), respectively.

	Speed 1	Speed 2	Speed 3
$H_{min}(r\|K_{Eve}(r) = 0, group = 1)$	1.22	1.237	1.267
$H_{min}(r\|K_{Eve}(r) = 0, group = 2)$	2.893	2.756	2.800
$H_{min}(r\|K_{Eve}(r) = 0, group = 3)$	5.257	4.988	5.075

Table 2.8 Number of node pairs vs. number of subsets for three different speeds by applying naïve algorithm with $\epsilon_1 = .001$, when RID is sent in the clear. Total network-wide achievable number of shared secret bits, in last column. Speed 1, speed 2 and speed 3 are uniform (.5,1), uniform (1,1.5) and uniform (1.5,2), respectively.

No. of Subsets	1			2			3			4			5			
Group	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	B_{total}
No. of Pairs, speed 1	203	125	3	31	16	1	4	2	0	3	1	0	0	0	0	862
No. of Pairs, speed 2	209	127	1	62	34	0	13	10	0	2	4	0	0	1	0	1153
No. of Pairs, speed 3	265	147	1	51	26	0	5	8	0	0	4	0	1	1	0	1172

The increase in the number of route requests being generated at the whole network level with the increase of the nodes' speeds is expected, since higher node speeds result in an increased number of *broken links* – therefore, nodes have to send new discovery packets for finding new paths. On the other hand, the increase in the number of paths of a given length is roughly proportional to the original number of paths, thus leading to roughly the same minimum entropy values.

The number of achieved random bits, along with the number of subsets in the whole network are shown in Tables 2.8 and 2.9 for $\epsilon_1 = 10^{-3}$. Not surprisingly, the total network-wide number of achieved shared secret bits (between any pairs of nodes) also increases with the node speeds.

Table 2.9 Number of node pairs vs. number of subsets for three different speeds by applying naïve algorithm with $\epsilon_1 = .001$, when RID is protected. Total network-wide number of shared secret bits, in last column.Speed 1, speed 2 and speed 3 are uniform (.5,1), uniform (1,1.5) and uniform (1.5,2), respectively.

No. of Subsets	1	2	3	4	5	6	7	B_{total}
No. of Pairs, speed 1	215	75	22	6	1	0	1	$4.98 \cdot 10^3$
No. of Pairs, speed 2	200	77	37	17	12	3	2	$6.63 \cdot 10^3$
No. of Pairs, speed 3	260	86	31	12	6	2	1	$6.65 \cdot 10^3$

Table 2.10 Size of min-entropy based on 4 different ranges in the case of clear RID.

Range	range=3	range=6	range=9	range=12	range=15
$H_{min}(r \mid K_{Eve}(r) = 0, group = 1)$	.16	.65	1.056	1.22	1.023
$H_{min}(r \mid K_{Eve}(r) = 0, group = 2)$	1.26	1.88	2.581	2.893	2.64
$H_{min}(r \mid K_{Eve}(r) = 0, group = 3)$	2.58	3.988	4.751	5.257	4.74

2.5.2.2 The effects of transmission range

In the following, we explore the effect of the wireless node range in the number of attained random bits. To perform this experiment, we simulate networks with the same parameters as those in Table 2.4, except with different wireless node ranges: 3, 6, 9 ,12 and 15 meters. The number of secret random bits per subset in the case of fully-protected RIDs, for wireless ranges 3, 6, 9, 12 and 15 meters, are 8.49, 9.45, 10.28 , 10.66 and 10.25 bits respectively. In the case of RIDs sent in the clear, the entropy for each group in above the ranges is shown in Table 2.10. The total number of secret random bits, along with the number of subsets in the whole network is shown in Tables 2.11 and 2.12 for $\epsilon_1 = 10^{-3}$ in the case of protected and clear RID respectively.

Based on simulation results, the number of full routes of any length in the whole network for these five ranges (3, 6, 9, 12 and 15 meters) are respectively 852, 2815, 8984, 14544 and 21648. When the transmission range increases, the nodes can establish communication links more easily, causing an increase in the number of full routes, and hence in the number of

Table 2.11 Number of subsets, obtained by the naïve algorithm with $\epsilon_1 = .001$, in the case of sending RID in clear for different transmission range. Total network-wide achievable number of shared secret bits, in last column.

No. of Subsets	1			2			3			4			5			
Group	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	B_{total}
No. of Pairs, range=3	27	1	0	1	0	0	0	0	0	0	0	0	0	0	0	89
No. of Pairs, range=6	121	15	1	2	0	0	0	0	0	0	0	0	0	0	0	113
No. of Pairs, range=9	234	122	1	14	6	0	2	1	0	0	1	0	0	0	0	651
No. of Pairs, range=12	203	125	3	31	16	1	4	2	0	3	1	0	0	0	0	862
No. of Pairs, range=15	213	109	0	48	13	0	10	2	0	0	2	0	0	0	0	1480

Table 2.12 Number of node pairs vs. number of subsets for five different ranges by applying naïve algorithm. Total network-wide number of shared secret bits, in last column.

No. of Subsets	1	2	3	4	5	6	7	B_{total}
No. of Pairs, range=3	29	1	0	0	0	0	0	263.19
No. of Pairs, range=6	134	6	0	0	0	0	0	$1.38 \cdot 10^3$
No. of Pairs, range=9	284	60	5	2	0	0	0	$4.39 \cdot 10^3$
No. of Pairs, range=12	215	75	22	6	1	0	1	$4.98 \cdot 10^3$
No. of Pairs, range=15	207	75	26	5	7	0	0	$5.02 \cdot 10^3$

shared secret bits. On the other hand, by increasing the transmission range, an eavesdropper can get information about routes more easily. It is therefore expected that the number of shared secret bits decreases as the transmission range keeps increasing beyond a certain point. For example when the range is 50m, the eavesdropper will overhear any route.

2.5.3 Increasing The Secret's Length by Spoiling Knowledge

Spoiling knowledge was introduced in [12] as a means of (publicly) adjusting a probability distribution to increase its min entropy. In our specific example, this translates to purposely discarding the most likely full routes from the SRT. But since all routes of the same length have the same probability (from Eve's perspective), we can only increase the min entropy by discarding all the routes of a given length. The downside, of course, is that the number of partitions satisfying the properties outlined in Section 2.5.1 also decreases.

To show the effect of spoiling knowledge, we have considered a scenario in which the speed of nodes is uniform (.5,1) and maximum eavesdropping range is 12 meters. In this part we have compared the results for two different algorithms, i.e., naïve algorithm and heuristic algorithm. For our specific scenario, disregarding the full routes of length 3 yields a min entropy of roughly $H_{min}(r|K_{Eve}(r) = 0)) = -\log_2(9.02 \cdot 10^{-6}) = 16.76$ bits. The number of subsets produced for the whole network, for $\epsilon_1 = 10^{-3}$, is shown in table 2.13, for our two different partitioning algorithms. In this case, B_{total} is $6.13 \cdot 10^3$ bits for naïve algorithm and $7.59 \cdot 10^3$ bits for heuristic algorithm. It is interesting to note that the spoiling knowledge technique achieves a gain of roughly 23% and 49%, in the naïve algorithm and in the heuristic algorithm respectively.

Table 2.13 Number of subsets, obtained by the naïve and heuristic algorithms, when considering only full routes of length at least 4. Total network-wide achievable number of shared secret bits, in last column.

No. of Subsets	1	2	3	4	5	6	7	8
No. of Pairs-naïve	195	60	11	2	2	0	0	0
No. of Pairs-heuristic	165	56	31	11	5	1	0	1

CHAPTER 3. ON THE SECRET KEY CAPACITY OF SIBLING HIDDEN MARKOV MODELS

3.1 Introduction

Establishing a secret key between two or more legitimate parties is the basic principle of cryptography and secure communication. In the absence of public-key infrastructure, several alternative key-establishment approaches have been proposed, and rely on information-theoretic methods. The merit of such approaches is that they offer information-theoretic security, and thus do not rely on any assumptions on the computational capabilities of adversaries. Key establishment based on information theoretic methods was initially studied by Maurer [5] and Ahlswede and Csiszár [6]. By considering the i.i.d repetitions of correlated random variables at the disposal of two legitimate users, called Alice (X) and Bob (Y) and one eavesdropper, named Eve (Z), it was shown that when $X - Y - Z$ form a Markov chain, the secret key capacity equals the conditional mutual information $I(X, Y|Z)$. This result holds under the additional assumption that Alice and Bob can communicate over an error-free but insecure public channel, and the eavesdropper Eve cannot manipulate the information exchanged between Alice and Bob over this channel (the *passive attacker* assumption).

The abundance of research building on [5] and [6] maintains the i.i.d character of the correlated processes available to Alice, Bob and Eve. Nevertheless, it is becoming increasingly clear that real-life scenarios have to be approached based on non-i.i.d assumptions. As a concrete example, in the recently published KERMAN protocol [31, 30], Alice and Bob establish a secret key from routing meta-data in an ad hoc network. More precisely, their source of randomness is the network's connectivity, which is highly dynamic due to the movement of mobile nodes in the ad hoc network. However, [31, 30] rely on the assumption that various observations of the network by the same participant are independent of each other – such an assumption is practical when the network observations are obtained (sampled) at intervals of time large enough to de-correlate them[1]. This is the equivalent of the *fast fading* assumption in wireless channels. By contrast, in this chapter we study the scenario in which the network evolves slowly (relative to the network-observation sampling time), and consequently the network states at two contiguous sampling times are correlated.

We model the network with a Markov chain – in which a state could describe the network connectivity, perhaps represented as an adjacency matrix – and we model the network observation mechanism by a noisy channel – at each time instant, an observer would see an incomplete or noisy version of the adjacency matrix. The specifics of the observation channel are beyond the scope of this chapter. Instead, we proceed to investigate the secret key capacity of a *Sibling Hidden Markov Model* (SHMM), where two legitimate users and one eavesdropper have access to imperfect observations of the same underlying Markov chain. The applicability of this new model is much more general than the network observation example outlined above. The model can accurately capture the behavior of any source model for common randomness, as long as the source can be approximately modeled as a Markov chain. Additional practical examples of interest include: noisy observations of wireless transmissions of correlated data (text, voice or moving images [33, 34]), imperfect observation of the evolution of a complex system (like a community in a social network, or a geographic community [35, 36] – in this case, the legitimate parties' advantage could be based on a better cultural understanding and interpretation of the observed phenomena), imperfect observations of power fluctuations at different points of a smart grid [37], etc.

[1] Experimental characterizations of the autocorrelation exhibited by the observed phenomena, like the investigation of MIMO channel gains in [32], can be used to determine the downsampling necessary to satisfy the i.i.d. assumptions. Nevertheless, such downsampling comes at the cost of secrecy rate losses.

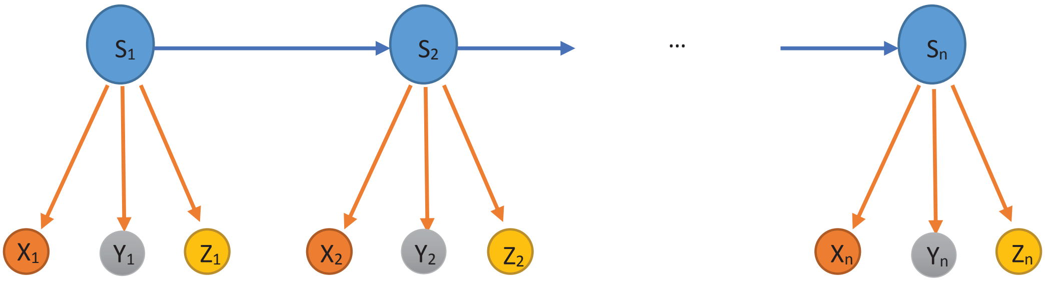

Figure 3.1 Sibling Hidden Markov Model for generating the secret key

One technical problem arises immediately. Since the techniques used for studying i.i.d cases are based on typicality arguments, they cannot be readily employed for non-i.i.d models such as the HMM. To evaluate the secret key capacity of the SHMM, our approach starts by extending known single shot results. Tyagi and Watanabe [38] propose to upper-bound the secret key capacity in terms of the probability of missed detection in a binary hypothesis test. We shall use this bound, along with the single shot lower bound studied in [39] to explore the secret key capacity of the SHMM. The main difficulty in our approach comes from the method's computational complexity. To address this issue, we transform the bounds into a log-likelihood ratio of the joint probabilities of the observed variables, and notice that the computational obstacle resides in the non-additivity of the log-likelihood ratio. At this point, following a method developed by Fuh in [40], we express each joint probability distribution as the L_1 norm of a product of random matrices. It was shown in [40] that the logarithm of this matrix product norm can be written as a cumulative functional of a Markov chain, which under certain common assumptions converges to a Lyapunov exponent (as the size of the chain grows towards infinity). Finally, we employ numerical methods to estimate each of the Lyapunov exponents, and compute our bounds.

The rest of this chapter is organized as follows. Section 3.2 provides a brief overview of the related work. In section 3.3 we describe the system model and assumptions, while in section 3.5 we present the derivation of the upper and lower bounds on the secret key capacity of the SHMM. In section 3.6, we explain how we can express the bounds in terms of the L_1 norms of products of random matrices, and finally in terms of Lyapunov exponents, and we discuss how to calculate each Lyapunov exponent based on a numerical method. Section 3.7 is devoted to simulation results, while section 3.8 contains concluding remarks. We should mention that partial results from this chapter were previously published in [41]. By comparison, the current version provides the characterization of both lower and upper bounds, and is more complete and self-contained.

3.2 Related Work

Two important techniques to study single-shot scenarios have been developed, namely smooth entropies and information spectrum methods [42]. As an example of the smooth entropy method, in [9] Renner and Wolf derived single-shot upper and lower bounds on the size of the secret key. In [43], new bounds were proposed based on the large deviation technique and the idea of smoothing, while [44] derived new bounds for the length of the secret key based on the information spectrum methods. Unfortunately extending these bounds not only to our SHMM model, but even for simpler non-i.i.d cases is not computationally tractable [38].

Hayashi and Watanabe studied the problem of secure uniform random number generation (SURNG) for Markov chains in [45]. In this problem X and Y are two correlated random variables with joint distribution P_{XY}. The aim is to build a new random variable from X which has no correlation with Y. To produce such a random variable, a two-universal hash function is applied to the random variable X. The authors have considered a Markov model with the transition matrix $W(x, y|x', y') = P(X_n = x, Y_n = y|X_{n-1} = x', Y_{n-1} = y')$. This model has been shown in Figure 3.2. The authors evaluated the Rényi entropy for the Markov

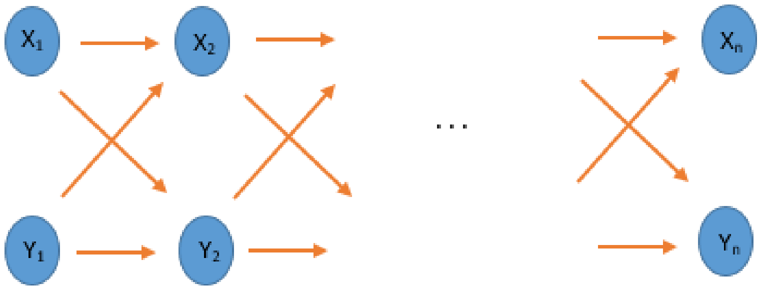

Figure 3.2 Markov Model for generating the secret key

chain in terms of a newly defined measure: the Rényi entropy for the transition matrix. To derive computable asymptotic bounds based on different discussed asymptotic regimes [45] and non-asymptotic bounds, the transition matrix must be non-hidden or strongly non-hidden. A transition matrix is non-hidden if $\sum_x W(x,y|x',y') = W_Y(y|y')$ and it is strongly non-hidden if $\sum_x W(x,y|x',y')^{1+\theta}$ is independent of x' for every $\theta \in (-1,\infty)$. Although this Markov model has its own merit and application, we believe that our proposed SHMM applies to a broader range of real-life scenarios.

3.3 System Model

In the study of non-i.i.d cases, the information-spectrum methods of [46] are powerful tools. In the remainder of this section we formalize our SHMM, and we introduce some notation and terminology from hypothesis testing [47, 48, 49] – this terminology is relevant for understanding the upper and lower bounds of [50], which are the starting point in our study. The reader familiar with these concepts can skip ahead to Section 3.5.

3.4 Generating Secret Key Problem

Assume that the common randomness at the disposal of Alice and Bob is represented by two correlated random variables, X and Y respectively. In addition, assume that an adversary Eve possesses side information (Z) correlated with the two legitimate users' information (X, Y and Z take values from $\mathcal{X}, \mathcal{Y}$ and $\mathcal{Z}$ respectively). Alice and Bob want to generate a secret key through the use of an additional public but authenticated channel, i.e, the adversary eavesdropper can overhear the public discussion, but cannot actively interfere with it. The key establishment process is constructed, as usual, of two phases: information reconciliation and privacy amplification. In the information reconciliation phase, the legitimate users exchange communication over the public channel, and end up agreeing on the same exact randomness-bearing sequence. Communication between Alice and Bob in this phase can be either one-way or interactive. Since some information about the agreed-upon randomness-bearing sequence has been revealed to Eve from both her own side information Z and from the the public communication of the former phase, the goal of privacy amplification is to increase the secrecy of the first phase output. This is usually performed by applying a randomness extractor. Eventually, Alice and Bob will have random variables K_A and K_B respectively as their *shared* secret key. The two keys have to be the same with high probability, and have to be distributed uniformly given the adversary's knowledge.

Definition 7. *Alice and Bob, by public communication represented as F, can generate an $(\epsilon, \delta)-$secure secret key [51] if*

$$P(K_A = K_B = K) \geq 1 - \epsilon$$

and

$$d(P_{KFZ}, \frac{1}{|\mathcal{K}|} P_{FZ}) \leq \delta$$

where

$$d(P,Q) = \frac{1}{2}\sum_{x} |P(x) - Q(x)|.$$

In Definition 7, d is the *statistical distance* (also known as the *total variation distance*), and quantifies the distance between the real and the ideal probability distributions. Note that the probability distribution of the the secret key is ideally uniform and independent of Eve's information. It is known [52] that the statistical distance defined above can be equivalently expressed as $d(P,Q) = max_{X_0 \subset \mathcal{X}}\{|P(X_0) - Q(X_0)|\}$.

A new definition of the secret key, called "ϵ-security", was proposed in [50] by combining the conditions in Definition 7:

Definition 8. *ϵ-secure secret key can be generated between Alice and Bob if*

$$d(P_{K_A K_B F Z}, P_{unif} P_{FZ}) \leq \epsilon,$$

where

$$P_{unif}(K_A, K_B) = \frac{\mathbb{1}(K_A = K_B)}{|\mathcal{K}|}.$$

3.4.1 Sibling Hidden Markov Models

As discussed in section 3.1, we study a model in which Alice (X_n), Bob (Y_n) and Eve (Z_n) have access to imperfect observations of a source represented by a Markov chain (S_n). This model has been illustrated in Fig 3.1. Although our approach can theoretically be applied to random variables taking values from finite sets, due to computational issues, our study has been restricted to all-binary random variables, i.e, X_n, Y_n, Z_n and $S_n \in \{0,1\}$. Moreover, we have assumed that the underlying Markov chain is irreducible and aperiodic, and that its initial distribution is the stationary distribution, π. The transition matrix for the underlying

Markov chain is represented by T_S

$$T_S = \begin{bmatrix} 1-\alpha_S & \alpha_S \\ \beta_S & 1-\beta_S \end{bmatrix}.$$

Corresponding emission matrices for Alice, Bob and Eve are denoted respectively by

$$E_A = \begin{bmatrix} 1-\alpha_A & \alpha_A \\ \beta_A & 1-\beta_A \end{bmatrix}, \quad E_B = \begin{bmatrix} 1-\alpha_B & \alpha_B \\ \beta_B & 1-\beta_B \end{bmatrix}$$

and

$$E_Z = \begin{bmatrix} 1-\alpha_Z & \alpha_Z \\ \beta_Z & 1-\beta_Z \end{bmatrix}.$$

3.5 The Secret Key Establishment Potential of the Sibling Hidden Markov Model

3.5.1 Upper Bound

It is only feasible for Alice and Bob to generate secret bits when, conditioned on Eve's side information, they have additional shared information. Based on this idea, [50] derived a single-shot upper bound for the length of the ϵ-secure secret key length ($S_\epsilon(X,Y|Z)$) in terms of β (defined in 1.3.3), as a distance between the joint probability distribution P_{XYZ} and $P_{X|Z}P_{Y|Z}P_Z$. Namely, for every $0 \leq \epsilon < 1$ and $0 < \eta < 1-\epsilon$, we have

$$\begin{aligned} S_\epsilon(X,Y|Z) \leq -\log\beta_{\epsilon+\eta}(P_{XYZ}, P_{X|Z}P_{Y|Z}P_Z) \\ +2\log(\frac{1}{\eta}). \end{aligned} \tag{3.1}$$

Unfortunately, there is currently no way to efficiently calculate β directly for our SHMM. However, by defining the *spectrum divergence* [53] between two distributions as

$$\begin{aligned} & D_s^{\epsilon}(P_{\theta_0}||P_{\theta_1}) \\ &= \sup\{R : P_{\theta_0}\{x \in \mathcal{X} : \log \frac{P_{\theta_0}(x)}{P_{\theta_1(x)}} \leq R\} \leq \epsilon\}, \end{aligned} \tag{3.2}$$

and by applying the Neyman Pearson lemma, we have the following lemma [44].

Lemma 3. *For every $\epsilon \in \{0, 1\}$ and $0 < \eta_1 < 1 - \epsilon$, the following relation between β and the spectrum divergence holds:*

$$-D_s^{\epsilon+\eta_1}(P_{\theta_0}\|P_{\theta_1}) - \log \frac{1}{\eta_1} \leq \log \beta_\epsilon \leq -D_s^{\epsilon}(P_{\theta_0}\|P_{\theta_1}). \tag{3.3}$$

Consequently, the following upper bound for the achievable ϵ-secure secret key length can be expressed by combining (3.1) and (3.3):

$$\begin{aligned} S_\epsilon(X, Y|Z) \leq D_s^{\epsilon+\eta_1+\eta}&(P_{XYZ}||P_{X|Z}P_{Y|Z}P_Z) \\ &+ \log(\frac{1}{\eta^2\eta_1}). \end{aligned} \tag{3.4}$$

The advantage of (3.4) over (3.1) resides in computational costs which will be discussed in the upcoming sections. By extending the new single-shot upper bound for a sequence of length n, for every $0 \leq \epsilon < 1$ and $0 < \eta, \eta_1 < 1 - \epsilon$, the maximum length of an ϵ-secure secret key can be written as

$$\begin{aligned} & S_\epsilon(X^n, Y^n|Z^n) \\ &\leq D_s^{\epsilon+\eta_1+\eta}(P_{X^nY^nZ^n}||P_{X^n|Z^n}P_{Y^n|Z^n}P_{Z^n}) \\ &\qquad + \log(\frac{1}{\eta^2\eta_1}). \end{aligned} \tag{3.5}$$

3.5.2 Lower Bound

Recently [39] proposed a protocol to generate a single-shot secret key based on interactive communication. The results of [39] can be considered as an achievable lower bound for the length of the secret key. Recall that the protocol has two steps: information reconciliation and privacy amplification. As discussed in section 1.3.1, intuition suggests that $H_{max}(P_{XY}|P_Y)$ bits have to be communicated between Alice and Bob in the information reconciliation phase for *every realization* of (X, Y). This amount can be proved by the single shot Slepian-Wolf theorem [54]. Nevertheless, [39] proposes an interactive communication protocol to decrease the number of exchanged bits for specific realizations of (X, Y). To this end, the spectrum of $P_{X|Y}$ is divided into L slices, each of width Δ. This technique is called *information slicing* and was introduced in [46]. Instead of binning X with the bin size of $H_{max}(P_{XY}|P_Y)$ bits, the bin size starts from almost $H_{min}(P_{XY}|P_Y)$ and is increased iteratively, each time by as much as Δ bits. As a consequence of this method, it can be observed that when $H_{min}(P_{XY}|P_Y)$ coincides with $H_{max}(P_{XY}|P_Y)$ then it is sufficient that Alice and Bob communicate with each other just one time – no need for spectrum slicing. This observation makes things a bit easier for us – namely, in the asymptotic regime, where the probability distributions concentrate at their averages, and H_{min} becomes close to H_{max}, we can actually achieve the lower bound through a very simple protocol, in which the entire reconciliation information is transmitted in one step. By contrast, in the non-asymptotic case, even if we could calculate the lower bound, the protocol to achieve it would not be feasible, as the slicing technique involves calculating the (smooth) minimum and maximum conditional entropies of X given Y – in turn, this relies on the calculation of the entire spectrum of $P_{X|Y}$.

In the general case, the number of bits that need to be exchanged between Alice and Bob in this reconciliation protocol is roughly $\log \frac{1}{P_{x|y}}$ for the specific realization (x, y) of (X, Y). Moreover, for privacy amplification the authors of [39] have used two-universal hash functions as strong seeded extractors. Altogether, based on the proposed protocol, we can derive the lower bound of the length of the secret key, i.e, for every $\eta_3 > 0$ and $\lambda \geq 0$, there exists an (ϵ, δ)-secure secret key taking value from a set $\mathcal{S} \in \{0,1\}^u$ with

$$\epsilon \leq P_{XY}(\mathcal{T}_0) + L2^{-\eta_3} \tag{3.6}$$

and

$$\begin{aligned} \delta \leq P(i_{XY}(X,Y) - i_{XZ}(X,Z) \leq \lambda + \Delta) + L2^{-\eta_3} \\ + \frac{1}{2}\sqrt{2^{u-\lambda+\eta_3+3\log L}} + \frac{1}{L} + P_{XY}(\mathcal{T}_0), \end{aligned} \tag{3.7}$$

where

$$i_{XY}(x,y) := \log \frac{P_{XY}(x,y)}{P_X(x)P_Y(y)}$$

and

$$\begin{aligned} \mathcal{T}_0 = \{(x,y) | \log \frac{1}{P_{X|Y}(x,y)} < H_{min}(P_{XY}|P_Y) \text{ or} \\ \log \frac{1}{P_{X|Y}(x,y)} > H_{max}(P_{XY}|P_Y)\}. \end{aligned}$$

The *secret key capacity* C when X, Y and Z are n-symbol sequences, can be defined as $1/n$ times the length of the secret key when the security parameters such as ϵ and δ go to zero.

Lemma 4. *The upper bound and lower bound of the secret key capacity for a sequence is equal to the left most point in the spectrum of* $\frac{1}{n}\log\frac{P_{X^nY^nZ^n}P_{Z^n}}{P_{X^nZ^n}P_{Y^nZ^n}}$ *and* $\frac{1}{n}\log(\frac{P_{X^nY^n}}{P_{X^nZ^n}}\frac{P_{Z^n}}{P_{Y^n}})$ *respectively, when* (X^n, Y^n, Z^n) *is distributed by* $P_{X^nY^nZ^n}$.

Proof. The upper bound on the secret key capacity is obtained directly from (3.5), and the lower bound by appropriately choosing the parameters of (3.6) as

$$\begin{aligned} L = L_n = \frac{H_{max}(P_{X^nY^n}|P_{Y^n}) - H_{min}(P_{X^nY^n}|P_{Y^n})}{\Delta} \\ \eta_3 = \eta_{3_n} = \frac{n\Delta}{2} \\ \lambda = \lambda_n = n(P - \liminf_{n\to\infty} \frac{1}{n}(i_{X^nY^n}(X^n, Y^n) \\ -i_{X^nZ^n}(X^n, Z^n))) - \Delta \\ = n(P - \liminf_{n\to\infty}(\frac{1}{n}\log \frac{P_{X^nY^n}}{P_{X^nZ^n}}\frac{P_{Z^n}}{P_{Y^n}})) - \Delta. \end{aligned}$$

The third term after the inequality in (3.8) can be written as

$$\frac{1}{2}\sqrt{2^{n(\frac{u}{n} - \frac{\lambda_n}{n} + \frac{\eta 3_n}{n} + \frac{3\log L}{n})}}. \tag{3.8}$$

So, when $n \to \infty$, (ϵ and δ) $\to 0$, and the lower bound for secret key capacity is $P - \liminf_{n\to\infty}(\frac{1}{n}\log \frac{P_{X^nY^n}}{P_{X^nZ^n}}\frac{P_{Z^n}}{P_{Y^n}})$. □

Our aim is now to evaluate the terms in Lemma 4 in the asymptotic case.

3.6 Calculating the Bounds

According to Lemma 4, we need to compute the probability distributions of the specific log-likelihood ratios $\frac{1}{n}\log\frac{P_{X^nY^nZ^n}P_{Z^n}}{P_{X^nZ^n}P_{Y^nZ^n}}$ and $\frac{1}{n}\log(\frac{P_{X^nY^n}}{P_{X^nZ^n}}\frac{P_{Z^n}}{P_{Y^n}})$. The main problem in this step is related to the non-additivity of the logarithm of the joint probability distributions of the observations. To make it clearer, consider one joint distribution of the observations such as $P_{X^nY^n}$. This distribution can be written as

$$P_{X^nY^n} = \sum_{S^n} P_{S^n}P_{X^n|S^n}P_{Y^n|S^n} \tag{3.9}$$

$$= \sum_{S^n} P_{S_1}P_{X_1|S_1}P_{Y_1|S_1}\prod_{i=2}^{n} P_{S_{i+1}|S_i}P_{X_i|S_i}P_{Y_i|S_i}$$

where $S_i \in \{0,1\}$ and $i \in \{1,2,...,n\}$. Similar expressions, involving intractable summations, can be written for the other joint probability distributions. To deal with such intractable summations, [40] employs a method based on the L_1 norm of product of random matrices. Namely, and as an example, $P_{X^nY^n}$ can be written as follows:

$$P_{X^nY^n} = \|M_{X_n,Y_n} \;...\; M_{X_2,Y_2}M_{X_1,Y_1}\pi\| = \|T_{n_{XY}}\pi\| \tag{3.10}$$

where:

$$\pi = \begin{bmatrix} \frac{\beta_S}{\alpha_S+\beta_S} \\ \frac{\alpha_S}{\alpha_S+\beta_S} \end{bmatrix},$$

$$M_{X_1,Y_1} = \begin{bmatrix} f_{10}(X_1Y_1) & 0 \\ 0 & f_{11}(X_1Y_1) \end{bmatrix},$$

$$M_{X_n,Y_n} = \begin{bmatrix} (1-\alpha_S)f_{n0}(X_nY_n) & \beta_S f_{n0}(X_nY_n) \\ \alpha_S f_{n1}(X_nY_n) & (1-\beta_S)f_{n1}(X_nY_n) \end{bmatrix},$$

$$T_{n_{XY}} = M_{X_n,Y_n}\ldots M_{X_2,Y_2}M_{X_1,Y_1}, \tag{3.11}$$

and

$$f_{n0} = P(X_n|S_n = 0)P(Y_n|S_n = 0), \tag{3.12}$$

$$f_{n1} = P(X_n|S_n = 1)P(Y_n|S_n = 1). \tag{3.13}$$

We can write the other joint distributions in Lemma 4 in a similar way, with the only difference appearing in (3.12), (3.13), where, for $P_{X^nY^nZ^n}$,

$$f_{n0} = P(X_n|S_n = 0)P(Y_n|S_n = 0)P(Z_n|S_n = 0)$$

$$f_{n1} = P(X_n|S_n = 1)P(Y_n|S_n = 1)P(Z_n|S_n = 1),$$

for $P_{X^nZ^n}$,

$$\begin{aligned} f_{n0} &= P(X_n|S_n=0)P(Z_n|S_n=0) \\ f_{n1} &= P(X_n|S_n=1)P(Z_n|S_n=1), \end{aligned}$$

for P_{Z^n},

$$\begin{aligned} f_{n0} &= P(Z_n|S_n=0) \\ f_{n1} &= P(Z_n|S_n=1), \end{aligned}$$

and for P_{Y^n},

$$\begin{aligned} f_{n0} &= P(Y_n|S_n=0) \\ f_{n1} &= P(Y_n|S_n=1). \end{aligned}$$

In these equations, f_{n0} and f_{n1} can be denoted by f_0 and f_1 respectively, since the emission matrices do not depend on n.

Now the log-likelihood ratio in the upper bound of lemma (4) can be expressed as

$$\begin{aligned} &\frac{1}{n}\log\frac{P_{X^nY^nZ^n}P_{Z^n}}{P_{X^nZ^n}P_{Y^nZ^n}} \\ =\ &\frac{1}{n}\log(\frac{\|M_{X_n,Y_n,Z_n}\ \dots\ M_{X_2,Y_2,Z_2}M_{X_1,Y_1,Z_1}\pi\|}{\|M_{X_n,Z_n}\ \dots\ M_{X_2,Z_2}M_{X_1,Z_1}\pi\|} \\ =\ &\frac{\|M_{Z_n}\ \dots\ M_{Z_2}M_{Z_1}\pi\|}{\|M_{Y_n,Z_n}\ \dots\ M_{Y_2,Z_2}M_{Y_1,Z_1}\pi\|}) \\ =\ &\frac{1}{n}\log\frac{\|T_{n_{XYZ}}\pi\|\|T_{n_Z}\pi\|}{\|T_{n_{XZ}}\pi\|\|T_{n_{YZ}}\pi\|}, \end{aligned} \tag{3.14}$$

where $T_{n_Y}, T_{n_Z}, T_{n_{XZ}}$ are defined analogously to (3.11). Similarly, the log-likelihood ratio in the lower bound of lemma (4) can be written as

$$\begin{aligned} &\frac{1}{n}\log\frac{P_{X^nY^n}P_{Z^n}}{P_{X^nZ^n}P_{Y^n}} \\ = &\frac{1}{n}\log(\frac{\|M_{X_n,Y_n}\ \dots\ M_{X_2,Y_2}M_{X_1,Y_1}\pi\|}{\|M_{X_n,Z_n}\ \dots\ M_{X_2,Z_2}M_{X_1,Z_1}\pi\|} \\ &\cdot\frac{\|M_{Z_n}\ \dots\ M_{Z_2}M_{Z_1}\pi\|}{\|M_{Y_n}\ \dots\ M_{Y_2}M_{Y_1}\pi\|}) \\ = &\frac{1}{n}\log\frac{\|T_{n_{XY}}\pi\|\|T_{n_Z}\pi\|}{\|T_{n_{XZ}}\pi\|\|T_{n_Y}\pi\|}. \end{aligned} \tag{3.15}$$

For the sake of convenience and since the upcoming discussion is the same for all T_ns related to the different joint probability distributions, we denote all of them with T_n. In the general case, i.e, when all observations related to X, Y, Z are studied, the system $\{(S_n, X_n, Y_n, Z_n, T_n)\}$ is called a product of Markov random matrices.

The product of i.i.d. invertible random matrices was initially studied by [55]. It was shown that the logarithm of the L_1 norm of this product can be written as a simple functional of a Markov chain, which asymptotically, and taking into account irreducibility assumption, equals to the upper Lyapunov exponent. In [40] and [56], Fuh develops a methodology for calculating the asymptotic limits for products of Markov random matrices. The problems tackled therein refer to the efficiency of certain estimators [40] and change-point detection algorithms [56] in HMMs. The rest of this chapter is based on this methodology, and uses notation similar to [56].

Now $\frac{1}{n}\log\|T_n\pi\|$ can be expressed as

$$\log\frac{\|T_n\pi\|}{\|T_{n-1}\pi\|} + \log\frac{\|T_{n-1}\pi\|}{\|T_{n-2}\pi\|} + \ \dots\ + \log\frac{\|T_1\pi\|}{\|\pi\|}. \tag{3.16}$$

By defining $K_n = \frac{\|T_n\pi\|}{\|T_{n-1}\pi\|}$, and based on the dependency graph of Figure 3.3, $(S_n, X_n, Y_n, Z_n, K_n)$ or (S_n, K_n) form a Markov chain. It will be proved in the sequel that this Markov chain has an invariant probability measure. At a first glance, it would seem that K_n is two-dimensional,

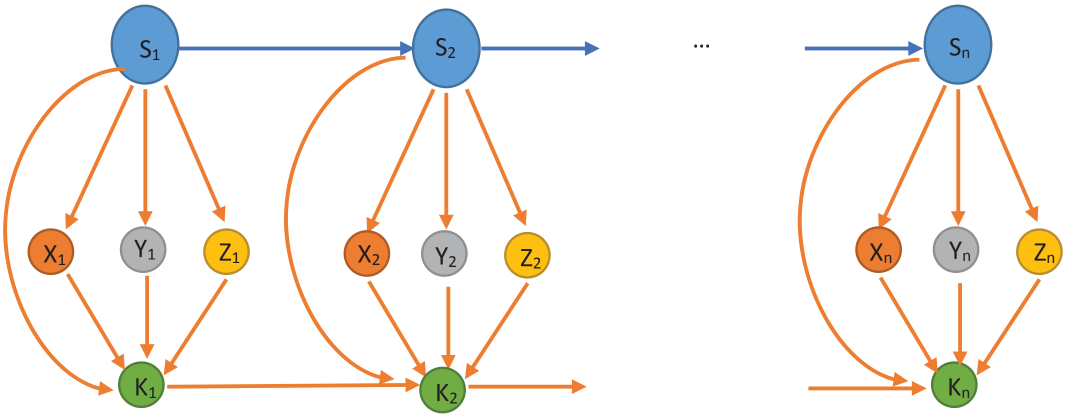

Figure 3.3 Dependency Graph of the Product of Random Matrix

but for our scenario it depends only on a one-dimensional variable. Define A_n^* and B_n^* as follows:

$$\frac{T_n\pi}{\|T_{n-1}\pi\|} = \begin{bmatrix} A_n^* \\ B_n^* \end{bmatrix}. \tag{3.17}$$

The left hand side of the above equation can be written as

$$\frac{T_n\pi}{\|T_{n-1}\pi\|} = \frac{M_n T_{n-1}\pi}{\|T_{n-2}\pi\|}\frac{\|T_{n-2}\pi\|}{\|T_{n-1}\pi\|}, \tag{3.18}$$

and thus K_n we can be expressed as:

$$\begin{aligned} \log\frac{\|T_n\pi\|}{\|T_{n-1}\pi\|} &= \log\|\frac{M_n T_{n-1}\pi}{\|T_{n-2}\pi\|}\frac{\|T_{n-2}\pi\|}{\|T_{n-1}\pi\|}\| \\ &= \log\|\begin{bmatrix} \frac{(1-\alpha_S)A_{n-1}^*+\beta_S B_{n-1}^*}{A_{n-1}^*+B_{n-1}^*} f_0 \\ \frac{\alpha_S A_{n-1}^*+(1-\beta_S)B_{n-1}^*}{A_{n-1}^*+B_{n-1}^*} f_1 \end{bmatrix}\| \end{aligned} \tag{3.19}$$

According to (3.19), we have following recursive equations for A_n^* and B_n^*:

$$\begin{aligned} A_n^* &= \frac{(1-\alpha_S)A_{n-1}^* + \beta_S B_{n-1}^*}{A_{n-1}^* + B_{n-1}^*} f_0, \\ B_n^* &= \frac{\alpha_S A_{n-1}^* + (1-\beta_S)B_{n-1}^*}{A_{n-1}^* + B_{n-1}^*} f_1, \\ A_0^* &= \frac{\beta_S}{\alpha_S+\beta_S} \text{ and } B_0^* = \frac{\alpha_S}{\alpha_S+\beta_S}, \end{aligned} \tag{3.20}$$

and thus

$$\frac{1}{n}\log\|T_n\pi\| = \frac{1}{n}\sum_1^n \log(A_n^* + B_n^*). \tag{3.21}$$

Since the Markov chain $(S_n, X_n, Y_n, Z_n, K_n)$ has an invariant probability measure, by using the Ergodic theorem of [57], as $n \to \infty$, (3.21) will converge to Lyapunov exponent γ (a brief overview of the Lyapunov exponent, including a definition and historical applications to similar contexts, is presented in the Appendix):

$$\lim_{n\to\infty}\frac{1}{n}\log\|T_n\pi\| = \gamma = E[\log(A_1^* + B_1^*)]. \tag{3.22}$$

As a result, we have the following theorem.

Theorem 3. *The upper and lower bounds for the secret key capacity based on the binary SHMM represented in Figure 3.1 when the underlying Markov chain is aperiodic and irreducible can be expressed respectively as the following*

$$\lim_{n\to\infty}\frac{1}{n}\log\frac{P_{X^nY^nZ^n}P_{Z^n}}{P_{X^nZ^n}P_{Y^nZ^n}} = \gamma_{xyz} + \gamma_z - \gamma_{xz} - \gamma_{yz}, \tag{3.23}$$

$$\lim_{n\to\infty}\frac{1}{n}\log\frac{P_{X^nY^n}P_{Z^n}}{P_{X^nZ^n}P_{Y^n}} = \gamma_{xy} + \gamma_z - \gamma_{xz} - \gamma_y, \tag{3.24}$$

where $\gamma_{xyz}, \gamma_{xy}, \gamma_{xz}, \gamma_{yz}, \gamma_y$ and γ_z are Lyapunov exponents and according to (3.22) they are related to the $T_{n_{XYZ}}, T_{n_{XY}}, T_{n_{XZ}}, T_{n_{YZ}}, T_{n_Y}$ and T_{n_Z} respectively.

Note that the left-most point coincides with the right most point in the related spectra of Lemma 4. So, the gap between the upper bound and the lower bound for the secret key capacity of the SHMM can be written as

$$\gamma_{xyz} - \gamma_{yz} - \gamma_{xy} + \gamma_y, \tag{3.25}$$

and when there is no Z (or equivalently Z is a constant), the upper and lower bounds become equal to each other.

To calculate these Lyapunov exponents, we have to compute the related invariant probability measure for each case. Based on (3.19), it is clear that K_n depends only on a one-dimensional variable, $W_n = \frac{A_n^*}{A_n^*+B_n^*}$. The invariant probability measure of the Markov chain is the same as the stationary distribution of W_n and according to (3.19), it depends on whether the underlying Markov chain, S_n, is 0 or 1. Consequently, we consider two stationary densities of W_n, i.e, m_0 (when $S_n = 0$), and m_1 (when $S_n = 1$). We can define the joint stationary density of (S_n, W_n) as

$$P(S_n = i, W_n \leq x) = \int_0^x m_i(w)dw. \tag{3.26}$$

For each $T_{n_{XYZ}}, T_{n_{XY}}, T_{n_{XZ}}, T_{n_{YZ}}, T_{n_Y}$ and T_{n_Z} there are corresponding m_0 and m_1. It has been shown in [56] that

$$\begin{aligned} m_0(x) = (1-\alpha_S)&\int_0^1 \frac{\partial}{\partial x} Q_0(z(w,x))m_0(w)dw \\ +\beta_S&\int_0^1 \frac{\partial}{\partial x} Q_0(z(w,x))m_1(w)dw, \\ m_1(x) = \alpha_S&\int_0^1 \frac{\partial}{\partial x} Q_1(z(w,x))m_0(w)dw \\ +(1-\beta_S)&\int_0^1 \frac{\partial}{\partial x} Q_1(z(w,x))m_1(w)dw \end{aligned} \tag{3.27}$$

where

$$z(w,x) = \frac{x}{1-x}\frac{\alpha_S w + (1-\beta_S)(1-w)}{(1-\alpha_S)w + \beta_S(1-w)},$$

and Q_i is as follows:

for $T_{n_{XYZ}}$:

$$Q_i(z) = P(\frac{f_0(X_nY_nZ_n)}{f_1(X_nY_nZ_n)} \leq z|S_n = i) \tag{3.28}$$

for $T_{n_{XY}}$:

$$Q_i(z) = P(\frac{f_0(X_nY_n)}{f_1(X_nY_n)} \leq z|S_n = i) \tag{3.29}$$

for $T_{n_{XZ}}$:

$$Q_i(z) = P(\frac{f_0(X_nZ_n)}{f_1(X_nZ_n)} \leq z|S_n = i) \tag{3.30}$$

for $T_{n_{YZ}}$:

$$Q_i(z) = P(\frac{f_0(Y_nZ_n)}{f_1(Y_nZ_n)} \leq z|S_n = i) \tag{3.31}$$

and for T_{n_Z}:

$$Q_i(z) = P(\frac{f_0(Z_n)}{f_1(Z_n)} \leq z|S_n = i). \tag{3.32}$$

Equation (3.28) is the Fredholm integral equation of the second kind and m_0 and m_1 can be calculated for each T_n from this equation. We will discuss the numerical method to solve Fredholm itegral in the next section.

Finally, to compute the Lyapunov exponents we can use the iterated expectation formula, $E(X) = E(E(X|Y))$ [56]:

$$\begin{aligned}
\gamma &= E(E(\log(A_n^* + B_n^*)|S_{n-1}, W_{n-1})) \\
&= \sum_{i=0,1} \int_0^1 \sum_{j=0,1} P(S_n = j|S_{n-1} = i) \\
\cdot E(\log(A_n^* + B_n^*)&|S_n = j, S_{n-1} = i, W_{n-1} = u) m_i(u) du \\
&= \int_0^1 [(1-\alpha_S)G_0(u) + \alpha_S G_1(u)] m_0(u) du \\
&+ \int_0^1 [\beta_S G_0(u) + (1-\beta_S) G_1(u)] m_1(u) du,
\end{aligned} \tag{3.33}$$

where

$$G_j(u) = E(\log(A_n^* + B_n^*)|S_n = j, S_{n-1} = i, W_{n-1} = u). \tag{3.34}$$

Using A_n^* and B_n^* from (3.19), we have

$$\begin{aligned} G_j(u) = \sum_{i_1} & \log([(1-\alpha_S)u + \beta_S(1-u)]f_0(i_1) \\ & +[\alpha_S + (1-\beta_S)(1-u)f_1(i_1)])f_j(i_1). \end{aligned} \tag{3.35}$$

Note that for each T_n we have to calculate corresponding G_j, for example to calculate G_j for $T_{n_{XYZ}}$, i_1 in (3.35) runs over all the combinations of XYZ.

3.7 Simulation Results

To solve for m, from the Fredholm integral equation (3.28) we have to approximate two quantities: first $\int_0^1 \frac{\partial}{\partial x} Q(z(w,x))m(w)dw$ and then $\frac{\partial}{\partial x} Q(z(w,x))$. We discretize both x and w into N points each, such that $0 = x_0 < x_1, \ldots, < x_N = 1$ and $0 = w_0 < w_1, \ldots, < w_N = 1$. Now the first quantity can be approximated as $[\frac{1}{N}\frac{\partial}{\partial x}Q(z(w_0,x))]m(w_0) + \sum_{j=1}^{N-1} \frac{1}{2N}[\frac{\partial}{\partial x}Q(z(w_j,x)) + \frac{\partial}{\partial x}Q(z(w_{j+1},x))]m(w_j)$ [56], while the second quantity can be approximated as $\frac{Q(z(w,x+\Delta))-Q(z(w,x))}{\Delta}$, where $\Delta = 1/N$.

Hence, (3.28) can be rewritten in the following matrix form:

$$\left[\begin{array}{c} m_0(0) \\ m_0(\frac{1}{N}) \\ \vdots \\ m_0(\frac{N-1}{N}) \\ \hdashline m_1(0) \\ m_1(\frac{1}{N}) \\ \vdots \\ m_1(\frac{N-1}{N}) \end{array}\right] = \left[\begin{array}{c|c} A_{N\times N} & B_{N\times N} \\ \hline C_{N\times N} & D_{N\times N} \end{array}\right] \left[\begin{array}{c} m_0(0) \\ m_0(\frac{1}{N}) \\ \vdots \\ m_0(\frac{N-1}{N}) \\ \hdashline m_1(0) \\ m_1(\frac{1}{N}) \\ \vdots \\ m_1(\frac{N-1}{N}) \end{array}\right].$$

This equation can be expressed as as $m = Mm$. As depicted, the matrix M is constructed of four sub-matrices, that is, A, B, C and D. Each of these four sub-matrices can be written as N column vectors, that is,

$$A_{N\times N} = \left[\begin{array}{cccc} a_1 & a_2 & ... & a_N \end{array}\right],$$

$$B_{N\times N} = \left[\begin{array}{cccc} b_1 & b_2 & ... & b_N \end{array}\right],$$

$$C_{N\times N} = \left[\begin{array}{cccc} c_1 & c_2 & ... & c_N \end{array}\right],$$

$$D_{N\times N} = \left[\begin{array}{cccc} d_1 & d_2 & ... & d_N \end{array}\right],$$

where

$$a_1 = \left[\begin{array}{c} \frac{1-\alpha}{N}\frac{\partial}{\partial x}Q_0(z(0,0)) \\ \frac{1-\alpha}{N}\frac{\partial}{\partial x}Q_0(z(0,\frac{1}{N})) \\ \vdots \\ \frac{1-\alpha}{N}\frac{\partial}{\partial x}Q_0(z(0,\frac{N-1}{N})) \end{array}\right],$$

$$\underset{i\neq 1}{a_i} = \begin{bmatrix} \frac{1-\alpha}{2N}\{\frac{\partial}{\partial x}Q_0(z(\frac{i-1}{N},0)) + \frac{\partial}{\partial x}Q_0(z(\frac{i}{N},0))\} \\ \frac{1-\alpha}{2N}\{\frac{\partial}{\partial x}Q_0(z(\frac{i-1}{N},\frac{1}{N})) + \frac{\partial}{\partial x}Q_0(z(\frac{i}{N},\frac{1}{N}))\} \\ \vdots \\ \frac{1-\alpha}{2N}\{\frac{\partial}{\partial x}Q_0(z(\frac{i-1}{N},\frac{N-1}{N})) + \frac{\partial}{\partial x}Q_0(z(\frac{i}{N},\frac{N-1}{N}))\} \end{bmatrix},$$

$$b_1 = \begin{bmatrix} \frac{\beta}{N}\frac{\partial}{\partial x}Q_0(z(0,0)) \\ \frac{\beta}{N}\frac{\partial}{\partial x}Q_0(z(0,\frac{1}{N})) \\ \vdots \\ \frac{\beta}{N}\frac{\partial}{\partial x}Q_0(z(0,\frac{N-1}{N})) \end{bmatrix},$$

$$\underset{i\neq 1}{b_i} = \begin{bmatrix} \frac{\beta}{2N}\{\frac{\partial}{\partial x}Q_0(z(\frac{i-1}{N},0)) + \frac{\partial}{\partial x}Q_0(z(\frac{i}{N},0))\} \\ \frac{\beta}{2N}\{\frac{\partial}{\partial x}Q_0(z(\frac{i-1}{N},\frac{1}{N})) + \frac{\partial}{\partial x}Q_0(z(\frac{i}{N},\frac{1}{N}))\} \\ \vdots \\ \frac{\beta}{2N}\{\frac{\partial}{\partial x}Q_0(z(\frac{i-1}{N},\frac{N-1}{N})) + \frac{\partial}{\partial x}Q_0(z(\frac{i}{N},\frac{N-1}{N}))\} \end{bmatrix},$$

$$c_1 = \begin{bmatrix} \frac{\alpha}{N}\frac{\partial}{\partial x}Q_1(z(0,0)) \\ \frac{\alpha}{N}\frac{\partial}{\partial x}Q_1(z(0,\frac{1}{N})) \\ \vdots \\ \frac{\alpha}{N}\frac{\partial}{\partial x}Q_1(z(0,\frac{N-1}{N})) \end{bmatrix},$$

$$\underset{i\neq 1}{c_i} = \begin{bmatrix} \frac{\alpha}{2N}\{\frac{\partial}{\partial x}Q_1(z(\frac{i-1}{N},0)) + \frac{\partial}{\partial x}Q_1(z(\frac{i}{N},0))\} \\ \frac{\alpha}{2N}\{\frac{\partial}{\partial x}Q_1(z(\frac{i-1}{N},\frac{1}{N})) + \frac{\partial}{\partial x}Q_1(z(\frac{i}{N},\frac{1}{N}))\} \\ \vdots \\ \frac{\alpha}{2N}\{\frac{\partial}{\partial x}Q_1(z(\frac{i-1}{N},\frac{N-1}{N})) + \frac{\partial}{\partial x}Q_1(z(\frac{i}{N},\frac{N-1}{N}))\} \end{bmatrix},$$

$$d_1 = \begin{bmatrix} \frac{1-\beta}{N}\frac{\partial}{\partial x}Q_1(z(0,0)) \\ \frac{1-\beta}{N}\frac{\partial}{\partial x}Q_1(z(0,\frac{1}{N})) \\ \vdots \\ \frac{1-\beta}{N}\frac{\partial}{\partial x}Q_1(z(0,\frac{N-1}{N})) \end{bmatrix},$$

$$\underset{i\neq 1}{d_i} = \begin{bmatrix} \frac{1-\beta}{2N}\{\frac{\partial}{\partial x}Q_1(z(\frac{i-1}{N},0)) + \frac{\partial}{\partial x}Q_1(z(\frac{i}{N},0))\} \\ \frac{1-\beta}{2N}\{\frac{\partial}{\partial x}Q_1(z(\frac{i-1}{N},\frac{1}{N})) + \frac{\partial}{\partial x}Q_1(z(\frac{i}{N},\frac{1}{N}))\} \\ \vdots \\ \frac{1-\beta}{2N}\{\frac{\partial}{\partial x}Q_1(z(\frac{i-1}{N},\frac{N-1}{N})) + \frac{\partial}{\partial x}Q_1(z(\frac{i}{N},\frac{N-1}{N}))\} \end{bmatrix}.$$

The sum of each column in matrix A, B, C and D are $1-\alpha$, β, α and $1-\beta$ respectively. To see this, note that $\sum_{i=1}^{N-1}\frac{d}{dx}Q(z(w,x_i)) = N$. This comes from the fact that $\frac{d}{dx}Q(z(w,x_i)) = \frac{Q(z(w,x_i+\Delta))-Q(z(w,x_i))}{\Delta}$, $1/\Delta = N$, and $\sum_{i=1}^{N-1}[Q(z(w,x_i+\Delta)) - Q(z(w,x_i)] = 1$ because Q is a cummulative distribution function, so $Q(z(w,1)) = 1$ and $Q(z(w,0)) = 0$. So, the sum of each column in matrix M is one. This guarantees that the matrix M has an eigenvalue equal to 1, which means that m is the eigenvector of matrix M corresponding to eigenvalue 1. Hence, m is invariant.

We have simulated this method for several cases. In all of the scenarios, $\alpha_S = \beta_S$ unless otherwise stated and we vary α_S from .05 to .5 by increasing .05 in each step.

To study the effect of Eve's channel on the secret key capacity, we considered symmetric channels for Alice, Bob and Eve. We fixed Alice and Bob's crossover probability .2 and .3 respectively and defined corresponding emission matrices as follows

$$E_A = \begin{bmatrix} .8 & .2 \\ .2 & .8 \end{bmatrix} \quad E_B = \begin{bmatrix} .7 & .3 \\ .3 & .7 \end{bmatrix}.$$

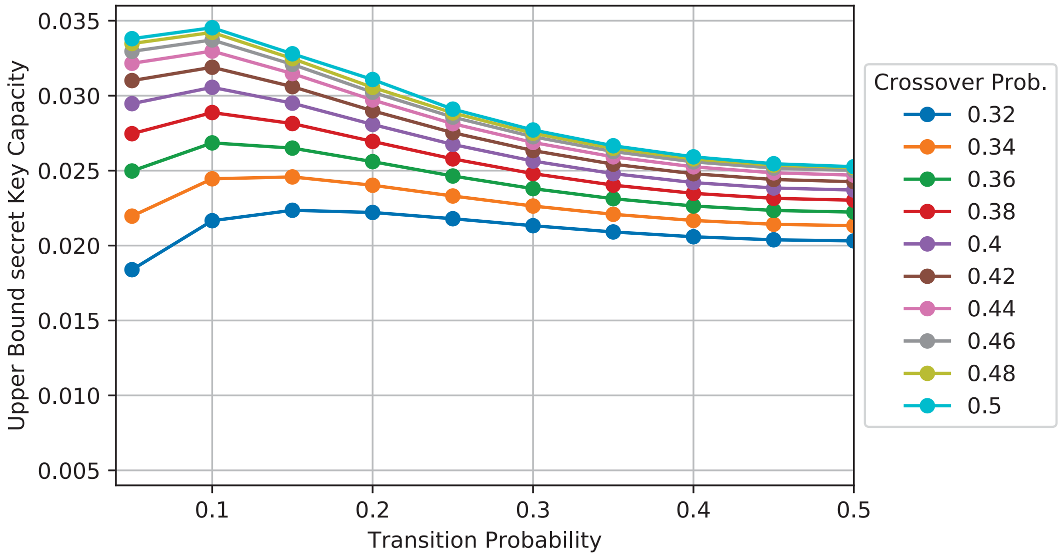

Figure 3.4 The upper bound secret key capacity by fixing Alice and Bob's emission matrices and changing Eve's crossover probability

Then we vary Eve's crossover probability, α_Z, from .3 to .5 by increasing it .02 in each step. The upper and lower bounds of the secret key capacity in this case are depicted in Figures 3.4 and 3.5. Note that the secret key capacity is dimension-less (its operational meaning is the number of secret key bits per observed bit). It can be seen when Alice and Bob's channel are superior to the Eve's channel (α_A, $\alpha_B < \alpha_Z$), Alice and Bob can generate more secret bits. Moreover, by increasing Eve's crossover probability the gap between upper and lower bound decreases and the minimum occurs when Eve's channel is in worst case according to Alice and Bob, i.e., when $\alpha_Z = .5$. To show it more accurately, we have depicted the lower and upper bounds for four different values of $\alpha_Z = .44, .46, .48$ and .5 in Figure 3.6.

Similarly, to evaluate the effect of one of the legitimate user's channel, such as Bob, on the secret key capacity, we considered symmetric channels for Alice, Bob and Eve. We

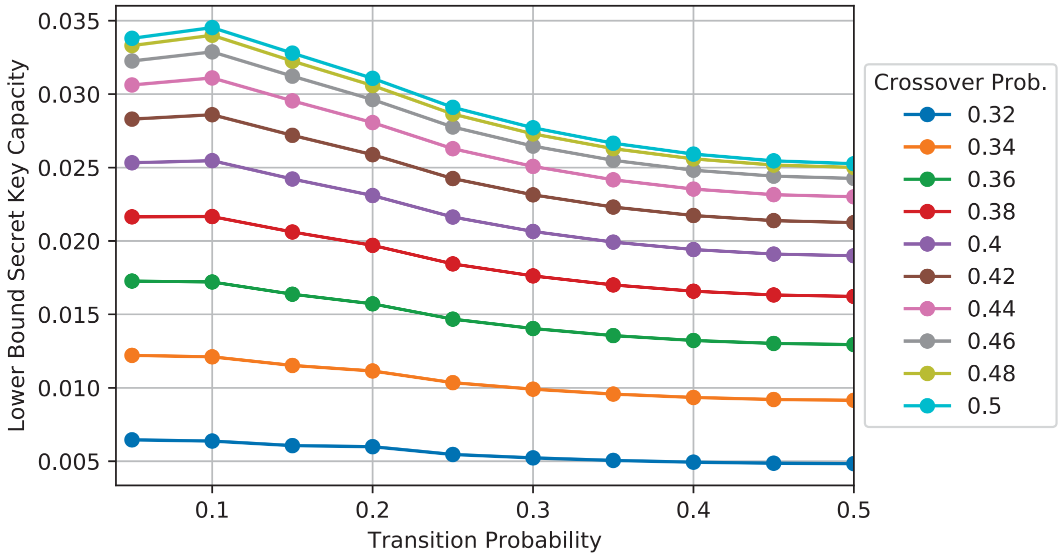

Figure 3.5 The lower bound secret key capacity by fixing Alice and Bob's emission matrices and changing Eve's crossover probability

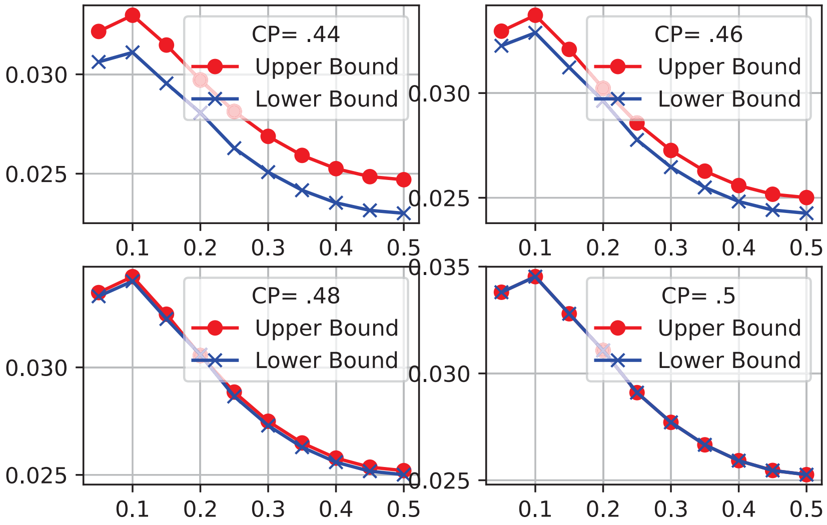

Figure 3.6 Comparing the lower and upper bound for four different values of Eve's crossover probability when Alice and Bob's emission matrices are fixed. CP in figures stands for crossover probability

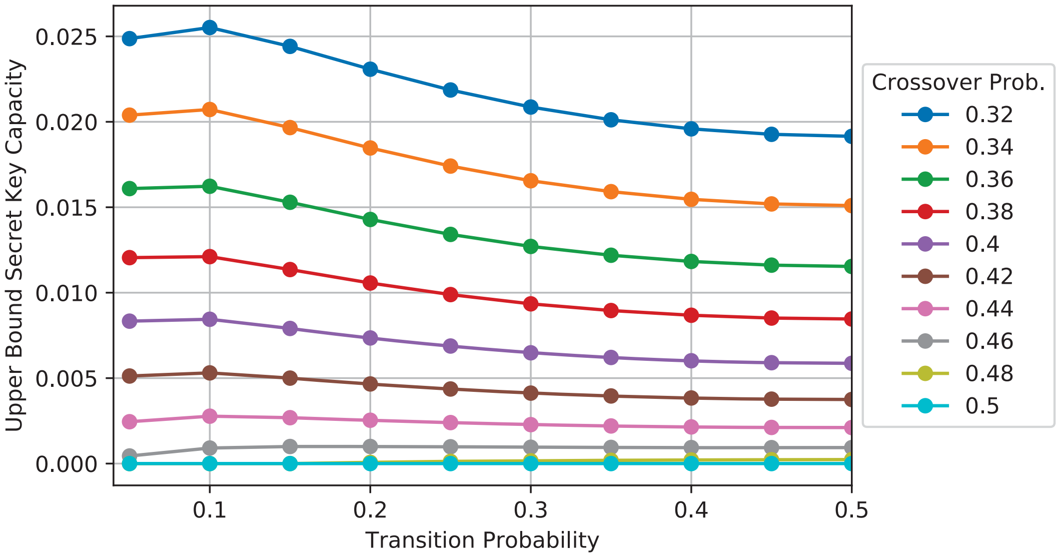

Figure 3.7 The upper bound secret key capacity by fixing Alice and Eve's emission matrices and changing Bob's crossover probability

fixed Alice and Eve's crossover probability .2 and .4 respectively and defined corresponding emission matrices as follows

$$E_A = \begin{bmatrix} .8 & .2 \\ .2 & .8 \end{bmatrix} \quad E_Z = \begin{bmatrix} .6 & .4 \\ .4 & .6 \end{bmatrix}.$$

Then we vary Bob's crossover probability, α_B, from .3 to .5 by increasing it .02 in each step. The upper and lower bounds of the secret key capacity in this case are depicted in Figures 3.7 and 3.8. Again, it can be seen when Bob's crossover probability $.4 < \alpha_B < .5$, the legitimate user's channels are not superior to Eve's channel (α_A, $\alpha_B < \alpha_Z$). So, they can generate almost nothing. Moreover, the gap between upper and lower bound is less when α_B is closer to .3. Since, Eve's channel is worst than legitimate users in this case.

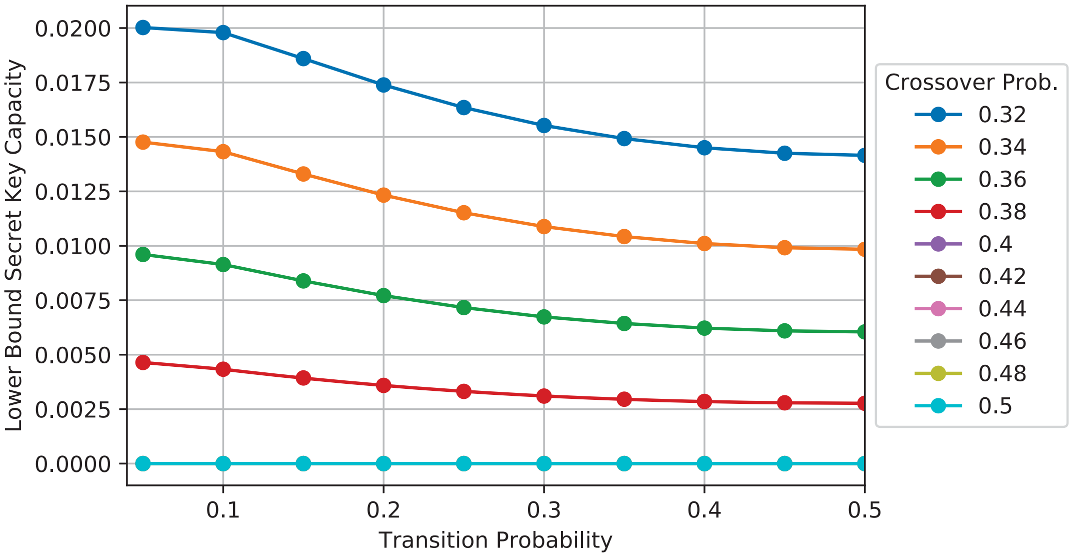

Figure 3.8 The lower bound secret key capacity by fixing Alice and Eve's emission matrices and changing Bob's crossover probability

3.8 Conclusion

To come closer to real-life scenarios in the context of common-randomness-based secret key establishment, we investigated the secret key potential of the SHMM. We provided an upper and a lower bound on the secret key capacity of the model, by extending existing single-shot results. While directly calculating the bounds for sequences of small length is computationally feasible, when the observed sequence gets larger the task becomes quickly intractable. We were able to calculate the upper and lower bounds in the asymptotic case, based on a technique involving Lyapunov exponents of Markov random matrices, and we provided the lower and upper bounds for the secret key capacity for several scenarios. It is interesting to note that the asymptotic case enables not only the calculation of the bounds, but also the the protocol to achieve the lower bound, while in the non-asymptotic (but large sequence length) scenarios, not only is it intractable to calculate the bounds directly, but even if the lower bound were given the protocol to achieve it is no longer feasible. Future work could focus on characterizing the secret key capacity of the SHMM in the non-asymptotic regime.

BIBLIOGRAPHY

[1] C. E. Shannon, “Communication theory of secrecy systems,” *Bell system technical journal*, vol. 28, no. 4, pp. 656–715, 1949.

[2] R. W. Yeung, *Information theory and network coding.* Springer Science & Business Media, 2008.

[3] A. D. Wyner, “The wire-tap channel,” *Bell system technical journal*, vol. 54, no. 8, pp. 1355–1387, 1975.

[4] I. Csiszár and J. Korner, “Broadcast channels with confidential messages,” *IEEE transactions on information theory*, vol. 24, no. 3, pp. 339–348, 1978.

[5] U. E. Maurer, “Secret key agreement by public discussion from common information,” *IEEE Transactions on Information Theory*, vol. 39, pp. 733–742, May. 1993.

[6] R. Ahlswede and I. Csiszar, “Common randomness in information theory and cryptography – Part I: secret sharing,” *IEEE Transactions on Information Theory*, vol. 39, pp. 1121–1132, July 1993.

[7] M. H. Yassaee, M. R. Aref, and A. Gohari, “Achievability proof via output statistics of random binning,” *Information Theory, IEEE Transactions on*, vol. 60, no. 11, pp. 6760–6786, 2014.

[8] R. Renner and S. Wolf, “Smooth rényi entropy and applications,” in *IEEE International Symposium on Information Theory (ISIT).* IEEE, 2004, pp. 233–233.

[9] ——, “Simple and tight bounds for information reconciliation and privacy amplification,” in *International Conference on the Theory and Application of Cryptology and Information Security.* Springer, 2005, pp. 199–216.

[10] S. Watanabe and M. Hayashi, “Non-asymptotic analysis of privacy amplification via rényi entropy and inf-spectral entropy,” in *IEEE International Symposium on Information Theory (ISIT).* IEEE, 2013, pp. 2715–2719.

[11] M. Santha and U. V. Vazirani, “Generating quasi-random sequences from semi-random sources,” *Journal of Computer and System Sciences*, vol. 33, no. 1, pp. 75–87, 1986.

[12] C. H. Bennett, G. Brassard, C. Crépeau, and U. M. Maurer, “Generalized privacy amplification,” *Information Theory, IEEE Transactions on*, vol. 41, no. 6, pp. 1915–1923, 1995.

[13] H. Tyagi and A. Vardy, “Universal hashing for information-theoretic security,” *Proceedings of the IEEE*, vol. 103, no. 10, pp. 1781–1795, 2015.

[14] W. Diffie and M. E. Hellman, “New directions in cryptography,” *Information Theory, IEEE Transactions on*, vol. 22, no. 6, pp. 644–654, 1976.

[15] M. Bellare and C. Namprempre, “Authenticated encryption: Relations among notions and analysis of the generic composition paradigm,” in *Advances in CryptologyASIACRYPT 2000.* Springer-Verlag, 2000, pp. 531–545.

[16] S. K. Park and K. W. Miller, “Random number generators: good ones are hard to find,” *Communications of the ACM*, vol. 31, no. 10, pp. 1192–1201, 1988.

[17] B. Sunar, “True random number generators for cryptography,” in *Cryptographic Engineering.* Springer, 2009, pp. 55–73.

[18] R. Ahlswede and I. Csiszar, “Common randomness in information theory and cryptography– Part II: cr capacity,” *Information Theory, IEEE Transactions on*, vol. 44, no. 1, pp. 225 –240, jan 1998.

[19] J. W. Wallace and R. K. Sharma, “Automatic secret keys from reciprocal mimo wireless channels: measurement and analysis,” *Trans. Info. For. Sec.*, vol. 5, pp. 381–392, September 2010.

[20] M. Bloch, J. Barros, M. Rodrigues, and S. McLaughlin, “Wireless information-theoretic security,” *Information Theory, IEEE Transactions on*, vol. 54, no. 6, pp. 2515 –2534, june 2008.

[21] C. Ye, S. Mathur, A. Reznik, Y. Shah, W. Trappe, and N. Mandayam, “Information-theoretically secret key generation for fading wireless channels,” *Inf. Forensics and Security, IEEE Transactions on*, vol. 5, no. 2, pp. 240 –254, June 2010.

[22] A. Agrawal, Z. Rezki, A. Khisti, and M. Alouini, “Noncoherent capacity of secret-key agreement with public discussion,” *Information Forensics and Security, IEEE Transactions on*, vol. 6, no. 3, pp. 565 –574, sept. 2011.

[23] Q. Wang, H. Su, K. Ren, and K. Kim, “Fast and scalable secret key generation exploiting channel phase randomness in wireless networks,” in *INFOCOM, 2011 Proceedings IEEE*, april 2011, pp. 1422 –1430.

[24] K. Ren, H. Su, and Q. Wang, “Secret key generation exploiting channel characteristics in wireless communications,” *Wireless Communications, IEEE*, vol. 18, no. 4, pp. 6 –12, august 2011.

[25] T.-H. Chou, S. Draper, and A. Sayeed, “Key generation using external source excitation: Capacity, reliability, and secrecy exponent,” *Information Theory, IEEE Transactions on*, vol. 58, no. 4, pp. 2455 –2474, April 2012.

[26] C. Ye and P. Narayan, “Secret key and private key constructions for simple multiterminal source models,” *Information Theory, IEEE Transactions on*, vol. 58, no. 2, pp. 639 –651, Feb. 2012.

[27] A. Khisti, S. Diggavi, and G. Wornell, “Secret-key generation using correlated sources and channels,” *Information Theory, IEEE Transactions on*, vol. 58, no. 2, pp. 652 –670, Feb. 2012.

[28] D. B. Johnson, D. A. Maltz, and Y. C. Hu, “The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR),” Tech. Rep., 2007. [Online]. Available: http://tools.ietf.org/html/rfc4728

[29] G. Brassard and L. Salvail, “Secret-key reconciliation by public discussion.” Springer-Verlag, 1994, pp. 410–423.

[30] M. R. K. Shoja, G. Amariucai, S. Wei, and J. Deng, “KERMAN: A key establishment algorithm based on harvesting randomness in MANETs,” in *Proc. of the International Conference on Security and Management (SAM’16)*, Las Vegas, NV, USA, July 25-28 2016, pp. 324–330.

[31] M. R. Khalili-Shoja, G. T. Amariucai, S. Wei, and J. Deng, “Secret common randomness from routing metadata in ad hoc networks,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 8, pp. 1674–1684, 2016.

[32] J. W. Wallace, M. A. Jensen, A. L. Swindlehurst, and B. D. Jeffs, "Experimental characterization of the mimo wireless channel: Data acquisition and analysis," *IEEE Transactions on Wireless Communications*, vol. 2, no. 2, pp. 335–343, 2003.

[33] L. Rossi, J. Chakareski, P. Frossard, and S. Colonnese, "A poisson hidden markov model for multiview video traffic," *IEEE/ACM Transactions on Networking (TON)*, vol. 23, no. 2, pp. 547–558, 2015.

[34] D. P. Heyman and T. Lakshman, "Source models for vbr broadcast-video traffic," *IEEE/ACM transactions on networking*, vol. 4, no. 1, pp. 40–48, 1996.

[35] F. N. Stokman and P. Doreian, *Evolution of social networks.* Gordon and Breach, 1997.

[36] P. N. Krivitsky and M. S. Handcock, "A separable model for dynamic networks," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 76, no. 1, pp. 29–46, 2014.

[37] K. Early and J. Z. Kolter, "An additive autoregressive hidden markov model for energy disaggregation," in *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[38] M. Hayashi and S. Watanabe, "Non-asymptotic and asymptotic analyses on markov chains in several problems," in *Information Theory and Applications Workshop (ITA).* IEEE, 2014, pp. 1–10.

[39] M. Hayashi, H. Tyagi, and S. Watanabe, "Secret key agreement: General capacity and second-order asymptotics," *IEEE Transactions on Information Theory*, vol. 62, no. 7, pp. 3796–3810, 2016.

[40] C.-D. Fuh, "On bahadur efficiency of the maximum likelihood estimator in hidden markov models," *Statistica Sinica*, pp. 127–154, 2004.

[41] M. R. K. Shoja, G. T. Amariucai, Z. Wang, S. Wei, and J. Deng, "Asymptotic converse bound for secret key capacity in hidden markov model," in *IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2017, pp. 1968–1972.

[42] N. Datta and R. Renner, "Smooth entropies and the quantum information spectrum," *IEEE Transactions on Information Theory*, vol. 55, no. 6, pp. 2807–2815, 2009.

[43] M. Hayashi, "Tight exponential analysis of universally composable privacy amplification and its applications," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7728–7746, 2013.

[44] M. Tomamichel and M. Hayashi, "A hierarchy of information quantities for finite block length analysis of quantum tasks," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7693–7710, 2013.

[45] M. Hayashi and S. Watanabe, "Uniform random number generation from markov chains: Non-asymptotic and asymptotic analyses," *IEEE Transactions on Information Theory*, vol. 62, no. 4, pp. 1795–1822, 2016.

[46] T. S. Han, *Information-spectrum methods in information theory.* Berlin, Germany:Springer-Verlag, 2003.

[47] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Booz-Allen and Hamilton Inc Mclean Va, Tech. Rep., 2001.

[48] J. Soto, "Statistical testing of random number generators," in *Proceedings of the 22nd National Information Systems Security Conference*, vol. 10, no. 99. NIST Gaithersburg, MD, 1999, p. 12.

[49] U. M. Maurer, “A universal statistical test for random bit generators,” *Journal of cryptology*, vol. 5, no. 2, pp. 89–105, 1992.

[50] H. Tyagi and S. Watanabe, “A bound for multiparty secret key agreement and implications for a problem of secure computing,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques.* Springer, 2014, pp. 369–386.

[51] M. Hayashi, H. Tyagi, and S. Watanabe, “Secret key agreement: General capacity and second-order asymptotics.”

[52] D. R. Stinson, “Universal hash families and the leftover hash lemma, and applications to cryptography and computing,” *Journal of Combinatorial Mathematics and Combinatorial Computing*, vol. 42, pp. 3–32, 2002.

[53] V. Y. Tan, “Asymptotic estimates in information theory with non-vanishing error probabilities,” *arXiv preprint arXiv:1504.02608*, 2015.

[54] S. Kuzuoka, “On the redundancy of variable-rate slepian-wolf coding,” in *International Symposium on Information Theory and its Applications (ISITA).* IEEE, 2012, pp. 155–159.

[55] H. Furstenberg and H. Kesten, “Products of random matrices,” *The Annals of Mathematical Statistics*, vol. 31, no. 2, pp. 457–469, 1960.

[56] C.-D. Fuh and Y. Mei, “Quickest change detection and kullback-leibler divergence for two-state hidden markov models,” *IEEE Transactions on Signal Processing*, vol. 63, no. 18, pp. 4866–4878, 2015.

[57] K. Dajani and C. Kraaikamp, *Ergodic theory of numbers.* Cambridge University Press, 2002.

[58] M. Viana, *Lectures on Lyapunov exponents.* Cambridge University Press, 2014, vol. 145.

[59] P. Jacquet, G. Seroussi, and W. Szpankowski, “On the entropy of a hidden markov process,” *Theoretical computer science*, vol. 395, no. 2-3, pp. 203–219, 2008.

[60] R. Gharavi and V. Anantharam, “An upper bound for the largest lyapunov exponent of a markovian product of nonnegative matrices,” *Theoretical computer science*, vol. 332, no. 1-3, pp. 543–557, 2005.

[61] M. Pollicott, “Computing entropy rates for hidden markov processes,” *Entropy of hidden Markov processes and connections to dynamical systems, London Math. Soc. Lecture Note Ser*, pp. 223–245, 2011.

[62] J. Vanneste, “Estimating generalized lyapunov exponents for products of random matrices,” *Physical Review E*, vol. 81, no. 3, p. 036701, 2010.

[63] T. Holliday, A. Goldsmith, and P. Glynn, “Capacity of finite state channels based on lyapunov exponents of random matrices,” *IEEE Transactions on Information Theory*, vol. 52, no. 8, pp. 3509–3532, 2006.

APPENDIX A. INFORMATION THEORY

Definition 9. *Consider X to be a random variable with probability distribution $P_X(x)$ whose entropy is defined as follows:*

$$H(X) = \sum_{x} p_X(x) log \frac{1}{p_X(x)} \tag{A.1}$$

Definition 10. *Consider X and Y are two random variables with joint probability distribution $P_{XY}(x,y)$. Joint entropy and mutual information are defined respectively as follows.*

$$H(X,Y) = \sum_{x,y} p_{XY}(x,y) log \frac{1}{p_{XY}(x,y)} \tag{A.2}$$

$$I(X;Y) = \sum_{x,y} p_{XY}(x,y) log \frac{p_{XY}(x,y)}{p_X(x)p_Y(y)}$$

Definition 11. *Consider X and Y are two random variables with joint probability distribution $P_{XY}(x,y)$. Conditional entropy of X given Y is defined as follows.*

$$H(X|Y) = \sum_{x,y} p_{XY}(x,y) log \frac{1}{p_{XY}(x|y)} \tag{A.3}$$

Definition 12. *Let assume $e(x;x^n)$ is the empirical distribution of x in a sequence x^n, i.e., $e(x;x^n) = \frac{N(x;x^n)}{n}$, where $N(x;x^n)$ is the number of occurrences of x in the sequence x^n. Strongly ϵ-typical set $T_\epsilon^n(X)$ is then defined as follows:*

$$T_\epsilon^n(X) = \{x^n \mid \quad |e(x;x^n) - p(x)| \le \epsilon, \quad \forall x \in \mathcal{X}\} \tag{A.4}$$

Definition 13. *Let assume $e(x,y;x^n,y^n)$ is the empirical distribution of (x,y) in a sequence (x^n,y^n), i.e., $e(x,y;x^n,y^n) = \frac{N(x,y;x^n,y^n)}{n}$ where $N(x,y;x^n,y^n)$ is the number of occurrences of (x,y) in the sequence (x^n,y^n). Then strongly ϵ-typical set $T_\epsilon^n(X,Y)$ is defined as follows.*

$$T_\epsilon^n(X,Y) = \{(x^n,y^n) \mid \quad |e(x,y;x^n,y^n) - p(x,y)| \le \epsilon, \quad \forall x \in \mathcal{X}\} \tag{A.5}$$

Let consider a discrete memoryless channel (DMC) depicted in Figure. Base on the achievability part of channel coding theorem, it can be proven that there exists a $(n, 2^{nR})$ code such that, for every $R < I(X;Y)$, the average probability of error goes to zero as n goes to infinity.

An important theorem in distributed source coding that remains in the root of the achievability proof of secret-key generation in i.i.d case is the Slepian-Wolf theorem. Based on Slepian-Wolf theorem, the lossless coding rate can be expressed as follows:

$$\begin{aligned} R_1 &\geq H(X_1|X_2) \qquad \text{(A.6)} \\ R_1 &\geq H(X_2|X_1) \\ R_1 + R_2 &\geq H(X_1, X_2) \end{aligned}$$

APPENDIX B. LYAPUNOV EXPONENT

To understand the Lyapunov exponent, we start with *linear cocycles.* In dynamical systems, linear cocycles are defined as follows [58].

Let $(M, \mathcal{B}, \mu)$ be a probability space, $f : M \to M$ be measure-preserving map and $A : M \to Gl(d)$ be a measurable function, where $Gl(d)$ is the set of invertible $d \times d$ matrices. Then

$$\begin{gathered} F : M \times R^d \to M \times R^d \\ (x, v) \to (f(x), A(x)v) \end{gathered} \tag{B.1}$$

is a linear cocycle. It can be seen that

$$F^n(x, v) = (f^n(x), A^n(x)v)$$

where we denoted $A^n(x) = A(f^{n-1}(x))...A(f(x))A(x)$.

It can be proved [58] that, if $\int_M \log \|A(x)\| d\mu(x) < \infty$ for any f-invariant probability measure μ, then there exists $\lambda : M \to R$ such that the following limit holds $\mu - a.e.$:

$$\lim_{n\to\infty} \frac{1}{n} \log \|A^n(x)\| = \lambda(x) \tag{B.2}$$

Moreover, if μ is ergodic then $\lambda(x)$ is constant and is called *the Lyapunov exponent* [55].

For our purposes, products of random matrices can be easily modeled as linear cocycles. To see this, let us assume that $\{M_i\}$ is the sequence of random matrices and consider f as a shift map as follows:

$$\begin{aligned} f : M \to M \\ f(M_i, M_{i+1}, M_{i+2}, ...) = (M_{i+1}, M_{i+2}, ...). \end{aligned} \tag{B.3}$$

Now consider A as follows:

$$\begin{aligned} A : M &\rightarrow Gl(d) \\ A(M_i, M_{i+1}, M_{i+2}, ...) &= M_i \end{aligned} \tag{B.4}$$

Hence, $A^n(x)$ is the product of a sequence of random matrices, and the results for linear cocycles can be applied to the product of random matrices.

The investigation of Lyapunov exponents in the context of HMMs backs to the study of entropy rate in these models. In this context, [59] shows that the joint probability distribution of the observed sequence can be expressed as the product of random matrices. Then, it shows that the entropy rate of the HMM is the same as the top Lyapunov exponent of this product of random matrices. However, since calculating Lyapunov exponents is usually not tractable, [59] proposes an entropy rate evaluation method which does not depend on the computation of the Lyapunov exponent.

Several techniques have been proposed in the literature for calculating or estimating Lyapunov exponents. In [60], the authors have derived an upper bound for the Lyapunov exponent. Pollicott in [61] used numerical methods to approximate the entropy rate in HMMs, which was expressed based on Lyapunov exponents. Another novel method for computing Lyapunov exponents was proposed in [62] – it is based on solving numerically an eigenvalue problem.

Interestingly, the application of Lyapunov exponents has not been restricted to the study of entropy rates in the HMMs. The authors in [63] show that the capacity of finite state Markov channels can also be represented in terms of Lyapunov exponents. To derive this capacity, [63] uses the product-of-random-matrices technique outlined above.