Information-Theoretic Secret Key Agreement

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
Rector Magnificus, prof.dr. R.A. van Santen, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op dinsdag 26 februari 2002 om 16.00 uur

door

Shengli Liu

geboren te Wuji, Hebei, China

Dit proefschrift is goedgekeurd door de promotoren:

prof.dr.ir. H.C.A. van Tilborg

en

prof.dr.ir. B.J.M. Smeets

CIP-DATA LIBRARY TECHNISCHE UNIVERSITEIT EINDHOVEN

Liu, Shengli

Information-theoretic secret key agreement / by Shengli Liu. –
Eindhoven : Technische Universiteit Eindhoven, 2002.
Proefschrift. – ISBN 90-386-1001-7

NUGI 811 Subject headings : cryptology / information theory
2000 Mathematics Subject Classification : 94A60, 94A15

Printed by Universiteitsdrukkerij Technische Universiteit Eindhoven

Contents

Chapter 1

Introduction

1.1 Why Information-Theoretic Security?

Most of the currently used private key systems or public key schemes base their security on *computational security*. A cryptographic system is said to be computationally secure if the amount of work to break such a system significantly exceeds the computational resources available to an adversary. For instance, if with the best known attacking method, the adversary still has to spend many years to break a system, even when assuming a constantly increasing computer speed, then we consider that system to be computationally secure. However, the current computationally secure systems are not guaranteed to be secure in the future, since new, better attacking techniques may be developed to reduce the security level of the system.

Contrary to computational security, *information-theoretic security* assumes no limit on the adversary's computing resources. An information-theoretically secure system can be safe even if the adversary has unlimited computing power. Information-theoretic security is also called unconditional security. It is the stronger type of security model.

In this thesis, we will study secret key agreement based on information-theoretic security. Secret key agreement is an important subject in cryptography. It deals with the problem how a secret key is generated and agreed to by two legitimate users. The traditional way to implement a secret key agreement is that a user encrypts a secret key with a public key cryptosystem, and then transmits it to the other user over a public channel. However, public key schemes are usually based on the assumption of intractability of some computational problem, for example factoring larger integers or taking discrete logarithms in some field. Hence, they are only computationally secure. On the other hand, it is shown in [71] that the two "intractable" problems mentioned above can be solved efficiently if quantum computers would come into being. That is why we are motivated to study secret key agreement based on information-theoretic security. On the other hand, we point

out that systems based on information-theoretic security are less practical than those systems based on computational security like public key cryptosystems. For instance, no digital signature scheme is feasible with information-theoretic security. In the context of a large open network, public key infrastructure facilitates the key administration. However, the need to share secret keys between any two parties in the network implies that information-theoretically secure systems can not be practically used.

Information-theoretic security is based on information theory, which is in turn based on probability theory and statistics. We will introduce in Section 1.2 some basic concepts and theorems that will be used in subsequent chapters. The process to accomplish information-theoretic secret key agreement will be described in Section 1.3. Practical scenarios are presented in Section 1.4. We shall investigate the scenarios throughout this thesis. Finally we will give an outline of the contents of this thesis in Section 1.5.

1.2 Preliminaries

1.2.1 Basic Concepts in Discrete Probability Theory

A *sample space* Ω is a finite or countably-infinite set. An element ω of Ω is called an *elementary event*, and a subset $\mathcal{A}$ of Ω is called an *event*. By $|\mathcal{A}|$ we denote the cardinality of the set $\mathcal{A}$. The *probability function* Pr maps a subset of Ω to a real number between 0 and 1. It satisfies the following properties.

(1) $0 \leq \Pr[\mathcal{A}] \leq 1$ for $\mathcal{A} \subseteq \Omega$;

(2) $\Pr[\emptyset] = 0$;

(3) $\Pr[\Omega] = 1$;

(4) $\Pr[\mathcal{A}] \leq \Pr[\mathcal{B}]$ if $\mathcal{A} \subseteq \mathcal{B}$;

(5) $\Pr[A \cup \mathcal{B}] = \Pr[\mathcal{A}] + \Pr[\mathcal{B}] - \Pr[\mathcal{A} \cap \mathcal{B}]$;

Since $\Pr[\mathcal{A} \cap \mathcal{B}] \geq 0$, we have $\Pr[A \cup \mathcal{B}] \leq \Pr[\mathcal{A}] + \Pr[\mathcal{B}]$. This inequality is called the *union bound*. If $\Pr[\mathcal{A} \cap \mathcal{B}] = \Pr[\mathcal{A}] \cdot \Pr[\mathcal{B}]$, the two events $\mathcal{A}$ and $\mathcal{B}$ are called *independent*. The *conditional probability* of $\mathcal{A}$ given $\mathcal{B}$ is defined as

$$\Pr[\mathcal{A}|\mathcal{B}] = \frac{\Pr[\mathcal{A} \cap \mathcal{B}]}{\Pr[\mathcal{B}]},$$

if $\Pr[\mathcal{B}] > 0$.

A random variable X is a mapping from Ω to some finite set $\mathcal{X}$ ($\mathcal{X}$ is called the *alphabet*), and the *probability distribution* P_X of X is given by

$$\Pr[X = x] = \sum_{\omega : X(\omega) = x} \Pr[\omega].$$

In this thesis, we will also use $P_X(x)$ to denote $\Pr[X = x]$. The joint probability distribution P_{XY} of two random variables $X : \Omega \to \mathcal{X}$ and $Y : \Omega \to \mathcal{Y}$ is the probability distribution over $\mathcal{X} \times \mathcal{Y}$. This can be generalized to a finite number of multiple random variables. The conditional probability of X given that $Y = y$ is

$$\Pr[X = x|Y = y] = \frac{\Pr[X = x, Y = y]}{\Pr[Y = y]},$$

if $\Pr[Y = y] \neq 0$. If for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$,

$$\Pr[X = x, Y = y] = \Pr[X = x] \cdot \Pr[Y = y]$$

holds, X and Y are called *independent*.

Definition 1.2.1 *When two probability distributions P_X and P_Y are defined on the same set, say $\mathcal{X}$, the variational distance between P_X and P_Y is defined by*

$$d_v(P_X, P_Y) = \frac{1}{2}\sum_{x\in\mathcal{X}} |P_X(x) - P_Y(x)|.$$

From now on, we assume $\mathcal{X} \in \mathbb{R}$. The *expected value* of a real-valued random variable X is defined as

$$E[X] = \sum_{x\in\mathcal{X}} x\Pr[X = x]$$

and its *variance* is

$$Var[X] = E[(X - E[X])^2] = E[X^2] - E[X]^2.$$

The expected value of a real-valued function $f : \mathcal{X} \to \mathbb{R}$ is defined as

$$E_X[f(X)] = \sum_{x\in\mathcal{X}} f(x)\Pr[X = x].$$

A real-valued function f is called *convex* [58] on the interval $[a, b]$ if for all $x_1, x_2 \in [a, b]$ and $0 \leq \lambda \leq 1$,

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2).$$

For any convex function,

$$f(E[X]) \leq E_X[f(X)]$$

holds, which is called the *Jensen inequality*.

Random variables $X_1, X_2, \ldots, X_n$ form a *Markov chain* if

$$\Pr[X_i = x_i|X_1 = x_1, X_2 = x_2, \ldots, X_{i-1} = x_{i-1}] = \Pr[X_i = x_i|X_{i-1} = x_{i-1}]$$

for all $i > 1$.

Chebychev's inequality shows that for any real-valued random variable X and any $t \in \mathbb{R}\,,\ t > 0$,

$$\Pr\left[|X - E[X]| \geq t\right] \leq \frac{Var[X]}{t^2}.$$

The following lemma was proposed in [31] (see also in [30]). It gives an upper bound that will be frequently used in the subsequent chapters.

Lemma 1.2.2 (Kolmogorov, [31]) *For any real-valued random variable X and real number $u \geq 0$, the following two inequalities hold:*

$$Pr[X \geq r] \leq E[e^{(X-r)u}] \text{ and } Pr[X \leq r] \leq E[e^{(r-X)u}].$$

If $\mathcal{X} = \{1, 2, \ldots, n\}$ and $\Pr[X = i] = \binom{n}{i} p^i (1-p)^{n-i}$, where $0 \leq p \leq 1$, the random variable X is called *binomially distributed* with parameters n and p. We denote this by $X \sim Binomial(n, p)$.

Applying Lemma 1.2.2 to a binomially distributed random variable X, we have the following theorem.

Theorem 1.2.3 *Let $X \sim Binomial(n, p)$ and $z = \frac{r}{n-r} \cdot \frac{1-p}{p}$, where $0 < p < 1$ and $0 < r < n$. Then*

$$Pr[X \geq r] \leq \frac{(pz + 1 - p)^n}{z^r}$$

for integers r satisfying $r \geq n \cdot p$, and

$$Pr[X \leq r] \leq \frac{(pz + 1 - p)^n}{z^r}$$

for integers r satisfying $r \leq n \cdot p$.

Proof: Since $\Pr[X = j] = \binom{n}{j} p^j (1-p)^{n-j}$, we infer from Lemma 1.2.2 that for all real values $u \geq 0$

$$\Pr[X \geq r] \leq E[e^{(X-r)u}] = \sum_{j=0}^{n} \binom{n}{j} p^j (1-p)^{n-j} e^{(j-r)u} = \left(p \cdot e^u + (1-p)\right)^n / e^{ur}.$$

Replace e^u by z and define $f(z) = \ln\left(E[z^{(X-r)}]\right) = \ln((pz + 1 - p)^n / z^r)$. Then $f'(z) = ((n-r)pz - (1-p)r)/(z(pz + 1 - p))$, and $f(z)$ achieves its minimum when $z = \frac{r}{n-r} \cdot \frac{1-p}{p}$. To ensure $z \geq 1$ ($e^u \geq 1$), $r \geq n \cdot p$ is required.

Similarly, for all real values $u \geq 0$ and $z = e^{-u}$ ($z \leq 1$), probability

$$\Pr[X \leq r] \leq E[z^{(r-X)}] = (pz + 1 - p)^n / z^r$$

is minimized for $z = \frac{r}{n-r} \cdot \frac{1-p}{p}$. For $z = \frac{r}{n-r} \cdot \frac{1-p}{p} \leq 1$, r should satisfy $r \leq n \cdot p$. $\square$

1.2.2 Basic Concepts in Information Theory

In this thesis, $\log(x)$ denotes the binary logarithm of x. Let $X : \Omega \to \mathcal{X}$ and $Y : \Omega \to \mathcal{Y}$ be random variables with probability distributions P_X and P_Y.

Definition 1.2.4 (Shannon, [69]) *The Shannon entropy of a random variable X is defined as*

$$H(X) = -\sum_{x \in \mathcal{X}} P_X(x) \log P_X(x) = E_X[-\log P_X].$$

Here $0 \log 0$ is defined as 0 since $\lim_{p \downarrow 0} p \log 1/p = 0$. The Shannon entropy $H(X)$ measures the average number of bits to describe a realization of a random variable X. In other words, it measures the average *uncertainty* about X. When there exists an x with $P_X(x) = 1$, then the uncertainty about X achieves its minimum of 0; When X is uniformly distributed, i.e., $P_X(x) = 1/|\mathcal{X}|$, the uncertainty about X achieves its maximum of $\log |\mathcal{X}|$. That means $0 \leq H(X) \leq \log |\mathcal{X}|$. When X is a binary random variable, its entropy is completely characterized by $p = \Pr[X = 0]$, and $h(p) = -p \log p - (1-p) \log(1-p)$ is called the *binary entropy function*.

Definition 1.2.5 *The conditional entropy of a random variable X given a random variable Y is defined as*

$$H(X|Y) = E_Y[H(X|Y = y)] = -\sum_{y \in \mathcal{Y}} P_Y(y) H(X|Y = y),$$

where $H(X|Y = y)$ is the entropy determined by the probability distribution $P_{X|Y=y}$.

The quantity $H(X|Y)$ measures the amount of uncertainty about X when Y is known. When the probability distribution P_{XY} is given, $H(XY)$ can be defined like $H(X)$ in Definition 1.2.4. Then $H(XY) = H(X|Y) + H(Y)$ follows.

Definition 1.2.6 *The mutual information between two random variables X and Y is defined as*

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X).$$

The quantity $I(X;Y)$ measures how much information Y gives about X. It is nonnegative since $H(X) \geq H(X|Y)$ and is 0 when X and Y are independent of each other. When $I(X;Y)$ is arbitrarily small, we say X and Y have a *negligible correlation*.

Similarly, the *conditional mutual information* of X and Y given Z can be defined as

$$I(X;Y|Z) = H(X|Z) - H(X|YZ) = H(Y|Z) - H(Y|XZ).$$

Information theory led to two fundamental developments in communication theory. One is the *source coding theory*, which shows that the description of a stochastic source can be compressed to a length arbitrarily close to its entropy, but further compression results in decoding errors. The other is the *channel coding theory*,

which shows that the *capacity* of a communication channel is the maximal rate for which the information can be transmitted over the channel reliably.

Suppose that a random variable X is the input to a channel, which outputs a random variable Y. Then the channel is characterized by $P_{Y|X}$, and its capacity $C(P_{Y|X})$ is determined by the maximal value of $I(X;Y)$, taken over all P_X, i.e.,

$$C(P_{Y|X}) = \max_{P_X} I(X;Y).$$

Besides Shannon entropy, there are other entropy measures that apply to different contexts. The following two entropies are useful for the so-called privacy amplification process to measure how many secret bits can be generated (we will discuss privacy amplification in detail in Chapter 3).

Definition 1.2.7 *Let X be a random variable with distribution P_X. The* collision probability *is given by*

$$P_c(X) = \sum_{x \in \mathcal{X}} P_X(x)^2.$$

The Rényi entropy *of order 2, Rényi entropy for short, is defined as minus the 2-log of the collision probability of X, i.e.,*

$$H_2(X) = -\log P_c(X).$$

The min-entropy *of X is defined as*

$$H_\infty(X) = -\log \max_{x \in \mathcal{X}}(P_X(x)).$$

From Jensen's inequality, we have

$$H_2(X) = -\log P_c(X) = -\log E_X[P_X] \leq -E_X[\log P_X] = H(X).$$

We also have

$$H_2(X) \geq -\log \sum_{x \in \mathcal{X}} (P_X(x) \max_{x \in \mathcal{X}}(P_X(x))) = -\log \max_{x \in \mathcal{X}}(P_X(x)) = H_\infty(X).$$

Therefore,

$$H_\infty(X) \leq H_2(X) \leq H(X).$$

The equalities hold when X is uniformly distributed.

Similarly, $H_\alpha(X|Y)$ can be defined for $\alpha = \{2, \infty\}$.

1.3 Information-Theoretic Secret Key Agreement

Another fundamental result of information theory is the necessary condition for the so-called perfect secrecy. Before we introduce the definition of perfect secrecy and its necessary condition, Shannon's classical model of secrecy systems is described.

Suppose that there are two legitimate users, namely Alice and Bob, who want to communicate with each other secretly. They are able to use an insecure channel, and Eve can see every message transmitted over the channel. A secret key K is shared between Alice and Bob, and Eve knows nothing about K. Let M be the plaintext that Alice wants to transmit to Bob over the insecure channel. Using K, Alice first encrypts M to a ciphertext C, then sends it to Bob over the insecure channel. After Bob gets C, he decrypts it using K to get M.

A cryptosystem is called *perfectly-secret*, if $I(M;C) = 0$, i.e., M and C are statistically independent. In other words, Eve gains no extra information about M after she gets C.

A cryptosystem is called *uniquely decodable*, if a unique plaintext M can be determined from the ciphertext C and the key K, i.e., $H(M|CK) = 0$.

Theorem 1.3.1 (Shannon, [70]) *For every perfectly-secret uniquely decodable cryptosystem, $H(K) \geq H(M)$ holds.*

Proof: $H(K) \geq H(K|C) = H(K|C) + H(M|CK) = H(MK|C)$
$= H(M|C) + H(K|MC) \geq H(M|C)$.

Hence $I(M;C) = H(M) - H(M|C) \geq H(M) - H(K)$. The fact that $I(M;C) = 0$ in a perfectly-secret system leads to $H(K) \geq H(M)$. □

An example for perfectly-secret cryptosystems is the *one-time pad*, or *Vernam cipher*. If the plaintext $\underline{M} = (M_1, M_2, \ldots, M_n)$ is an n-bit string and the key $\underline{K} = (K_1, K_2, \ldots, K_n)$ is uniformly distributed over $\{0,1\}^n$, then the ciphertext $\underline{C} = (C_1, C_2, \ldots, C_n)$ is determined by $C_i = M_i \oplus K_i$, $i = 1, 2, \ldots, n$. It is easy to see that $H(\underline{K}) = n$, $H(\underline{K}|\underline{MC}) = 0$, $I(\underline{M};\underline{K}) = 0$, and $I(\underline{K};\underline{C}|\underline{M}) = n$, which results in $I(\underline{M};\underline{C}) = 0$.

Theorem 1.3.1 shows that perfect secrecy requires that the length of the secret key is at least as long as that of the plaintext. Perfect secrecy seems to be impossible unless Alice and Bob share a secret key beforehand.

However, if Shannon's model for secrecy systems is slightly changed, perfect secrecy is achievable without a secret key between Alice and Bob. An important observation is that noise exists in any communication system, so Eve may not receive an exact copy of what Alice sends to Bob. A. Wyner [75] is the first researcher who investigated the problem of transmitting secret messages over noisy channels. In [75], a discrete memoryless channel, characterized by $P_{Y|X}$, was considered connecting sender Alice and receiver Bob, and another discrete memoryless channel, characterized by $P_{Z|Y}$, connecting Bob with the adversary Eve. Suppose that Alice sends a message X, Bob gets Y while Eve obtains Z, then $X \to Y \to Z$ is a Markov chain. Later, I. Csiszár and J. Körner [15] generalized Wyner's model by assuming that Alice sends a message to Bob and Eve through two discrete memoryless channels, which are characterized by $P_{YZ|X}$. Then Wyner's model of concatenation of channels becomes a special case, namely $P_{YZ|X} = P_{Y|X} \cdot P_{Z|Y}$, of Csiszár's and Körner's broadcast channels. It was shown that when Eve's channel is noisier than Bob's, Alice can always transmit secret information at some rate to Bob, i.e., the advantage between Alice and Bob can always be converted to secrecy. However,

when Eve's channel is superior to Bob's, no secrecy is achievable at all. In [43], Maurer further perfected the model by adding a public channel between Alice and Bob so that interactive communication is possible between them. If the public channel is *authentic*, i.e., the transmissions over the public channel cannot be modified or suppressed by Eve, it was proved that secrecy can be achieved even if Eve's channel is better than Bob's. Below we give Maurer's model of secrecy systems.

(1) Alice, Bob, and Eve have access to

   (a) noisy communication channel(s);

   (b) a public, insecure, and error-free channel.

(2) Denote by C' the information that Eve gets about the plaintext M. A negligible correlation between C' and M is allowed.

Any message, when transmitted, suffers from noise of the communication channel. This implies that neither Bob nor Eve will always get an exact copy of what Alice sends to them. That is why we use C' instead of C in this model to denote Eve's information about M.

The assumption of a public channel is also reasonable since it is easy to obtain such a channel. On the other hand, using error correction techniques, one can assume that any message over the public channel is error-free.

The requirement for $I(C;M)$ to be 0 in Shannon's model is too strict to achieve information-theoretic security. In Maurer's model, a small correlation between C' and M is allowed. This can be described as $I(C';M) < \epsilon$, i.e, $H(M|C') = H(M) - \epsilon$, for some $\epsilon > 0$. When ϵ is very small, we call M *highly secret.*

Based on the above modified model, the notion of information-theoretic secret key agreement was proposed in [43]. After a secret key is agreed to between Alice and Bob, a one-time pad can be used to transmit plaintexts with perfect secrecy.

An information-theoretic secret key agreement can be described as follows.

(1) *Initialization phase*: through noisy communication channels, Alice, Bob, and Eve obtain random variables X, Y, and Z, respectively, which are jointly distributed according to some probability distribution P_{XYZ}.

(2) *Communication phase*: Alice and Bob exchange information over a public channel. This is known as the public discussion and can be further divided into three phases:

   (a) During the *advantage distillation* phase, Alice and Bob exchange information, denoted by a random variable U. Alice gets a new random variable A from X and U. Similarly, Bob obtains a new random variable B from Y and U. This should result in the situation that Bob has more information about Alice's random variable A than Eve has, or Alice knows more about Bob's random variable B than Eve does. In other words, $H(A|X,U) = 0$, $H(B|Y,U) = 0$, and $H(A|B) < H(A|Z,U)$ or $H(B|A) < H(B|Z,U)$.

(b) Alice and Bob exchange some redundant information to correct the discrepancies between their random variables during the *information reconciliation* phase. Denote the redundant information by V. Using V, Alice and Bob arrive at a common string S, but Eve still has some uncertainty about S. In formula, $H(S|A,V) = 0$, $H(S|B,V) = 0$, and $H(S|Z,U,V) > 0$.

(c) *Privacy amplification* enables Alice and Bob to generate a highly secret string S' from the common but partially secret S. The way to accomplish privacy amplification is that Alice chooses a proper hash function, denoted by G, and sends it to Bob. Then they compute the hash value $S' = G(S)$, hoping that $H(S'|G,Z,U,V) = H(S') - \epsilon$ for a small ϵ.

(3) *Decision phase*: Alice and Bob both either accept or reject the protocol execution, depending on whether or not they believe $S' = G(S)$ can serve as a secret key.

We will assume in the rest of the thesis that Alice and Bob are honest players and always correctly execute the protocol.

Definition 1.3.2 *A (P_{XYZ}, r, ϵ) secret key agreement protocol means that*

- *the three correlated random variables X, Y, and Z, which Alice, Bob, and Eve get in the initialization phase, have probability distribution P_{XYZ}.*
- *Alice and Bob generate an r-bit secret key S', about which Eve's information is less than ϵ. In other words, $H(S'|X,W) = 0$, $H(S'|Y,W) = 0$, and $H(S'|Z,W) = H(S') - \epsilon$, where the random variable W denotes all the information that Alice and Bob exchanged during the communication phase.*

If we independently repeat the scenario of generating X, Y and Z during the initialization phase, we arrive at the so-called secret-key rate (see [43, 45, 64]). It allows us to measure the ability of this scenario to generate secret keys asymptotically.

Definition 1.3.3 (Wolf, [64]) *Suppose that Alice, Bob, and Eve get $\underline{X} = (X_1, X_2, \ldots, X_n)$, $\underline{Y} = (Y_1, Y_2, \ldots, Y_n)$, resp. $\underline{Z} = (Z_1, Z_2, \ldots, Z_n)$ during the initialization phase. Let $P_{\underline{XYZ}} = \prod_{i=1}^{n} P_{X_iY_iZ_i}$. We use X, Y, and Z to represent X_i, Y_i, and Z_i, since the joint probability distributions of X_i, Y_i, and Z_i are the same for $i = 1, 2, \ldots, n$. The secret-key rate of X and Y with respect to Z, denoted by $S(X;Y||Z)$, is defined as the largest nonnegative real number R such that for every $\epsilon > 0$ and sufficiently large $n = n(\epsilon)$, there exists a $(P_{\underline{XYZ}}, (R-\epsilon)n, \epsilon)$ protocol for secret key agreement. More precisely, Alice and Bob get S_A and S_B respectively, which satisfy*

$$H(S_A)/n \geq R - \epsilon,$$
$$Pr[S_A \neq S_B] \leq \epsilon,$$
$$I(S_A; \underline{Z}W) \leq \epsilon,$$
$$H(S_A) \geq \log |\mathcal{S}_A| - \epsilon.$$

(Recall that W denotes the public information during the communication phase)

It is easy to see that $S(X;Y||Z)$ is related to P_{XYZ}. In [43], the following upper bound and lower bound for $S(X;Y||Z)$ were proved:

$$S(X;Y||Z) \leq \min\{I(X;Y), I(X;Y|Z)\}, \tag{1.1}$$

$$S(X;Y||Z) \geq \max\left\{I(X;Y) - I(X;Z), I(X;Y) - I(Y;Z)\right\}. \tag{1.2}$$

If Eve sends a random variable Z over a channel, which is characterized by $P_{\overline{Z}|Z}$, and gets $\overline{Z}$, we have $S(X;Y||Z) \leq S(X;Y||\overline{Z}) \leq I(X;Y|\overline{Z})$.

To state an upper bound for $S(X;Y||Z)$, the notion of *intrinsic conditional mutual information* (*intrinsic information* for short) is needed (see [54, 53]).

Definition 1.3.4 *For a distribution P_{XYZ}, the intrinsic conditional mutual information between X and Y given Z, denoted by $I(X;Y \downarrow Z)$, is defined as the infimum of $I(X;Y|\overline{Z})$, taken over all possible conditional distributions $P_{\overline{Z}|Z}$. Here $\overline{Z}$ is the output of a channel characterized by $P_{\overline{Z}|Z}$. In formula,*

$$I(X;Y \downarrow Z) = \inf\left\{I(X;Y|\overline{Z}) : P_{XY\overline{Z}} = \sum_{z\in\mathcal{Z}} P_{XYZ} \cdot P_{\overline{Z}|Z}\right\}.$$

Note that $\overline{Z}$ is only related to Z, hence $XY \to Z \to \overline{Z}$ is a Markov chain. The following inequalities hold.

$$I(X;Y \downarrow Z) \leq I(X;Y \downarrow \overline{Z}),$$

$$I(X;Y \downarrow Z) \leq I(X;Y|Z),$$

$$I(X;Y \downarrow Z) \leq I(X;Y).$$

The definition of the intrinsic information and Inequality (1.1) lead to the following theorem.

Theorem 1.3.5 (Maurer, [53]) *For arbitrary random variables X, Y, and Z,*

$$S(X;Y||Z) \leq I(X;Y \downarrow Z). \tag{1.3}$$

1.4 Practical Scenarios for Secret Key Agreement

Throughout this thesis (except for Chapter 5), we will assume the so-called satellite scenario in the initialization phase of information-theoretic secret key agreement.

Suppose that a satellite broadcasts random binary bits $\underline{U} = (U_1, U_2, \ldots, U_N)$ with low signal power. Alice, Bob, and Eve receive $\underline{X} = (X_1, X_2, \ldots, X_N)$, $\underline{Y} = (Y_1, Y_2, \ldots, Y_N)$, and $\underline{Z} = (Z_1, Z_2, \ldots, Z_N)$ through three binary symmetric channels with respective bit error probabilities p_A, p_B, and p_E. We shall assume that the three channels are independent of each other (if the channels have a certain amount of dependency they may sometimes still be transformed into independent ones, see

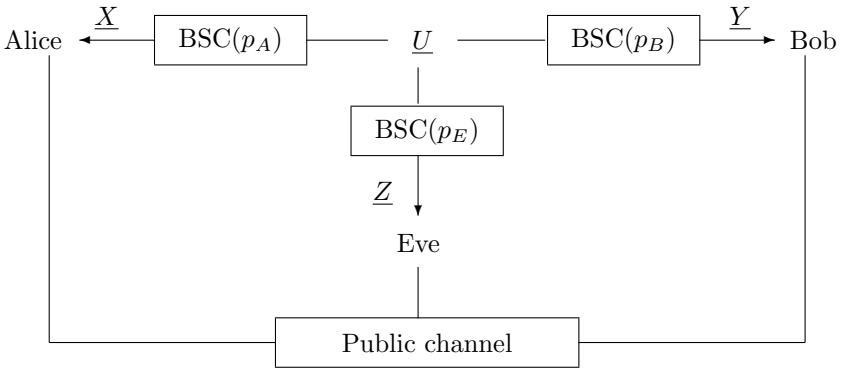

Figure 1.1: The satellite scenario

[43] for examples). Suppose that Alice and Bob are connected by a public channel, i.e., Eve can see any message between them (passive attack). The public channel can be either an *authentic* one or a *non-authentic* one. Authentic channels refers to the channels that physically unjammable. For instance, the voice channel or the newspapers. With these channels, Eve cannot modify or introduce fraudulent messages (active attack) without detection . However, such authentical channels are far from practical when Alice and Bob need to interactively exchange a large amount of data. With non-authentic channels, Eve can implement active attacks. Then the cryptographic techniques should be used to convert a non-authentic channel into a conceptually authentic channel.

This scenario uses models of binary discrete memoryless sources and channels, and such models are relatively easy to analyze and have often been considered in information theory. With this scenario for P_{XYZ}, Alice and Bob implement advantage distillation, information reconciliation, and privacy amplification over the public channel to arrive at a secret key. This scenario was first proposed by Maurer et al. (see [43]). A considerable amount of work for advantage distillation and information reconciliation has been done based on this scenario because of its simplicity and practicality [26, 23, 18, 19, 17].

Another practical scenario is that Alice uses a quantum channel to transmit the polarization information of photons to Bob. This process is also called a *quantum transmission session*. A quantum channel can be considered as a BSC channel connecting Alice and Bob, but the bit error probability is controlled by Eve, since the uncertainty principle of quantum mechanics ensures that Eve's eavesdropping introduces errors to what Bob receives. With a quantum transmission session serving as the initialization phase, quantum key agreement can achieve unconditional security. The only difference from information-theoretic secret key agreement is that there is no advantage distillation phase in quantum key agreement. The reason is that Alice and Bob only use those quantum transmission sessions, in which they have an advantage over Eve, to distill a secret key. Quantum cryptography was devel-

oped by Bennett et al. in [2]. How quantum transmission works will be discussed in Chapter 5.

1.5 Outline of the Thesis

In this thesis, we will answer the following questions:

- Given an authentic, public channel, what is the optimal way to implement advantage distillation and information reconciliation? In Chapter 2, we will first introduce the protocols proposed up to now for advantage distillation and information reconciliation. Then a protocol that combines advantage distillation and information reconciliation is presented so that Eve gets as little information as possible.

- When the public channel is non-authentic, the communications between Alice and Bob are vulnerable to Eve's active attacks, i.e., Eve may introduce fraudulent messages or modify Alice's or Bob's messages over the public channel. In the context of privacy amplification, Alice and Bob share common, though partially secret, strings. If a common string is used as an authentication key, what is the upper bound on Eve's information about the common string so that authentication is possible? How do Alice and Bob use the common string to authenticate messages during the privacy amplification phase? These questions will be answered in Chapter 3.

- Privacy amplification is a special case of secret key agreement. In the general context of secret key agreement, Alice and Bob only have access to some correlated strings. How do Alice and Bob in this case achieve authenticity to thwart Eve's active attacks on their communications over the public channel? In Chapter 4, we will study how Alice and Bob use their correlated strings obtained during the initialization phase, for authentication. When there is some advantage between Alice and Bob over Eve, Alice and Bob can use this advantage to accomplish authenticity.

- Just like in the satellite scenario, quantum key agreement can achieve unconditional security. However, a quantum channel provides another scenario, namely the quantum transmission session. The information Eve gets during a quantum transmission session is closely related to the length of the final secret key generated by Alice and Bob. How much information does Eve obtain during a quantum transmission session? In Chapter 5, we will study how to use some known information, such as the number of errors in Bob's quantum bits, the density of the light pulses, and so on, to derive a probabilistic upper bound on the amount of information Eve can learn from a quantum transmission session.

Chapter 2

Combining Advantage Distillation and Information Reconciliation

2.1 Introduction

We consider the so-called satellite scenario (see Section 1.4) in the initialization phase of an information-theoretic secret key agreement protocol. A satellite broadcasts random binary bits $\underline{U} = (U_1, U_2, \ldots, U_N)$ with low signal power. Alice, Bob, and Eve receive $\underline{A} = (A_1, A_2, \ldots, A_N)$, $\underline{B} = (B_1, B_2, \ldots, B_N)$ and $\underline{E} = (E_1, E_2, \ldots, E_N)$ through three independent, binary symmetric channels with respective bit error probabilities p_A, p_B, and p_E. After that, Alice and Bob begin public discussions over the public channel. In this chapter, we assume that the public channel connecting Alice and Bob is *authentic*. Eve can see any message between them, but she cannot modify or introduce fraudulent messages without detection. The public discussion consists of three phases, namely, *advantage distillation* (AD), *information reconciliation* (IR), and *privacy amplification* (PA).

This chapter focuses on practical protocols for advantage distillation and information reconciliation for this specific "realistic" satellite scenario. Since we want to discuss advantage distillation, it is necessary to assume that Eve has better equipment than the legitimate partners, i.e., $p_A > p_E$ and $p_B > p_E$ (otherwise Alice and Bob already have an advantage).

As defined in the previous chapter, the so-called *secret-key rate* of A and B with respect to E, denoted by $S(A;B||E)$ (see Definition 1.3.3), is the maximal rate at which Alice and Bob can generate a highly secret key by communication over the insecure, public channel, i.e., the fraction of secret bits that can be generated per realization of A, B, and E.

According to (1.2), we have

$$S(A;B||E) \geq \max\{I(A;B) - I(A;E), I(B;A) - I(B;E)\}. \tag{2.1}$$

This lower bound is not tight, but it does give a sufficient condition for the existence of an information-theoretic secret key agreement protocol. In the next section, we will show how Alice and Bob can employ the authenticity of the public channel to gain an advantage over Eve, i.e., how they start with $I(A;B) \leq I(A;E)$ and $I(A;B) \leq I(B;E)$ but arrive at the situation either $I(A;B|C) > I(A;E|C)$ or $I(A;B|C) > I(B;E|C)$ after some communication, denoted by the random variable C, over the public channel.

All known protocols for advantage distillation and information reconciliation will be presented in Section 2.2 and Section 2.3 respectively. We will also propose a general protocol in Section 2.4 to implement both advantage distillation and information reconciliation. The analysis for the proposed protocol and the corresponding simulation results of the protocol will be presented in Section 2.5. In Section 2.6, the relationship between our protocol and other known protocols will also be discussed. The conclusion of this chapter will be given in Section 2.7.

This chapter is mainly based on [36, 38].

2.2 Advantage Distillation Protocols

This section will describe the known advantage distillation protocols based on the aforementioned scenario.

2.2.1 The Repetition Code Protocol

The *repetition code* protocol is introduced in [43].

Every time Alice wants to transmit an information bit R to Bob, she generates the N-bit repetition codeword $\underline{R} = (R, R, \ldots, R)$, and transmits $\underline{R} + \underline{A} = (R + A_1, R + A_2, \ldots, R + A_N)$ over the public channel to Bob. Bob computes $\underline{R} + \underline{A} + \underline{B} = (R + A_1 + B_1, R + A_2 + B_2, \ldots, R + A_N + B_N)$. He accepts R if and only if it is exactly equal to either the all-zero codeword $\underline{0}$, in which cases he assumes that $R = 0$, or the all-one codeword $\underline{1}$, in which cases he assumes that $R = 1$. He tells Alice through the public channel whether he has accepted R.

The initial bit error probability between Alice's and Bob's strings $\underline{A}$ and $\underline{B}$ is $\epsilon_0 = p_A + p_B - 2p_Ap_B$. According to the protocol, Alice and Bob accept a bit only when $\underline{A} \oplus \underline{B} = \underline{0}$ or $\underline{1}$, which occurs with probability $\epsilon_0^N + (1 - \epsilon_0)^N$. With probability $1 - \epsilon_0^N - (1 - \epsilon_0)^N$ they get nothing. In other words, with probability $\epsilon_0^N + (1 - \epsilon_0)^N$ they distill 1 bit from their N-bit initial strings. The a posteriori bit error probability between their distilled bits is given by

$$\beta = \frac{\epsilon_0^N}{\epsilon_0^N + (1 - \epsilon_0)^N}. \tag{2.2}$$

To compute Eve's expected error probability we define

$$\begin{aligned} q_A &= 1 - p_A; \\ q_B &= 1 - p_B; \\ q_E &= 1 - p_E; \\ \alpha_{00} &= q_A q_B q_E + p_A p_B p_E; \\ \alpha_{01} &= q_A q_B p_E + p_A p_B q_E; \\ \alpha_{10} &= q_A p_B q_E + p_A q_B p_E; \\ \alpha_{11} &= q_A p_B p_E + p_A q_B q_E. \end{aligned} \tag{2.3}$$

When Alice transmits $\underline{R}+\underline{A}$ to Bob through the public channel, Eve intercepts it and she can calculate $\underline{R}+\underline{A}+\underline{E}$ using her own string $\underline{E}$. Let p_w denote the probability that the calculated vector is a particular given vector of Hamming weight $w, 0 \leq w \leq N$, given that Bob accepts a bit afterwards. Then $p_w = \alpha_{00}^{N-w}\alpha_{01}^{w} + \alpha_{10}^{N-w}\alpha_{11}^{w}$. Eve's best strategy is to guess $R = 0$ when the weight of $\underline{R} + \underline{A} + \underline{E}$, is (strictly) less than $\lceil N/2 \rceil$ and $R = 1$ otherwise. Therefore, Eve's expected error probability with respect to Alice's distilled bits is given by

$$\gamma = \frac{1}{\epsilon_0^N + (1-\epsilon_0)^N} \sum_{w=\lceil N/2 \rceil}^{N} \binom{N}{w} p_w. \tag{2.4}$$

Now we discuss the lower bound on the secret-key rate determined by the repetition code protocol. After the protocol, the mutual information between Alice and Bob is $I_B = 1 - h(\beta)$ while that between Alice and Eve is given by

$$I_E = \sum_{w=0}^{N} \binom{N}{w} \frac{p_w}{\epsilon_0^N + (1-\epsilon_0)^N} \left[1 - h\left(\frac{p_w}{p_w + p_{N-w}} \right) \right].$$

It follows from (2.1) that the secret-key rate satisfies

$$S(A;B||E) \geq r \cdot (I_B - I_E), \tag{2.5}$$

where the coefficient $r = \left[(1 - q_A q_B - p_A p_B)^N + (q_A q_B + p_A p_B)^N \right] / N$ reflects the reduction in bits for Alice and Bob (they go from N bits to 1 only when a word in the repetition code is received). For N large enough, I_B will be larger than I_E, so β will be smaller than γ and the secret-key rate $S(A;B||E)$ is strictly positive as long as $p_E > 0$.

Since r is an important parameter in the evaluation of the performance of a particular advantage distillation or information reconciliation protocol, we give a formal definition of it.

Definition 2.2.1 *Let n be the length of Alice's and Bob's original strings, where Bob's string is obtained from a transmission of Alice's string over a BSC with error probability p. Let $\underline{Q}$ be the side information that Alice and Bob exchanged over*

the public channel during an AD/IR protocol. The protocol ends with Alice having obtained string $\underline{A'}$ *and Bob* $\underline{B'}$.

Let β *be the a posteriori bit error probability between* $\underline{A'}$ *and* $\underline{B'}$, *and let* $I(\underline{A'}|\underline{Q})$ *denote Eve's information about* $\underline{A'}$ *knowing* $\underline{Q}$. *The* information rate *of the protocol is given by*

$$R_\beta(p) = 1 - I(\underline{A'}|\underline{Q})/n.$$

If Alice and Bob discard the bits from their strings needed to compensate for the information leaked to Eve because of their discussion over the public channel, their strings have shrunk after the reconciliation protocol. Thus, the information rate is just the length of the final reconciled strings divided by the length of original strings.

Without loss of generality, we may assume that it is always Bob who corrects the differences between his string and Alice's string.

The aim of AD/IR protocols is to let p approach to 0, while keeping $R_0(p)$ as large as possible. The higher value of $R_0(p)$ the protocol can achieve, the better the protocol. The amount of information exchanged over the public channel for distillation/reconciliation is lower bounded by $H(\underline{A}|\underline{B}) = n \cdot h(p)$, where $h(p) = -p\log(p) - (1-p)\log(1-p)$ denotes the *binary entropy function*. Therefore, an upper bound for $R_0(p)$ is $1 - h(p)$. Recall that log is used to denote the binary logarithm.

Note that the probability that $A_i \neq B_i, 1 \leq i \leq n$, is given by ϵ_0, so the information rate of the repetition code protocol is given by

$$R_\beta(\epsilon_0) = r = \frac{\epsilon_0^N + (1-\epsilon_0)^N}{N}, \tag{2.6}$$

where the value of N follows from (2.2) and the value of β.

2.2.2 The Iteration Protocols

The repetition code protocol has the property that when N increases, β decreases faster than γ. Hence, there exists an N_0 such that $\beta < \gamma$ when $N \geq N_0$. However, when N increases, $R_\beta(\epsilon_0)$ decreases exponentially in N. The repetition code protocol turns out to be extremely inefficient in terms of information rate. This case occurs when p_E is much smaller than p_A and p_B, since a large codeword length N has to be employed by Alice and Bob to gain an advantage over Eve. To improve the efficiency of the protocol, Maurer proposed to use iteration protocols in [42]. An iteration protocol consists of several rounds, and each round uses a repetition code protocol but with only shorter codeword length b. Alice's and Bob's strings shrink round after round, but their strings become more and more reliable. It was shown in [42] that a k-round iteration protocol with code length b corresponds to a $N = b^k$ repetition code protocol, except that the iteration protocol comes up with a larger information rate than the iteration protocol. We give the following formal definition of an iteration protocol.

Definition 2.2.2 *A k-round iteration protocol with code length b for advantage distillation, a $[b,k]$ iteration protocol for short, is a protocol that*

- *consists of k rounds and*
- *a repetition code protocol of length b is employed in each round, where the resulting bits serve as input for the next round.*

According to the above definition, a repetition code protocol of length N is just an $[N,1]$ iteration protocol.

Let us go back to the general iteration protocol. The following theorem states the performance of a general $[b,k]$ iteration protocol.

Theorem 2.2.3 *For the scenario in Section 1.4, let $\epsilon_0 = p_A + p_B - 2p_A \cdot p_B$ denote the initial bit error probability, then a $[b,k]$ iteration protocol satisfies the following properties:*

(1) Bob gets a new distilled string of bit error probability

$$\beta = \frac{\epsilon_0^{b^k}}{\epsilon_0^{b^k} + (1-\epsilon_0)^{b^k}}, \tag{2.7}$$

and the mutual information between each bit of Alice's new string and the corresponding one of Bob's new string is

$$I_B = 1 - h(\beta); \tag{2.8}$$

(2) with her best strategy, Eve can get a new string with bit error probability

$$\gamma = \frac{1}{\epsilon_0^{b^k} + (1-\epsilon_0)^{b^k}} \sum_{w=\lceil b^k/2 \rceil}^{b^k} \left(\begin{array}{c} b^k \\ w \end{array} \right) p_w, \tag{2.9}$$

with $p_w = \alpha_{00}^{b^k-w}\alpha_{01}^w + \alpha_{10}^{b^k-w}\alpha_{11}^w$, where $\alpha_{00}, \alpha_{01}, \alpha_{10}$, and α_{11} are defined in (2.3), and the mutual information between each bit of Alice's new string and that of Eve's new string is

$$I_E = \sum_{w=0}^{b^k} \left(\begin{array}{c} b^k \\ w \end{array} \right) \frac{p_w}{\epsilon_0^{b^k} + (1-\epsilon_0)^{b^k}} \left[1 - h\left(\frac{p_w}{p_w + p_{b^k-w}} \right) \right]; \tag{2.10}$$

(3) the information rate of the protocol is determined by

$$R_\beta(\epsilon_0) = \frac{1}{b^k} \frac{\epsilon_0^{b^k} + (1-\epsilon_0)^{b^k}}{\prod_{i=1}^{k-1} \left(\epsilon_0^{b^i} + (1-\epsilon_0)^{b^i} \right)^{b-1}}, \tag{2.11}$$

and the secret-key rate between Alice and Bob with respect to Eve satisfies

$$S(A;B||E) \geq R_\beta(\epsilon_0) \cdot (I_B - I_E). \tag{2.12}$$

Proof: In a $[b,k]$ iteration protocol, for every bit that Alice and Bob get after a round, b bits from the previous round are used to transmit the bit. Therefore, for each bit obtained after k rounds, a b^k-bit substring in the original string can be found related to this bit. For Alice and Bob to accept this bit, the error vector between the two b^k-bit strings should be a repetition codeword, i.e., the 1 vector or the 0 vector. That means that a $[b,k]$ iteration protocol corresponds to a repetition code protocol of length b^k (except that information rates are different), which explains the first two items in the theorem.

Now we explain how to determine the information rate $R_\beta(\epsilon_0)$ for the iteration protocol. After the first round, only a fraction $\epsilon_0^b + (1-\epsilon_0)^b$ of the original string is used to get a new shorter string (shrinking by $1/b$) which has bit error rate $\epsilon_1 = \frac{\epsilon_0^b}{\epsilon_0^b+(1-\epsilon_0)^b}$. This applies to every round. In round i, a fraction $\epsilon_{i-1}^b + (1-\epsilon_{i-1})^b$ of the string obtained in round $i-1$ is used to get a new $1/b$ shorter string of bit error rate $\epsilon_i = \frac{\epsilon_{i-1}^b}{\epsilon_{i-1}^b+(1-\epsilon_{i-1})^b} = \frac{\epsilon_0^{b^i}}{\epsilon_0^{b^i}+(1-\epsilon_0)^{b^i}}$. Hence, the information rate is $R_\beta(\epsilon_0) = \frac{1}{b^k}\prod_{i=0}^{k-1}\left[\epsilon_i^b + (1-\epsilon_i)^b\right]$, and we get (2.11) by substituting $\epsilon_i = \frac{\epsilon_0^{b^i}}{\epsilon_0^{b^i}+(1-\epsilon_0)^{b^i}}$. According to the lower bound on the secret-key rate $S(A;B||E) \geq I(A;B) - I(A;E)$, Equation (2.12) follows. □

In such an iteration protocol, both the a posteriori bit error probability β between Alice's and Bob's distilled string and the information rate $R_\beta(\epsilon_0)$ are related to the code length b and the number of rounds k. Therefore, we use a new symbol $R_\beta(\epsilon_0)[b,k]$ to denote the information rate. The following statement holds:

$$R_{\beta'}(\epsilon_0)[b,k+1] \approx \frac{R_\beta(\epsilon_0)[b,k]}{b} \tag{2.13}$$

if ϵ_0 is small or k is large, where

$$\beta' = \frac{\beta^b}{\beta^b + (1-\beta)^b}.$$

To prove the above statement, it is sufficient, according to (2.11), to prove that

$$\frac{\epsilon_0^{b^{k+1}} + (1-\epsilon_0)^{b^{k+1}}}{\prod_{i=1}^{k}\left(\epsilon_0^{b^i} + (1-\epsilon_0)^{b^i}\right)^{b-1}} \approx \frac{\epsilon_0^{b^k} + (1-\epsilon_0)^{b^k}}{\prod_{i=1}^{k-1}\left(\epsilon_0^{b^i} + (1-\epsilon_0)^{b^i}\right)^{b-1}},$$

i.e.,

$$\epsilon_0^{b^{k+1}} + (1-\epsilon_0)^{b^{k+1}} \approx \left[\epsilon_0^{b^k} + (1-\epsilon_0)^{b^k}\right]^b.$$

The above formula holds when ϵ_0 is very small or k is large (which implies that b^k is very large), since $\epsilon_0^{b^k} \approx 0$.

A $[b,k+1]$ iteration protocol has one round more than a $[b,k]$ protocol, and increases the reliability of Alice's and Bob's distilled strings with a smaller β' than

β, while the price is that the information rate $R_{\beta'}(\epsilon_0)[b, k+1]$ is reduced to a fraction $1/b$ of $R_\beta(\epsilon_0)[b, k]$.

If ϵ_0 is small, (2.13) holds even when k is small. In this case, we get

$$R_\beta(\epsilon_0)[b, k] \approx \frac{\epsilon_0^b + (1-\epsilon_0)^b}{b^k} \tag{2.14}$$

from (2.13) and $R_{\beta''}(\epsilon_0)[b, 1] = \left[\epsilon_0^b + (1-\epsilon_0)^b\right]/b$ where $\beta'' = \epsilon_0^2/(\epsilon_0^2 + (1-\epsilon_0)^2)$.

We note that the $[2, k]$ iteration protocol is the most efficient one, in terms of information rate, among all $[b, k]$ iteration protocols.

Let b_1, k_1 and b_2, k_2 be the parameters of two iteration protocols respectively. Suppose that $2 \leq b_1 < b_2$ and $b_1^{k_1} = b_2^{k_2}$. According to Theorem 2.2.3, the two iteration protocols have the same performance, for example $\beta = \beta_{b_1^{k_1}} = \beta_{b_2^{k_2}}$, except for the information rates.

If ϵ_0 is small, from (2.14) we get that $R_\beta(\epsilon_0)[b_1, k_1] \approx \left(\epsilon_0^{b_1} + (1-\epsilon_0)^{b_1}\right)/b_1^{k_1}$ and $R_\beta(\epsilon_0)[b_2, k_2] \approx \left(\epsilon_0^{b_2} + (1-\epsilon_0)^{b_2}\right)/b_2^{k_2}$. It is easy to see that

$$R_\beta(\epsilon_0)[b_1, k_1] > R_\beta(\epsilon_0)[b_2, k_2], \tag{2.15}$$

since $b_1 < b_2$.

On the other hand, even if ϵ_0 is not small, Figure 2.2 shows that (2.15) still holds. Therefore, we conjecture that $[2, k]$ iteration protocol is the most efficient iteration protocol. We show the information rate $R_\beta(\epsilon_0)[b, k]$ as a function of $N = b^k$ for

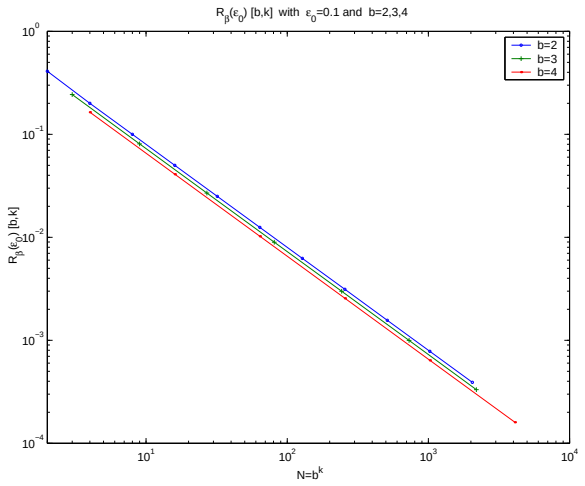

Figure 2.1: The information rate $R_\beta(\epsilon_0)[b, k]$ as a function of $N = b^k$ for $\epsilon_0 = 0.1$ for a $[b, k]$ iteration protocol

$\epsilon_0 = 0.1$ (a small value) in Figure 2.1 and for $\epsilon_0 = 0.4$ (a large value) in Figure 2.2. It is easy to see that the curves for different b's are almost straight lines with slopes $\log_{10} b$ because $\log_{10} R_\beta(\epsilon_0)[b, k] - \log_{10} R_\beta(\epsilon_0)[b, k+1] \approx \log_{10} b$.

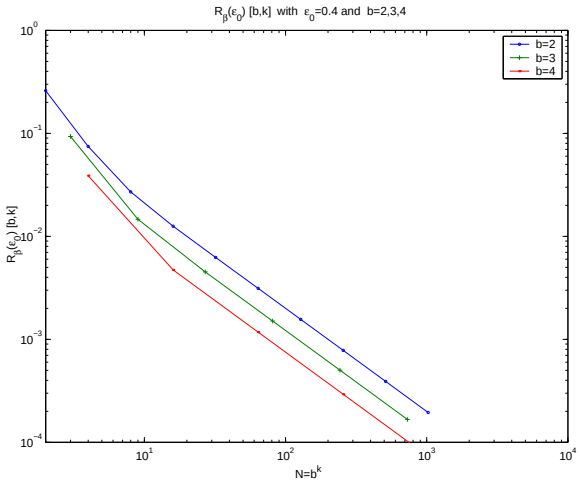

Figure 2.2: The information rate $R_\beta(\epsilon_0)[b,k]$ as a function of $N = b^k$ for $\epsilon_0 = 0.4$ for a $[b,k]$ iteration protocol

2.2.3 The Bit Pair Iteration Protocol

The *bit pair iteration* protocol is proposed by Gander et al. in [26] (see also [8]). It consists of a number of rounds and operates on the binary strings of Alice and Bob. In each round, Alice and Bob perform the following steps:

They both divide their strings into pairs of bits.

Round i: Alice sends Bob the parity of each of her bit pairs over the public channel. Bob computes the parity bit of each of his bit pairs, and compares it to the parity bit received from Alice.

If the parities match, Bob announces OK on the public channel. The first bit of the pair is discarded to compensate for the information that leaked to Eve with the parity bit. The second bit of the pair is retained for the next round.

If the parities differ, the bit pair is discarded entirely.

The retained bits (those not discarded) are taken together and form the input bit string for the next round.

In each round of the protocol, Alice and Bob accept a bit if and only if their parities for a pair of bits coincides with each other. In that case, the *error pattern* between the two pairs is either (0,0) or (1,1). It is easy to see that after round i, Alice and Bob accept a bit with probability $\epsilon_{i-1}^2 + (1-\epsilon_{i-1})^2$, and Bob's bit error probability will decrease from ϵ_{i-1} to $\epsilon_i = \frac{\epsilon_{i-1}^2}{\epsilon_{i-1}^2+(1-\epsilon_{i-1})^2}$. Therefore, a k-round bit pair iteration protocol is just a $[2,k]$ iteration protocol with information rate $R_\beta(\epsilon_0)[2,k]$. It has the same effect as long repetition code protocols but with a much higher efficiency in terms of information rate. Round after round, Eve's advantage is reduced even though Eve has more information about Alice's bits than

Bob does in the first few rounds.

The lower bound on the secret-key rate derived in [26] is given by

$$S(X;Y||Z) \geq R_\beta(\epsilon_0)[2,k] \cdot (I_B - I_E)\,, \tag{2.16}$$

where I_B and I_E have the same value as in (2.5), but

$$R_\beta(\epsilon_0)[2,k] = \frac{1}{2^k} \frac{\epsilon_0^{2^k} + (1-\epsilon_0)^{2^k}}{\prod_{i=1}^{k-1} \left(\epsilon_0^{2^i} + (1-\epsilon_0)^{2^i}\right)} \approx \frac{\epsilon_0^2 + (1-\epsilon_0)^2}{2^k}. \tag{2.17}$$

Recall that the information rate of the repetition code protocol of length 2^k is $R_\beta(\epsilon_0)[2^k,1] = \frac{\epsilon_0^{2^k} + (1-\epsilon_0)^{2^k}}{2^k}$. We see that $R_\beta(\epsilon_0)[2,k] >> R_\beta(\epsilon_0)[2^k,1]$. As we claimed in the previous subsection, the $[2,k]$ iteration protocol, thus the bit pair iteration protocol, is the most efficient iteration protocol.

2.3 Known Practical Information Reconciliation Protocols

Information reconciliation takes place after advantage distillation in information-theoretic secret key agreement, i.e, it is performed after Alice and Bob have gained an advantage over Eve in terms of mutual information between their strings. The need for information reconciliation first showed up in the practical quantum key agreement protocol [2]. During this protocol, Alice and Bob first use a quantum channel to transmit quantum bits. Possible eavesdropping by Eve and natural noise inherent to practical quantum facilities (such as a malfunction of the photon detector or system misalignment), cause discrepancies (errors) between Alice's and Bob's quantum bits. Alice and Bob need to reconcile their quantum bits over an authentic public channel to eliminate all the discrepancies before they use *privacy amplification* techniques to distill a secret key.

The usual model of information reconciliation is that two channels, a binary symmetric channel (BSC) with bit error probability p and a public channel, connect Alice and Bob. Alice has a binary string $\underline{A}$, which she transmits over the BSC channel to Bob who receives it as $\underline{B}$. Alice and Bob then use the public channel to exchange some information to reconcile their strings. All information sent over the public channel can be seen by Eve. The more Eve knows, the fewer secret bits Alice and Bob can distill from the reconciled common string in the subsequent privacy amplification phase. Therefore, the aim of information reconciliation is to remove all the errors in $\underline{B}$ while reducing the information leakage to Eve.

To evaluate the performance of information reconciliation protocols, we can use the information rate $R_0(p)$ defined in Section 2.2. As we showed in Subsection 2.2.1, the upper bound for $R_0(p)$ is $1 - h(p)$. Here we assume that $\underline{A}$ and $\underline{B}$ have length n, and $h(p) = -p\log(p) - (1-p)\log(1-p)$ is the binary entropy function.

In [2], Bennett et al. discuss for the first time a reconciliation protocol that proceeds after a quantum transmission protocol. We will refer to it as the `BBBSS` protocol. In this protocol, Alice and Bob first randomly permute the bit positions

in their strings and partition them into blocks. For each block they compute and compare the parities. If Alice's and Bob's parity check bits corresponding to the same block are different, a binary search is performed to locate an error in this block (we will explain the procedure of binary search in Subsection 2.3.2). They repeat this procedure with increasing block lengths until all errors are removed. However, no rule was suggested to determine the block lengths.

Yamazaki et al. [76] and van Dijk [18] independently developed the same rule to determine the block length. The idea of the rule is to minimize the number of parities Alice and Bob have to exchange over the public channel to remove an error. Applying this rule to the `BBBSS` protocol, Yamazaki presented an *optimized BBBSS* protocol in [76]. We denote this protocol as $\texttt{BBBSS}^{opt}$. This protocol improves upon the information rate of the `BBBSS` protocol.

Brassard et al. proposed another practical protocol, named `Cascade` [7], which is performed in several passes. The block lengths are chosen such that, from Pass 2 onwards, at least half of the errors are corrected per pass. Furthermore, the error correction in some Pass i involves the error corrections in other passes. `Cascade` achieves a higher information rate than the $\texttt{BBBSS}^{opt}$ protocol when p is small, but a lower rate for larger p.

In [73], Sugimoto and Yamazaki gave an optimized version of `Cascade` by choosing block lengths for each pass in a different way, and a slight improvement on it was given in [77]. We call their protocol that was presented in [77] *optimized Cascade*, and denote it by $\texttt{Cascade}^{opt}$. According to their simulation results, $\texttt{Cascade}^{opt}$ achieves the highest $R_0(p)$ among all known practical information reconciliation protocols. We will reprint simulation results for all reconciliation protocols mentioned above in Table 2.1 in Subsection 2.3.5.

In the following subsections, we will discuss each protocol mentioned above in more detail. The framework of each protocol is described first, and the rules for selecting the block length will be presented afterwards. The ideas of these protocols will help us to develop our own general protocol which will be presented in Section 2.4.

First we will introduce the BICONF primitive since it often serves as a last major primitive in reconciliation protocols.

2.3.1 The BICONF Primitive

The BICONF primitive was first introduced in [2] (it is a combination of two primitives named BINARY and CONFIRM, see [7] for more details). It was used as a primitive in [7, 76, 73, 77]. It deals with the case when there are only a small number of errors or no errors left in Bob's string (Alice and Bob may not know that they are in this case).

Alice and Bob begin by choosing the same randomly generated subset of bits from their strings. Then Alice tells Bob the parity of her subset, and Bob checks whether his subset has the same parity. The primitive ends in case of identical parity, otherwise a binary search is performed to locate an error.

If Alice's string is different from Bob's, the BICONF primitive will detect this

with probability 1/2; if the two strings are identical, the primitive says so with probability 1. Therefore, if the primitive is executed sufficiently many times, say l times, with the same parities for Alice and Bob, Alice and Bob will be rightly convinced that their strings are identical with probability at least $1 - 2^{-l}$.

2.3.2 Optimized BBBSS

The $\texttt{BBBSS}^{opt}$ protocol was presented in [76]. Alice and Bob begin with two binary strings $\underline{a}$ and $\underline{b}$, of length n and with bit error rate p. It proceeds in several passes. After Pass i ($i = 1, 2, \dots$), the protocol arrives at two strings $\underline{a}^{(i)}$ and $\underline{b}^{(i)}$, for Alice and Bob respectively, of length $n^{(i)}$ with bit error rate $p^{(i)}$. Initialize $\underline{a}^{(0)} = \underline{a}$, $\underline{b}^{(0)} = \underline{b}$, and $p^{(0)} = p$.

Pass i can be described in the following way.

1 Randomly permute bit positions of string $\underline{a}^{(i-1)}$ (and $\underline{b}^{(i-1)}$ in the same way) so that errors are randomly distributed in $\underline{b}^{(i-1)}$.

2 Determine the optimal block length w_i using $p^{(i-1)}$ (how to do this will be shown later). Divide $a^{(i-1)}$ and $b^{(i-1)}$ into blocks of length w_i. Alice and Bob exchange parities for each block over the public channel. If the two parities are the same, they go to the next block. Otherwise, they run a binary search to locate and remove an error in the block (as a result, the block is divided into several subblocks due to the binary search), before going to the next block. The *binary search* is performed in this way,

- **(1)** Alice halves her block into two subblocks, and sends Bob the parity of the first subblock over the public channel.
- **(2)** Bob also divides his block in the same way. After he gets Alice's parity, he compares it with the parity of his own first subblock. If the two parities agree with each other, Alice and Bob perform a binary search to the second subblock, otherwise to the first subblock, until the block size is one, in which case an error has been located.

3 Alice and Bob discard the last bit of each block (or subblock) to compensate the information leakage by parity. They get new strings $\underline{a}^{(i)}$ and $\underline{b}^{(i)}$ with bit error probability $p^{(i)}$.

The bit error probability $p^{(i)}$ after Pass i is determined with the help of the number of errors, denoted by z, corrected up to Pass i in the following way:

$$p^{(i)} = \frac{np - z}{n^{(i)}}.$$

Now we show how to select the optimal block length w_i based on the bit error probability $p^{(i)}$. Suppose that the blocks have length w, then the probability of detecting an error in such a block is given by

$$\sum_{l=1}^{\frac{w}{2}} \binom{w}{2l-1} \left(p^{(i)}\right)^{2l-1} \left(1-p^{(i)}\right)^{w-2l+1} = \frac{1-\left(1-2p^{(i)}\right)^{w}}{2}, \tag{2.18}$$

where for the sake of convenience we assume that w is even.

To locate an error Alice and Bob need to exchange $1 + \log(w)$ parity check bits over the public channel (we point out that $1 + \log(w-1)$ is used in [76] which is not precise). However, if the block has an even number of errors, which happens with probability $(1 + (1 - 2p^{(i-1)})^w)/2$, only one parity bit needs to be transmitted over the public channel. Therefore, the expected number of parities that need to be exchanged for removing one error during Pass i is given by:

$$\frac{\frac{1+(1-2p^{(i-1)})^w}{2} + \frac{1-(1-2p^{(i-1)})^w}{2}(\log w + 1)}{\frac{1-(1-2p^{(i-1)})^w}{2}} = \log w + \frac{2}{1-(1-2p^{(i-1)})^w}. \quad (2.19)$$

The denominator reflects the expected number of errors that will be removed during the pass. The optimal block length w_i should minimize (2.19). However, when $w_i > n^{(i)}/2$, Alice and Bob set $w_i = n^{(i)}/2$, and run the BICONF primitives for subsequent passes. When no errors are found in a number of successive passes, say 10, Alice and Bob stop running the protocol and accept the final strings as reconciled strings. The failure probability of the protocol is about 2^{-10}.

2.3.3 Van Dijk and Koppelaar's Protocol

Van Dijk and Koppelaar developed the same rule as described in the previous subsection for determining the optimal block length from the bit error rate in [19] (see also [23]). We call their protocol DK protocol. In the DK protocol, whenever a new block needs to be created, the bit error probability is estimated first by the number of errors that have been corrected, then the optimal block length is computed by minimizing (2.19). Let the string length be n and Bob's initial bit error probability be p. Then $\Pr[l] = \binom{n}{l} p^l (1-p)^{n-l}$ is the probability that Bob's string contains l errors. The probability that Bob's string contains exactly l errors given that it contains at least z errors is given by

$$\Pr[l| \geq z] = \frac{\Pr[l]}{\sum_{j \geq z} \Pr[j]}$$

if $l \geq z$ and $\Pr[l| \geq z] = 0$ if $l < z$. The bit error probability when z errors have been corrected is estimated by

$$\widehat{p} = \sum_{l \geq z} \Pr[l| \geq z] \frac{l-z}{n}. \quad (2.20)$$

As we will point out in Subsection 2.6.4, their method of estimating bit error probability is not precise, because the information of the number of errors is not complete for estimating accurate bit error probabilities.

2.3.4 The Cascade Protocol

The `Cascade` protocol [7] also consists of several passes. In the first pass, the block length w_1 is determined by the initial bit error probability p according to some rule

(we will show the rule later). The positions of the bits in Alice's and Bob's initial strings are indexed by $1, 2, \ldots, n$. Alice and Bob divide their strings into blocks. Block v in Pass 1, denoted by $\mathbf{Block}_v^1$, is composed of those bits whose positions come from the set $K_v^1 = \{l|(v-1)w_1 < l \leq vw_1\}$. For each block, they exchange parities and perform a binary search to correct an error in case of different parities. After Pass 1, all blocks or sublocks have an even (including zero) number of errors.

In Pass $i, i > 1$, Alice and Bob set the block length to $w_i = 2w_{i-1}$. After a random permutation of their strings, they separate the strings into $\lceil \frac{n}{w_i} \rceil$ blocks, not necessarily of equal size. More precisely, they choose a random function $f_i : [1, 2, \ldots, n] \rightarrow [1, 2 \ldots, \lceil \frac{n}{w_i} \rceil]$, and the bits from the set $K_j^i = \{l|f_i(l) = j\}$ form $\mathbf{Block}_j^i$. Let $\mathcal{K}$ denote the set of all the blocks, in this and all preceding passes, containing an odd number of errors. The set $\mathcal{K}$ can be determined by Alice and Bob by exchanging the parities of all blocks over the public channel.

For the smallest block in $\mathcal{K}$ (if there are more Alice and Bob randomly choose one) a binary search is executed to locate and remove an error from this block. Let l' be the position of this error. Let $\mathcal{B}$ be the set of blocks (in this and all other passes) that contain bit l'. Update $\mathcal{K}$ by $(\mathcal{B} \bigcup \mathcal{K}) \setminus (\mathcal{B} \bigcap \mathcal{K})$. Set $\mathcal{B} \bigcap \mathcal{K}$ consists of the blocks that used to contain an odd number of errors but now contain an even number of errors due to the correction of the error in bit l'. Set $\mathcal{K}$ is updated by excluding $\mathcal{B} \bigcap \mathcal{K}$ from $\mathcal{B} \bigcup \mathcal{K}$, so the new $\mathcal{K}$ consists of the blocks containing an odd number of errors. Repeat this procedure until $\mathcal{K} = \emptyset$.

Now let us see how the block length w_i for Pass i is determined. The idea to determine w_i is to decrease the number of errors at least by half in each pass from Pass 2 onwards. Let $\delta_i(j)$ be the probability that $2j$ errors remain in K_v^1 after Pass i. Let E_i be the expected number of errors in K_v^1 after Pass i. Then the following equation holds,

$$E_i = 2 \sum_{j=1}^{\lfloor \frac{w_1}{2} \rfloor} j\delta_i(j),$$

with

$$E_1 = w_1 p - \frac{1 - (1-2p)^{w_1}}{2}.$$

Let γ_i be the probability that at least 2 errors are corrected in Pass $i, i > 1$. Then a lower bound on it is given by

$$\gamma_i \geq 1 - \left(1 - \left(1 - \frac{w_i}{n}\right)^{\frac{nE_{i-1}}{w_1}}\right)^2 \approx 1 - \left(1 - e^{-\frac{w_i E_{i-1}}{w_1}}\right)^2$$

when $n \rightarrow \infty$.

As a consequence, $\delta_i(j)$ is bounded by

$$\delta_i(j) \leq \left(\sum_{l=j+1}^{\lfloor \frac{w_1}{2} \rfloor} \delta_{i-1}(l)\right) + \delta_{i-1}(j)(1 - \gamma_i).$$

Choose w_1 such that

$$\sum_{l=j+1}^{\lfloor \frac{w_1}{2} \rfloor} \delta_1(l) \leq \frac{1}{4}\delta_1(j)$$

and

$$E_1 \leq \frac{\ln 2}{2}.$$

Let $w_{i+1} = 2w_i$ for $i > 1$. It follows that $1 - \gamma_i \leq \left(1 - e^{-2E_1}\right)^2 \leq \frac{1}{4}$. Then we have $\delta_i(j) \leq \delta_{i-1}(j)/2$, hence $E_i \leq E_{i-1}/2$.

The suggested number of passes is 4. According to the simulation for $p = 0.01$, 0.05, 0.10, 0.15 and $n = 10000$ in [7], all errors are removed after 4 passes with `Cascade`.

2.3.5 The Optimized Cascade Protocol

Sugimoto and Yamazaki did more work on the `Cascade` protocol in [73]. Observing the simulation results of `Cascade`, they found that almost all errors are corrected during the first two passes, and only very few errors were left for further passes to correct. Another observation is that almost half of the errors are removed in Pass 1, and the other half in Pass 2. The block lengths w_1 and w_2 are selected to minimize the number of publicly exchanged parities. After Pass 2, the BICONF primitive is employed in each pass to correct the few remaining errors. We call such a pass a *BICONF pass*. When one error is found in a BICONF Pass i, where $i > 2$, other errors may also be corrected by going back to blocks of previous passes that contain the newly corrected error. If no error is detected after l subsequent passes, Alice's string will agree with Bob's with probability at least $1 - 2^{-l}$.

There are np errors in Bob's string on average. If half of the errors are corrected in Pass 1, then the expected number of parities exchanged is

$$L_{\text{exp}}^{(\text{Pass } 1)} = \frac{n}{w_1} + \frac{np}{2}\log(w_1). \tag{2.21}$$

The other half is corrected by pairs in Pass 2. For any pair of errors, one is corrected in some block of Pass 2, where $\log(w_2)$ more parities are expected to be exchanged, while the other in some block of Pass 1, where about $\log(w_1)$ more parities are expected to be exchanged. Therefore, the expected number of parities in Pass 2 is

$$L_{\text{exp}}^{(\text{Pass } 2)} = \frac{n}{w_2} + \frac{np}{4}\left[\log(w_1) + \log(w_2)\right]. \tag{2.22}$$

The optimal values $w_1 = \left\lfloor \frac{4\ln 2}{3p} \right\rfloor$ and $w_2 = \left\lfloor \frac{4\ln 2}{p} \right\rfloor$ are determined by minimizing $\left(L_{\text{exp}}^{(\text{Pass } 1)} + L_{\text{exp}}^{(\text{Pass } 2)}\right)$.

Later, Yamazaki and Sugimoto made a further improvement, because the number of corrected errors in Pass 1 and Pass 2 is not half to half when p is large (for example

$p = 0.15$). According to (2.18), the probability that an error can be detected in a block of Pass 1 is given by

$$P_{\text{odd}} = \frac{1 - (1-2p)^{w_1}}{2}.$$

There are n/w_1 blocks, so about $P_{\text{odd}} \cdot n/w_1$ errors are expected to be corrected in Pass 1, and $np - P_{\text{odd}} \cdot n/w_1$ errors are left for Pass 2. Therefore, (2.21) and (2.22) should be replaced by the following two equations

$$L_{\text{exp}}^{(\text{Pass } 1)} = \frac{n}{w_1} + \frac{n \cdot P_{\text{odd}}}{w_1} \log(w_1). \tag{2.23}$$

$$L_{\text{exp}}^{(\text{Pass } 2)} = \frac{n}{w_2} + \frac{1}{2}\left(np - \frac{n \cdot P_{\text{odd}}}{w_1}\right)\left[\log(w_1) + \log(w_2)\right]. \tag{2.24}$$

The optimal w_1 and w_2 are obtained by minimizing $\left(L_{\text{exp}}^{(\text{Pass } 1)} + L_{\text{exp}}^{(\text{Pass } 2)}\right)$. There are no explicit formulas for w_1 and w_2 in this case, but numerical optimal values for w_1 and w_2 can be obtained. We show the block lengths w_1 and w_2 in Table 2.1 compared with those in the `Cascade` protocol. The corresponding simulation results for the information rate $R_0(p)$ for different protocols, with $n = 10000$ as the length of the to be reconciled strings, are reprinted in Table 2.2. Recall that $1 - h(p)$ is an upper bound of $R_0(p)$.

protocol	$p = 0.01$	$p = 0.05$	$p = 0.10$	$p = 0.15$
`Cascade`	$w_1 = 73$	$w_1 = 14$	$w_1 = 7$	$w_1 = 5$
`Cascade`^{opt}	$w_1 = 70$	$w_1 = 14$	$w_1 = 7$	$w_1 = 4$
`Cascade`	$w_2 = 146$	$w_2 = 28$	$w_2 = 14$	$w_2 = 10$
`Cascade`^{opt}	$w_2 = 301$	$w_2 = 61$	$w_2 = 31$	$w_2 = 25$

Table 2.1: Block lengths of the first two passes in `Cascade` and `Cascade`^{opt}

`protocol`	$R_0(0.01)$	$R_0(0.05)$	$R_0(0.10)$	$R_0(0.15)$
`BBBSS`^{opt}	0.8996	0.6491	0.4471	0.3063
`Cascade`	0.9090	0.6609	0.4233	0.2305
`Cascade`^{opt}	0.9139	0.6917	0.4904	0.3316
Upper bound for $R_0(p)$	0.9192	0.7136	0.5311	0.3902

Table 2.2: Information rates for different protocols for n=10000

It seems that the `Cascade`^{opt} protocol is the best protocol up to now. The question is, can we do better than the `Cascade`^{opt} protocol? The answer is affirmative, as we will show in the next section.

2.4 A General Protocol for Advantage Distillation and Information Reconciliation

Our protocol keeps the basic idea of other reconciliation protocols. During the protocol, Bob's string is being reconciled to approach Alice's string (her string is also changing due to discarding). We call these strings to be reconciled *working strings*. Our protocol is summarized as follows and will be explained in detail later.

1 Given a bit error probability ϵ for Bob's working string in some Pass i, the *main principle* for choosing the optimal block length w_i is to minimize the amount of information publicly exchanged (i.e., the information that is leaked to Eve) for correcting one error.

2 Alice and Bob first apply a random permutation to their strings of bits before each pass, and then create new blocks from their strings. The block length is optimally chosen to minimize the information leakage. A pass ends when all bits are involved in some block and Alice and Bob have the same parities for all blocks (or subblocks).

3 A new block is created only when all previously constructed blocks (or split subblocks) contain an even number of errors. For each newly created block, Alice and Bob exchange the corresponding parities. Different parities for some block indicates that an odd number of errors exist in Bob's version of this block. Alice and Bob always choose one of the blocks of minimal cardinality (if there are more they randomly agree on one) that contains an odd number of errors to perform a binary search for an error.

4 Define t to be the first pass with the optimal weight, determined by the main principle, larger than 2. The protocol can be divided into three parts:

- Pass 1 up to $t-1$. The optimal block length $w_i,\ i = 1, \ldots, t-1$, is determined to be 2 by the main principle.
- Pass t and Pass $t+1$. Almost all remaining errors will be eliminated in these two passes. Whenever an error is detected in a block of Pass $t+1$, another error can be found and removed in some block of Pass t. Since blocks in Pass t are also involved in correcting errors in Pass $t+1$, the main principle does not apply to Pass t. The optimal block length w_t should be chosen to minimize the amount of publicly exchanged information needed to remove all the remaining errors. The optimal block length w_{t+1} for Pass $t+1$ is determined by the main principle. Bob's bit error probability is decreasing during Pass $t+1$, hence w_{t+1} has to be increased correspondingly.
- Pass $i, i > t+1$. The protocol runs BICONF primitives. We call a pass which runs the BICONF primitive a BICONF pass. These passes will remove the remaining few errors left and also determine when the

protocol should cease. Finding one error also leads to error corrections in blocks of previous passes. After l successive BICONF passes without an error detected, the protocol can stop with a failure probability at most 2^{-l}.

When the initial bit error probability p is small enough to make w_1 larger than 2, the protocol turns out to consist only of the last two parts.

5 For Pass $i,\ i > t+1$, only a few errors are left (which has been verified by the simulation results in [73]), and the bit error probability $p^{(t+1)}$ after Pass $t+1$ is so small that the optimal block length $w_i,\ i > t+1$, according to the main principle, is as large as half of the length of the working strings. Consequently, there are only two blocks of almost equal length, and Alice and Bob will have to check just one of the two blocks. That explains why Pass $i,\ i > t+1$, actually runs a BICONF pass.

6 To compensate for the parity information leaked to Eve, Alice and Bob discard one bit (the first bit for example) from each block (or subblock). Therefore, the working strings are shrinking from pass to pass.

7 During a binary search for error correction, some bits are explicitly exposed. We call these bits *explicit bits*. They are the subblocks containing only one bit. Those explicit bits should be excluded from other blocks that contain them. This may create new explicit bits, which should also be excluded in the same way until no explicit bits show up any more.

In the context of the satellite scenario as described in Section 1.4, suppose that Alice has $\underline{a}$ as a particular realization of $\underline{A}$, and that Bob has $\underline{b}$. We can think of Bob having received his string $\underline{b}$ from Alice who in fact has transmitted the string $\underline{a}$ over a Binary Symmetric Channel (BSC) with error probability $p = p_A + p_B - 2p_Ap_B$.

From now on, we view the reconciliation problem from a coding theory point of view. Let $\underline{e} = \underline{a} \oplus \underline{b}$ be the *error pattern* between $\underline{a}$ and $\underline{b}$. We use an n-bit binary vector $\underline{h}$ to represent each block, and we call it the *parity check vector*. The *index set* of $\underline{h}$ records the indexes of nonzero elements in $\underline{h}$, i.e., the positions of the bits composing the block. Then Alice's parity check for the block is given by $\underline{h} \cdot \underline{a}^T$, and Bob's by $\underline{h} \cdot \underline{b}^T$. The difference between these two parity checks, $\underline{h} \cdot \underline{e}^T = 0$ or 1, is called the *syndrome*. All the row vectors $\underline{h}$ form a parity check matrix H, and $\underline{s}^T = H\underline{e}^T$ is called the *syndrome vector*. Now the problem of reconciliation becomes how to dynamically construct a parity check matrix H with the help of the known syndrome vector such that the number of parity check vectors needed for H to correct all the errors in $\underline{b}$ is as small as possible. A parity check matrix $H_{k\times n}$ uniquely (in isomorphic sense) determines an $[n, n-k]$ code, so the problem can also be described as how to construct an $[n, n-k]$ code with feedback of the corresponding syndrome information such that the code rate $(n-k)/n$ is as large as possible.

Before we continue to describe the protocol in more detail, we shall summarize the notation that will be used.

p: the bit error probability of the BSC connecting Alice and Bob (Alice transmits a random string $\underline{A}$, but Bob receives this over a BSC with error probability p and obtains string $\underline{B}$).

n: the length of Alice's initial string $\underline{A}$ and Bob's string $\underline{B}$.

$\underline{a}, \underline{b}$: concrete realizations of the random variables $\underline{A}$ and $\underline{B}$, both of which are of length n.

$\underline{e}$: the error pattern resulting from the transmission over the BSC. It is the vector of length n given by $\underline{e} = \underline{a} \oplus \underline{b}$.

H: the parity check matrix.

$\underline{h}_l$: the lth row of the parity check matrix H.

$\underline{s}$: the syndrome vector determined by $H\underline{e}^T$.

$k_{(i)}$: the number of rows in H after Pass i.

$\underline{a}^{(i)}, \underline{b}^{(i)}$: the working strings after Pass i. Their length is $n - k_{(i)}$.

$\mathcal{J}_{(i)} = \{l_1, l_2, \ldots, l_{k_{(i)}}\}$: the set of $k_{(i)}$ positions that eventually will be discarded after Pass i. (At the beginning of the protocol we set $\mathcal{J}_{(0)}$ to $\emptyset$.)

$\mathcal{Q}^{(i)}$: the valid position set during Pass i with initial value $\{1, 2, \ldots, n\} \setminus \mathcal{J}_{(i-1)}$. It decreases in size during Pass i until it is empty, i.e., no valid positions left for a vector. The initial value denotes all the $n - k_{(i-1)}$ valid bit positions which constitute the string $\underline{a}^{(i-1)}$ (and $\underline{b}^{(i-1)}$) and which will be involved in Pass i.

Construction for H starts with a rowless matrix. Set $\mathcal{J}_{(0)} = \emptyset$ and $k = 0$ at the beginning.

Description of Pass i:

Step 1 At the beginning of Pass i, Alice has $a^{(i-1)}$ and Bob $b^{(i-1)}$. The positions of the bits making up $a^{(i-1)}$ (or $b^{(i-1)}$) are given by $\mathcal{Q}^{(i)} = \{1, 2, \ldots, n\} \setminus \mathcal{J}_{(i-1)}$. Set $\mathcal{J}_{(i)} = \mathcal{J}_{(i-1)}$. Determine the current optimal weight w_i (how to determine the optimal weight will be explained in Subsection 2.5.3) for Pass i. Let t be the first pass with the optimal weight w_t larger than 2. Pass i is divided into several rounds. Each round involves the creation of a new vector (refer to item 3 in the summary of our protocol).

Step 2 In case of Pass $t + 1$, the current optimal weight is determined before every round. Alice adds a new row vector $\underline{h}_{k+1}$ of weight w_i to H which has 1's in w_i positions, say $\{l_{j_1}, l_{j_2}, \ldots, l_{j_{w_i}}\}$ in ascending order, randomly chosen from $\mathcal{Q}^{(i)}$, and 0's in the other positions. Update $\mathcal{Q}^{(i)}$ by $\mathcal{Q}^{(i)} \setminus \{l_{j_1}, l_{j_2}, \ldots, l_{j_{w_i}}\}$, and $\mathcal{J}_{(i)}$ by $\mathcal{J}_{(i)} \bigcup \{l_{j_1}\}$. Alice sends $\underline{h}_{k+1}$ and the value of the parity check $\underline{h}_{k+1} \cdot \underline{a}^T$ to Bob over the public channel. Bob also adds the vector $\underline{h}_{k+1}$ to his parity check matrix H and then calculates and tells Alice the syndrome $s_{k+1} = \underline{h}_{k+1} \cdot \underline{a}^T \oplus \underline{h}_{k+1} \cdot \underline{b}^T = \underline{h}_{k+1} \cdot \underline{e}^T$.

Both increase k by 1 and continue with the next step.

Step 3 Alice and Bob check if any of the row vectors $\underline{h}_l$, $1 \leq l \leq k$, has weight at least 2 and syndrome 1. If that is the case, they continue with Step 4, otherwise Alice and Bob check $|\mathcal{Q}^{(i)}|$.

- For Pass i, $i \leq t+1$, if $|\mathcal{Q}^{(i)}| \leq w_i$, let $w_i = |\mathcal{Q}^{(i)}|$. Go back to Step 2 for the last round;
- For Pass i, $i > t+1$, this is the last round;
- In case of the last round, Go to step 5. Otherwise, Alice and Bob go back to Step 2 for the next round.

Step 4 Alice and Bob both select the row vector $\underline{h}_m$ with $1 \leq m \leq k$, which is of lowest weight among those rows of H whose weights are at least 2 and which have syndromes equal to 1 (if there are more of such vectors, Alice and Bob select the first one).

Alice and Bob perform a binary search to locate an error in the index set corresponding to the 1-entries of $\underline{h}_m$ in the following way. Alice selects half of the elements from $\underline{h}_m$'s index set to get a new vector $\underline{h}_{k+1}$ whose weight is about half that of $\underline{h}_m$. Alice updates H by adding $\underline{h}_{k+1}$ to it and replacing $\underline{h}_m$ by $\underline{h}_m \oplus \underline{h}_{k+1}$. Alice sends $\underline{h}_{k+1}$ and $\underline{h}_{k+1} \cdot \underline{a}^T$ to Bob. Bob updates H in the same way, and feeds back the syndrome $s_{k+1} = \underline{h}_{k+1} \cdot \underline{a}^T \oplus \underline{h}_{k+1} \cdot \underline{b}^T$ to Alice.

Alice and Bob repeat Step 4 (each time increasing k by 1) until they get a vector (either $\underline{h}_k$ or $\underline{h}_m$) of weight 1 and with syndrome equal to 1, i.e., they have located an error, and Bob corrects it. Whenever a vector of weight 1 shows up in H, let l be the index of the nonzero element in the vector. Alice and Bob set all other elements in column l of H to be zero. The syndrome vector is updated correspondingly. In other words, Alice and Bob eliminate the influence of the explicit bit by removing it from all other parity check equations. Reset $\mathcal{J}_{(i)} = \{l_1, l_2, \ldots, l_k\}$, where $\{l_1, l_2, \ldots, l_k\}$ is the set of the first nonzero elements of the k row vectors in H. Update $\mathcal{Q}^{(i)}$ by $\mathcal{Q}^{(i)} \bigcap \left(\{1, 2, \ldots, n\} \setminus \mathcal{J}_{(i)}\right)$. Return to Step 3.

Step 5 Alice and Bob discard the first nonzero elements, $h_{1,l_1}, h_{2,l_2}, \ldots, h_{k_{(i)},l_{k_{(i)}}}$, of each row of H. With discarding we mean that these coordinates no longer play a role in successive passes. Removing those corresponding coordinates in $\underline{a}$ and $\underline{b}$ leads to $\underline{a}^{(i)}$ and $\underline{b}^{(i)}$.

Alice and Bob now have some set $\mathcal{J}_{(i)} = \{l_1, l_2, \ldots, l_{k_{(i)}}\}$. They both go to the next pass in Step 1.

At the end of each full pass of the protocol, Bob has changed $\underline{b}^{(i)}$ into a vector $\bar{\underline{a}}^{(i)}$ that may still differ from $\underline{a}^{(i)}$, but very likely in fewer places. Continuing with the successive passes, $\bar{\underline{a}}^{(i)}$ will approach or become equal to $\underline{a}^{(i)}$. Alice and Bob stop with the reconciliation protocol when the last l passes (out of a total of r passes with $r > l + t$) do not locate any errors. The protocol fails with probability less than 2^{-l}.

2.5 Analysis of the Protocol

2.5.1 Selecting the Weight for a New Parity Check Vector

We shall first discuss the selection of the weight of the new row vectors (parity check vectors) for the parity check matrix H, given that Bob's working string has bit error rate ϵ. This problem is equivalent to the determination of the optimal length of the blocks in Subsection 2.3.2. The principle for choosing the optimal weight is to minimize the expected number of parity check bits (syndromes) that needs to be transmitted publicly to correct one error.

Suppose that the bit error probability is ϵ and that the weight of the newly created row vector for H is w in some pass i. The binary search can only split a parity check vector and locate an error when the vector contains an odd number of errors. Hence, the probability of detecting an error in such a vector is given by

$$\sum_{i=1}^{\frac{w}{2}} \begin{pmatrix} w \\ 2i-1 \end{pmatrix} \epsilon^{2i-1}(1-\epsilon)^{n-2i+1} = \frac{1-(1-2\epsilon)^w}{2}, \tag{2.25}$$

where w is assumed to be even for simplicity.

Alice and Bob publicly exchange their parity check bits to determine the corresponding syndrome for the parity check vector. If the syndrome is 1, which happens with probability $(1-(1-2\epsilon)^w)/2$, then $\log(w)$ more parity check bits should be exchanged over the public channel to locate the error. Therefore, the expected number of parity check bits that need to be exchanged for a vector of weight w is

$$1 + \frac{1-(1-2\epsilon)^w}{2} \cdot \log(w). \tag{2.26}$$

The expected number of parity check bits for removing one error is given by (2.26) divided by (2.25), i.e.,

$$\frac{1 + \frac{1-(1-2\epsilon)^w}{2} \cdot \log(w)}{\frac{1-(1-2\epsilon)^w}{2}} = \log(w) + \frac{2}{1-(1-2\epsilon)^w}. \tag{2.27}$$

Define the function $f(w,\epsilon) = \log(w) + 2/(1-(1-2\epsilon)^w)$. It is convex for fixed ϵ. The rule for determining the optimal weight is to select a w such that the value of $f(w,\epsilon)$ is as small as possible, but with the constraint that $w \geq 2$ (since it makes no sense when $w = 1$). We have no explicit formula for the optimal weight w as a function of ϵ but numerical data are shown in Figure 2.3. We see that when ϵ is small enough w grows exponentially but when ϵ is larger than 0.20, w is 2. On the other hand, the length n of the string that is to be reconciled also has some influence on the optimal weight. For example, if the optimal weight w determined by minimizing $f(w,\epsilon)$ is larger than $n/2$, there will be two vectors, one of weight w while the other of weight less than w. In this case, it is better to choose $w = n/2$ (and to get two vectors of equal weight) because of the convexity of $f(w,\epsilon)$. Therefore, the optimal weight for

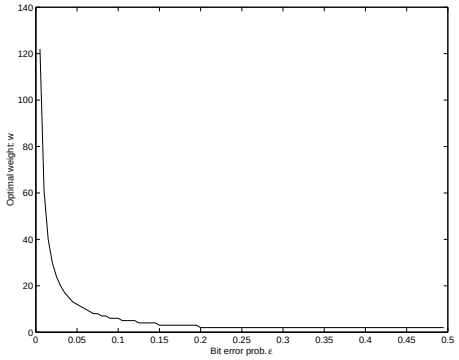

Figure 2.3: The optimal weight w for different bit error rates ϵ

constructing a new vector in our protocol is given by

$$w^{\text{opt}} = \min\left\{\max\left\{2, \left\{w | f(w,\epsilon) = \min_{x\in\mathbb{N}} f(x,\epsilon)\right\}\right\}, \frac{n}{2}\right\}, \tag{2.28}$$

where ϵ is the bit error probability and n is the length of Bob's (Alice's) working string, and $f(x,\epsilon) = \log(x) + 2/(1-(1-2\epsilon)^x)$.

It should be noted that this rule does not apply to the case that the vectors will be further split into successive passes to find other errors.

2.5.2 The Bit Error Rate after Every Pass

From (2.28) we can see that the current bit error probability plays an important role in determining the optimal weight. We will show in this section how to estimate the bit error probability for each pass. Obviously, the difference between the original strings $\underline{a}$ and $\underline{b}$ is determined by the BSC which has bit error probability p. Hence, $p^{(0)}$, the bit error probability when Pass 1 begins, is given by $p^{(0)} = p$. (In the satellite scenario $p = p_A + p_B - 2p_Ap_B$.)

The error pattern $\underline{e} = \underline{a} \oplus \underline{b}$ is a concrete realization. But for either Alice or Bob, it is still a random variable because neither is sure about the other party's string until the end of the protocol.

The columns of the parity check matrix H are indexed by the bit positions in the error vector $\underline{e}$ and the rows by bit positions in the syndrome vector $\underline{s}$. A so-called *belief network* is defined by H, in which every bit e_l is the parent of some syndromes, and each syndrome s_m is the child of some bits. The network of bits and syndromes form a bipartite graph: bits are only connected to syndromes, and vice versa.

As an example, we consider the simple case that in our protocol $t = 4$, i.e., $w_1 = w_2 = w_3 = 2$. After discarding a bit from each parity check set, a remaining bit after Pass 3 is typically related to 8 original bits. Suppose that the 8 original bits are indexed with $1, 2, \ldots, 8$. The first pass results in the first 4 rows in a parity check matrix H (see (2.29)). Then bits 1, 3, 5, and 7 are discarded (not involved

in successive passes). Bits 2, 4, 6, and 8 compose the 5th and 6th rows in H (see (2.29)) in Pass 2. Then bit 2 and bit 6 are discarded. In Pass 3, bit 4 and bit 8 compose the 7th row in H (see (2.29)), then bit 4 is discarded. We do not include the parity check vectors with syndrome 1 in H, since the two bits making up the vector will be discarded after splitting, and have nothing to do with the remaining bits. The parity check matrix for bit 8 is

$$H = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{2.29}$$

and the syndrome vector is

$$\underline{s} = (0, 0, 0, 0, 0, 0, 0).$$

The belief network concerning this bit for the first 3 passes is shown in Figure 2.4 (a). The white circles denote bits e_l while black bullets represent syndromes s_m. In Figure 2.4 (b) the same belief network is depicted in a more symmetric way. Notice that only bit 8 remains after Pass 3 since all other bits have been discarded according to our protocol.

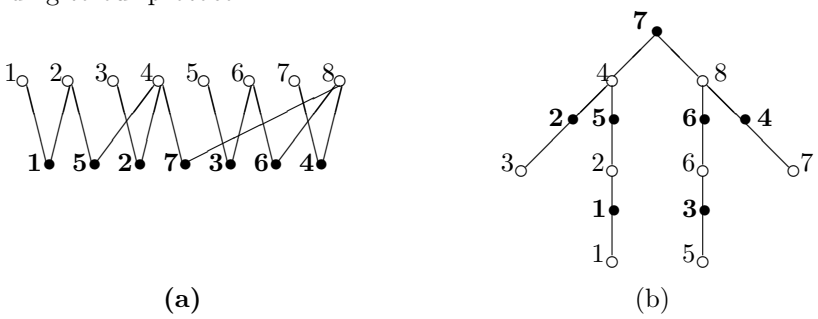

Figure 2.4: Typical belief network for a final bit after $t-1$ (=3) passes

After 3 passes, the whole belief network should consist of separate belief networks depicted in Figure 2.4, and the number of retained bits after 3 passes decides the number of separate belief networks. We point out that we neglected those vectors in H that are split since they have become discarded explicit bits. If we include these bits in the belief network, they are also separate bipartite graphs (such a typical graph can be drawn as a circle connected with a black bullet), but have nothing to do with the final reconciled bits. This is the reason why we do not include them in our belief network.

Pass t is the first pass in which the optimal weight w_t of the new vectors exceeds 2. Since each newly created check vector will either be split up or remain the same depending on the value of its syndrome, a typical belief network after Pass t should look like the graph in Figure 2.5 with the number of "leaves" being $w_t, w_t/2, w_t/4, \ldots, 2$ or 1 (for the sake of convenience we assume here that w_t is a power of 2). As a graph it is a tree. It contains no loops and its "leaves" are those bits that have not been discarded from the preceding $t-1$ passes.

Figure 2.5: Typical belief network for Pass t

For the first t passes we can determine the a posteriori error probabilities

$$\Pr\left[e_l = 1 | \underline{s} = H \cdot \underline{e}\right],\ l = 1, 2, \ldots, n,$$

for all the n bits with the *Belief Propagation Decoder* (BPD) Algorithm (originally investigated by R.G. Gallager in [24]) from [40]. The BPD-algorithm uses $\underline{p}$, $H_{k\times n}$ and $\underline{s}$ as its inputs, and can be described as follows.

BPD($\underline{p}, H_{k\times n}, \underline{s}$) Algorithm

Notation:

$\underline{p} = (p_1, p_2, \ldots, p_n)$: vector for the n individual a priori bit error probabilities.

s_m: syndrome bit of row m of H, $m = 1, 2, \ldots, k$. It is the m-th bit in $\underline{s}$.

$\mathcal{L}(m) = \{l : H_{m,l} = 1\}$: the set of bits that participate in syndrome s_m.

$\mathcal{M}(l) = \{m : H_{m,l} = 1\}$: the set of syndromes in which bit l participates.

$\mathcal{L}(m) \setminus l$: the set $\mathcal{L}(m)$ with bit l excluded.

$\mathcal{M}(l) \setminus m$: the set $\mathcal{M}(l)$ with syndrome m excluded.

q^x_{ml}: the probability that bit l of $\underline{e}$ is x, given the information obtained via syndromes other than syndrome s_m.

r^x_{ml}: the probability of syndrome s_m being satisfied if bit l of $\underline{e}$ is considered to be fixed at value x and the other bits have a separate distribution given by the probabilities $\{q^0_{ml'}, q^1_{ml'}\} : l' \in \mathcal{L}(m) \setminus l$.

Initialization:

For every (l, m) such that $H_{ml} = 1$, let $q^0_{ml} = 1 - p_l$ and $q^1_{ml} = p_l$.

Horizontal Step:

Compute two probabilities for all syndromes s_m and each $l \in \mathcal{L}(m)$. The first probability, denoted by r^0_{ml}, is the probability that the observed syndrome s_m occurs given that $e_l = 0$. In the expression below, we sum over all possible values of the other bits $\{e_{l'} : l' \neq l\}$. These have a distribution that can be expressed in the

probabilities $\{q^0_{ml'}, q^1_{ml'}\}$. We get

$$r^0_{ml} = \sum_{\{e_{l'} \mid l' \in \mathcal{L}(m) \setminus l\}} \left\{ \Pr\left[s_m \mid e_l = 0, \{e_{l'} \mid e_{l'} \in \{0,1\}, l' \in \mathcal{L}(m) \setminus l\}\right] \prod_{l' \in \mathcal{L}(m) \setminus l} q^{e_{l'}}_{ml'} \right\}. \tag{2.30}$$

The other probability, denoted by r^1_{ml}, is defined similarly but with $e_l = 1$. So, it is given by

$$r^1_{ml} = \sum_{\{e_{l'} \mid l' \in \mathcal{L}(m) \setminus l\}} \left\{ \Pr\left[s_m \mid e_l = 1, \{e_{l'} \mid e_{l'} \in \{0,1\}, l' \in \mathcal{L}(m) \setminus l\}\right] \prod_{l' \in \mathcal{L}(m) \setminus l} q^{e_{l'}}_{ml'} \right\}. \tag{2.31}$$

The conditional probabilities in these summations are either zero or one, depending on whether the observed s_m matches the hypothesized values for e_l and $e_{l'}$.

Vertical Step:

Update the values of the probabilities q^0_{ml} and q^1_{ml} with the computed values of r^0_{ml} and r^1_{ml}. For each (l, m) such that $H_{ml} = 1$, compute

$$q^0_{ml} = \alpha_{ml} \cdot (1 - p_l) \prod_{m' \in \mathcal{M}(l) \setminus m} r^0_{m'l}, \tag{2.32}$$

$$q^1_{ml} = \alpha_{ml} \cdot p_l \prod_{m' \in \mathcal{M}(l) \setminus m} r^1_{m'l}, \tag{2.33}$$

where the scalar α_{ml} is chosen such that the new values add up to one, i.e., $q^0_{ml} + q^1_{ml} = 1$.

Output:

$$q^0_l = \alpha_l \cdot (1 - p_l) \prod_{m \in \mathcal{M}(l)} r^0_{ml}, \tag{2.34}$$

$$q^1_l = \alpha_l \cdot p_l \prod_{m \in \mathcal{M}(l)} r^1_{ml}, \tag{2.35}$$

where the scalar α_l is chosen such that $q^0_l + q^1_l = 1$.

When the bipartite graph defined by the matrix H contains no cycles [61] (in our protocol this is the case for the first t passes), the algorithm will produce the exact a posteriori bit error probability q^1_l of bit l given the states of all the syndromes in a truncated belief network. The truncated belief network is formed by centering on bit l and extending out to a radius equal to twice the number of iterations of the two steps. But in the last iteration, the vertical step is replaced by the execution of the output subroutine.

Let $\underline{p}^{(i)} = (p^{(i)}_1, p^{(i)}_2, \ldots, p^{(i)}_n)$ be the error probability vector after Pass i. Let $p^{(i)}$ denote Bob's average bit error rate after Pass i. Let $H^{(i)}$ denote the part of H that is constructed during Pass i, and $\underline{s}^{(i)}$ be the corresponding syndrome vector. Let $H_{(i)}$

denote the part that is constructed during i passes and $\underline{s}_{(i)}$ be the corresponding syndrome vector. We can get the bit error probabilities $\underline{p}^{(i)} = (p_1^{(i)}, p_2^{(i)}, \ldots, p_n^{(i)})$ in the following two ways.

(1) Input $\underline{p}^{(0)} = (p, p, \ldots, p)$, $H_{(i)}$ and $\underline{s}_{(i)}$ to the BPD-algorithm, iterate i times, and output $(q_1^1, q_2^1, \ldots, q_n^1)$. Set $p_j^{(i)} = q_j^1$, for $j = 1, 2, \ldots, n$.

(2) The BPD-algorithm is executed i times. Each time run the horizontal step and the output subroutine once but with $\underline{p}^{(l-1)}$, $H^{(l)}$ and $\underline{s}^{(l)}$ as input instead, and output $\underline{p}^{(l)} = (p_1^{(l)}, p_2^{(l)}, \ldots, p_n^{(l)}) = (q_l^1, q_2^1, \ldots, q_n^1)$ where $l = 1, 2, \ldots i$.

In the following analysis for bit error probabilities for each pass, we use the latter method.

Remark. For any explicit bits, the bit error probability is either 0 or 1 depending on whether the value of the corresponding syndrome is 0 or 1. An explicit bit with syndrome 1 means an error has been located. Another note is that for those bits that are discarded before Pass l, the corresponding columns in $H^{(l)}$ will consist of 0's, and the corresponding elements in $\underline{p}^{(l)}$ will not be defined.

Pass 1 up to Pass $t-1$:

After Pass i $(1 \le i \le t-1)$, the remaining bits are mutually independent and have the same bit error probability, so the average bit error rate is just equal to any individual bit error probability.

After Pass $t-1$, a retained bit is related to a parity check matrix $H_{(2^{t-1}-1)\times 2^{t-1}}$ (like $H_{7\times 8}$ in (2.29)). A corresponding belief network like Figure 2.4 can also be drawn. Now we split $H_{(2^{t-1}-1)\times 2^{t-1}}$ into $H^{(1)}_{2^{t-2}\times 2^{t-1}}$ for Pass 1, $H^{(2)}_{2^{t-3}\times 2^{t-1}}$ for Pass 2, and so on.

In Pass 1, we have

$$H^{(1)}_{2^{t-2}\times 2^{t-1}} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \ldots & \ldots & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & \ldots & \ldots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \ldots & \ldots & 0 & 0 & 1 & 1 \end{pmatrix},$$

Initialize q_{ml}^0 and q_{ml}^1 to $1 - p^{(1)}$ and $p^{(1)}$ respectively for $m = 1, 2, 3, \ldots, 2^{t-2}$ and $l = 1, 2, 3, \ldots, 2^{t-1}$. The horizontal step results in $r_{ml}^0 = (1-p)$ and $r_{ml}^1 = p$ according to (2.30) and (2.31). Therefore the bit error probability for each bit (as well as the average bit error probability) after Pass 1 is

$$p^{(1)} = p_1^{(1)} = p_2^{(1)} = \cdots = p_{2^{t-1}}^{(1)} = \frac{p^2}{p^2 + (1-p)^2}$$

by (2.34) and (2.35). After discarding the first bit from each vector, only bits

$2, 4, \ldots, 2^{t-1}$ are left. In Pass 2, we have

$$H^{(2)}_{2^{t-3}\times 2^{t-1}} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & \ldots & \ldots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \ldots & \ldots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ldots & \ldots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \ldots & \ldots & 0 & 1 & 0 & 1 \end{pmatrix}.$$

For each (m, l) such that $H^{(2)}_{m,l} = 1$, q^0_{ml} and q^1_{ml} are initialized to the values of $1 - p^{(1)}$ and $p^{(1)}$ respectively. The horizontal step results in $r^0_{ml} = (1 - p^{(1)})$ and $r^1_{ml} = p^{(1)}$ and the bit error probability for each bit after Pass 2 is

$$p^{(2)} = p^{(2)}_2 = p^{(2)}_4 = \cdots = p^{(2)}_{2^{t-1}} = \frac{\left(p^{(1)}\right)^2}{\left(p^{(1)}\right)^2 + \left(1 - p^{(1)}\right)^2} = \frac{p^4}{p^4 + (1-p)^4}$$

according to (2.34) and (2.35).

It is easy to see that passes 1 up to $t - 1$ of the protocol turn out to run a [2, $t - 1$] iteration protocol. Therefore, the bit error probability for each bit after Pass $t - 1$ is

$$p^{(t-1)} = p^{(t-1)}_{2^{t-2}} = p^{(t-1)}_{2^{t-1}} = \frac{\left(p^{(t-2)}\right)^2}{\left(p^{(t-2)}\right)^2 + \left(1 - p^{(t-2)}\right)^2} = \frac{p^{2^{t-1}}}{p^{2^{t-1}} + (1-p)^{2^{t-1}}}$$

according to (2.11). Finally, only bit, namely 2^{t-1}, remains after discarding bit 2^{t-2}.

According to the properties of a [2, $t - 1$] iteration protocol, the information rate of the iteration protocol, or the proportion of remaining bits, is determined by (2.7), i.e.,

$$R_\beta(p)[2, t-1] = \frac{1}{2^{t-1}} \frac{p^{2^{t-1}} + (1-p)^{2^{t-1}}}{\prod_{i=1}^{t-2} \left(p^{2^i} + (1-p)^{2^i}\right)}$$

with $\beta = p^{(t-1)}$.

Pass *t*:

Pass t is the first pass with initial bit error probability $p^{(t-1)} < 0.2$. So the optimal weight should be larger than 2 if (2.28) is employed. However, (2.28) is not applicable to Pass t any more, since the vectors may be involved in error corrections in successive passes.

After Pass t, the syndrome is always 0 for those row vectors in $H^{(t)}$ with weight $w \geq 2$, otherwise the vector would have been split. That means syndrome 1 only accompanies vectors of weight 1. We will not consider the vectors of weight 1 since they are explicit bits (their bit error probabilities are either 1 or 0) and will be discarded. Let us analyze the bit error probabilities after Pass t for those bits that are in a row vector, say $\underline{h}_m$, of weight w, $w \geq 2$, with corresponding syndrome being 0. We index the nonzero elements in $\underline{h}_m$ by $1, 2, \ldots, w$ for simplicity. According

to the BPD-algorithm, q_{ml}^0 and q_{ml}^1 are initialized to $1-p^{(t-1)}$ and $p^{(t-1)}$ for $l = 1, 2, \ldots, w$. Equation (2.30) can be simplified as the probability that an even number of errors are in the $(w-1)$ remaining 1-entries (other than the l-th entry) of $\underline{h}_m$, i.e.,

$$r_{ml}^0 = \frac{1+\left(1-2p^{(t-1)}\right)^{w-1}}{2}.$$

Similarly, (2.31) is just the probability that an odd number of errors are in the $(w-1)$ remaining 1-entries of $\underline{h}_m$, i.e.,

$$r_{ml}^1 = \frac{1-\left(1-2p^{(t-1)}\right)^{w-1}}{2}.$$

According to (2.34) and (2.35), the bit error probability for each bit in $\underline{h}_m$ after Pass t is

$$p_1^{(t)} = p_2^{(t)} = \cdots = p_w^{(t)} = \frac{p^{(t-1)} \cdot \frac{1-\left(1-2p^{(t-1)}\right)^{w-1}}{2}}{p^{(t-1)} \cdot \frac{1-\left(1-2p^{(t-1)}\right)^{w-1}}{2} + \left(1-p^{(t-1)}\right) \cdot \frac{1+(1-2p^{(t-1)})^{w-1}}{2}}. \tag{2.36}$$

Pass t generally results in row vectors of different weights in $H^{(t)}$ afterwards. The error probabilities are the same for bits that are in row vectors of equal weight, but differ for those that are from vectors of different weights. Contrary to the previous passes, the elements in $\underline{p}^{(t)}$ are generally not equal. Having calculated all individual bit error probabilities, the average bit error rate $p^{(t)}$ can be calculated with those $n-k_{(t)}$ retained bits, i.e.,

$$p^{(t)} = \frac{1}{n-k_{(t)}} \sum_{c=1}^{n-k_{(t)}} p_{l_c}^{(t)}. \tag{2.37}$$

Pass i, $i > t$:

From Pass $t+1$ on, the belief network will contain loops, so the a posteriori bit error probability vector, $\underline{p}^{(i)}$, $i \geq t+1$, cannot be determined by the BPD-algorithm. In fact, the computation of the a posteriori probabilities has been shown to be intractable in [61] for belief networks that correspond to the problem $H\underline{e}'^T = \underline{s}'^T$ and contain loops. We cannot get the exact value for $\underline{p}^{(i)}$, $i \geq t+1$, but we know that the bit error rate is not fixed at $p^{(t)}$ with Pass $t+1$ going on. More precisely, it is decreasing due to the involvement of vectors of Pass t in error corrections. Below we will try to approximate the decreasing average bit error rate during Pass $t+1$ and that after Pass i, $i \geq t+1$.

The initial average bit error rate for Pass $t+1$ is $p^{(t)}$. Let $n'' = n - k_{(t)}$ be the string length before Pass $t+1$ begins. Let z count the number of errors corrected from Pass $t+1$ onwards. With the help of $p^{(t)}$ and z, we estimate the average bit error rate from Pass $t+1$ onwards in the following way.

We assume that errors in Bob's current string are binomially distributed with parameters n'' and $p^{(t)}$. Let $\Pr[l] = \binom{n''}{l} \left(p^{(t)}\right)^l \left(1-p^{(t)}\right)^{n''-l}$ be the probability that Bob's current string contains l errors. Let $\Pr[l| \geq z]$ denote the probability that Bob's working string contains l errors given that it contains at least z errors, i.e.,

$$\Pr[l| \geq z] = \frac{\Pr[l]}{\sum_{j \geq z} \Pr[j]}$$

if $l \geq z$ and $\Pr[l| \geq z] = 0$ if $l < z$. If z errors have already been corrected from Bob's working string since Pass $t+1$, there are on average at least $2z$ explicit bits (about half in error and half correct) showing up during binary searches for the z errors. Hence the current average bit error probability in Pass $t+1$, denoted by $\widehat{p}_{[z]}$, can be approximated by

$$\widehat{p}_{[z]} = \sum_{l \geq z} \Pr[l| \geq z] \frac{l-z}{n''-2z}. \tag{2.38}$$

It is easy to see that the average bit error rate $\widehat{p}_{[z]}$ is decreasing with increasing z. The average bit error rate after Pass i, $i \geq t+1$, can be approximated in the same way, i.e.,

$$p^{(i)} = \widehat{p}_{[z]},$$

where z is the number of errors corrected from Pass $t+1$ onwards.

2.5.3 Determining the Optimal Weight in Each Pass

In the previous two subsections, we have introduced the rule for choosing the optimal weight for the parity check vectors making up H, and the method of estimating the bit error rate in each pass. Now it is time to determine the optimal weight (or block length) for each pass.

Pass 1 up to Pass $t-1$

For Pass 1, the initial bit error probability is given by p $(= p^{(0)})$. Since the bit error probability does not change with the pass going on, the optimal weight determined by (2.28) remains constant during the whole pass. As shown in Figure 2.3 in Subsection 2.5.1, $w = 2$ as long as the bit error probability is at least 0.20.

Pass 1 up to Pass $t-1$ are described in the following algorithm.

Algorithm 2.5.1
Initialize $i = 1$;
While $\left(p^{(i-1)} \geq 0.20\right)$
{

(1) **Determine** $w_i^{opt} = 2$;

(2) **Construct** $H^{(i)}$ **according to the protocol;**

(3) **Run the BPD$\left(\underline{p}^{(i-1)}, H^{(i)}\right)$ algorithm and output $\underline{p}^{(i)}$ with**

$$p^{(i)} = p_j^{(i)} = \frac{\left(p^{(i-1)}\right)^2}{\left(p^{(i-1)}\right)^2 + \left(1 - p^{(i-1)}\right)^2}, \quad j = 1, 2, \ldots, n;$$

(4) **Determine the information rate by**

$$R_{p^{(i)}}(p) = \frac{\prod_{k=0}^{i-1} \left(p^{(k)}\right)^2 + \left(1 - p^{(k)}\right)^2}{2^i};$$

(5) $i \leftarrow i + 1$.

}
$t \leftarrow i$.

In these passes, the individual bit error probabilities in vector $\underline{p}^{(i)}$ are all equal, so the average bit error rate $p^{(i)}$ is also equal to the individual ones.

Pass t and Pass $t + 1$

Pass t is the first pass with an initial bit error probability $p^{(t-1)} < 0.20$. The optimal weight determined by (2.28) should have been larger than 2. However, the rule for selecting the optimal weight by (2.28) assumes that the row vectors constructed in some pass are involved in error corrections only in this very pass. Pass t is an exception to this assumption since the row vectors of Pass t may also be involved in the error corrections during Pass $t + 1$. The selection of the optimal weight w_t should minimize the total amount of leaked information to remove all remaining errors. Since the bit error probability for the remaining bits is constant during Pass t, the optimal weight w_t^{opt} is fixed as well.

On the other hand, Equation (2.28) does apply to Pass $t + 1$ because almost all remaining errors will be corrected in this pass. There may be some (very few) errors left for subsequent passes to correct. Correcting these errors may also involve the vectors of Pass $t + 1$, but the number of errors left is so small (this has been verified by simulation results in [73]) that determining the optimal weight for Pass $t + 1$ by (2.28) gives negligible deviation.

Pass $t + 1$ and Pass t are correlated in two aspects: first of all, the value of w_t influences the initial average bit error rate $p^{(t)}$ for Pass $t + 1$. Secondly, error corrections in vectors of Pass t resulting from error corrections during Pass $t + 1$ decrease the average bit error rate of Pass $t + 1$.

Now we are ready to describe how to estimate the amount of the information leaked to Eve from Pass t onwards until all errors are removed, given that

$$\underline{p}^{(t-1)} = \left(p^{(t-1)}, p^{(t-1)}, \ldots, p^{(t-1)}\right)$$

is the initial bit error probability vector and w_t is the weight for vectors (before splitting) of $H^{(t)}$. Suppose that Alice's and Bob's working strings have length n'

before Pass t begins. Let P_{odd} denote the probability that an odd number of errors exist in a row vector of weight w_t. Then $P_{\text{odd}} = \left(1 - \left(1 - 2p^{(t-1)}\right)^{w_t}\right)/2$ according to (2.25).

Let the random variable Z_1 denote the number of errors corrected during Pass t. It is binomially distributed with parameters $(\frac{n'}{w_t}, P_{\text{odd}})$. The probability that z_1 errors are corrected in Pass t is given by

$$\Pr\left[Z_1 = z_1\right] = \binom{\frac{n'}{w_t}}{z_1} {P_{\text{odd}}}^{z_1} (1 - P_{\text{odd}})^{\frac{n'}{w_t} - z_1}. \tag{2.39}$$

The amount of information (the number of syndrome bits) shown publicly during Pass t given that $Z_1 = z_1$ is determined by

$$L_{w_t}^{(\text{Pass } t)}[z_1] = n'/w_t + z_1 \log(w_t).$$

After Pass t, there are $n'/w_t - z_1$ row vectors of weight w_t in $H^{(t)}$ while z_1 row vectors have been split in binary searches to find and correct z_1 errors. Assume for a moment that w_t is a power of 2, say $w_t = 2^m$. Then a row vector of weight w_t will be split into $m+1$ vectors of weights $w_t/2, w_t/4, \ldots, 2, 1, 1$, respectively. The belief network for $H^{(t)}$ consists of a collection of independent bipartite graphs like in Figure 2.5: each of the $n'/w_t - z_1$ graphs are with w_t leaves while the other graphs are with less leaves such as $w_t/2, w_t/4, \ldots, 2, 1$.

For a typical bipartite graph with w leaves, which is determined by some row vector $\underline{h}$ in $H^{(t)}$, let the bits corresponding to the w leaves be indexed by $1, 2, \ldots, w$. According to the BPD-algorithm shown in Subsection 2.5.2, the error probabilities for the w bits, after Pass t, are given by

$$p_1^{(t)} = p_2^{(t)} = \cdots = p_w^{(t)} = \text{BER}\left(p^{(t-1)}, w\right),$$

where $\text{BER}\left(p^{(t-1)}, w\right)$ is defined by

$$\text{BER}\left(p^{(t-1)}, w\right) = \frac{p^{(t-1)} \cdot \frac{1-\left(1-2p^{(t-1)}\right)^{w-1}}{2}}{p^{(t-1)} \cdot \frac{1-\left(1-2p^{(t-1)}\right)^{w-1}}{2} + \left(1 - p^{(t-1)}\right) \cdot \frac{1+(1-2p^{(t-1)})^{w-1}}{2}}. \tag{2.40}$$

After discarding a bit from each row vector in $H^{(t)}$, we can calculate the average bit error rate after Pass t by

$$\begin{aligned} p^{(t)} = & \frac{\left(\frac{n'}{w_t} - z_1\right) \cdot (w_t - 1) \cdot \text{BER}\left(p^{(t-1)}, w_t\right)}{\left(\frac{n'}{w_t} - z_1\right) \cdot (w_t - 1) + z_1 \cdot \sum_{l=1}^{m-1} \left(\frac{w_t}{2^l} - 1\right)} \cdot \\ & + \frac{z_1 \cdot \sum_{l=1}^{m-1} \left\{\left(\frac{w_t}{2^l} - 1\right) \cdot \text{BER}\left(p^{(t-1)}, \frac{w_t}{2^l}\right)\right\}}{\left(\frac{n'}{w_t} - z_1\right) \cdot (w_t - 1) + z_1 \cdot \sum_{l=1}^{m-1} \left(\frac{w_t}{2^l} - 1\right)} \end{aligned} \tag{2.41}$$

When w_t is a power of 2, the splitting is unique (we do not care about the order). Unfortunately, not every w_t happens to be a power of 2. Hence the splitting is a probabilistic process. For instance, when $w = 7$, it can be split in the following three ways

$$7 \left\{ \begin{array}{l} 3 \\ \\ 4 \end{array} \right. \quad \left\{ \begin{array}{l} 2 \\ \\ 2 \end{array} \right. \quad \left\{ \begin{array}{l} 1 \\ 1 \end{array} \right. , \qquad 7 \left\{ \begin{array}{l} 4 \\ \\ 3 \end{array} \right. \quad \left\{ \begin{array}{l} 1 \\ \\ 2 \end{array} \right. \quad \left\{ \begin{array}{l} 1 \\ 1 \end{array} \right. , \qquad \text{and } 7 \left\{ \begin{array}{l} 4 \\ \\ 3 \end{array} \right. \quad \left\{ \begin{array}{l} 2 \\ 1 \end{array} \right. .$$

The split row vectors are with weights (3,2,1,1), (4,1,1,1), or (4,2,1).

We remark that during a bisective search to locate an error in a vector of weight w, the average number of explicit bits showing up is between 2 and 7/3. The reason is the following: during the bisective search, an error will lie in a vector of weight either 2 or 3. When the error lies in a vector of weight 2, two bits will become explicit (one in error, the other is correct); When the error lies in a vector of weight 3, with probability 1/3, only the error bit will become explicit, and with probability 2/3, three explicit bits will show up (one in error, the other two are correct), so there are $1/3 + (2/3) \cdot 3 = 7/3$ explicit bits on average. We assume for reasons of simplicity that 2 explicit bits will be exposed whenever an error is removed.

Now we show how to estimate $p^{(t)}$, taking into account the probabilistic behavior of splitting in a binary search. Suppose that w_t is divided into two halves, namely $w_l = \left\lfloor \frac{w_t}{2} \right\rfloor$ and $w_r = \left\lceil \frac{w_t}{2} \right\rceil$. Since errors are randomly located, splitting will continue with the left half with probability about $\frac{w_l}{w_t}$ while with the right half with probability about $\frac{w_r}{w_t}$. After splitting a vector of weight w_t and discarding one bit from each split sub-vectors, the number of bits contributes to the new reconciled string after Pass t is given by the following recurrence relation

$$\mathrm{NUM}(w_t) = \frac{w_l}{w_t} \left[(w_r - 1) + \mathrm{NUM}(w_l) \right] + \frac{w_r}{w_t} \left[(w_l - 1) + \mathrm{NUM}(w_r) \right]$$

with initial value $\mathrm{NUM}(1) = 0$. The sum of the corresponding error probabilities for those bits is given recursively by

$$\begin{aligned} \mathrm{PROB}\left(w_t, p^{(t-1)}\right) = & \frac{w_l}{w_t} \left[(w_r - 1) \cdot \mathrm{BER}\left(p^{(t-1)}, w_r\right) + \mathrm{PROB}\left(w_l, p^{(t-1)}\right) \right] \\ & + \frac{w_r}{w_t} \left[(w_l - 1) \cdot \mathrm{BER}\left(p^{(t-1)}, w_l\right) + \mathrm{PROB}\left(w_r, p^{(t-1)}\right) \right] \end{aligned}$$

with initial value $\mathrm{PROB}(1) = 0$.

When the vectors of weight w_t that have not been split are also taken into account, the average bit error rate after Pass t is determined by

$$p^{(t)} = \frac{\left(\frac{n'}{w_t} - z_1\right) \cdot (w_t - 1) \cdot \mathrm{BER}\left(p^{(t-1)}, w_t\right) + z_1 \cdot \mathrm{PROB}\left(w_t, p^{(t-1)}\right)}{\left(\frac{n'}{w_t} - z_1\right) \cdot (w_t - 1) + z_1 \cdot \mathrm{NUM}\left(w_t\right)}. \tag{2.42}$$

Note that (2.41) is a special case of (2.42).

The length of the working strings after Pass t is

$$n'' = n' - L_{w_t}^{(\text{Pass } t)}[z_1].$$

Let $\mathcal{J}^{(t)}$ be the set of positions of those bits discarded during Pass t. Then $|\mathcal{J}^{(t)}| = L_{w_t}^{(\text{Pass } t)}[z_1]$. The expected number of errors left after Pass t is

$$\text{Rem}_{\text{err}} = n' \cdot p^{(t-1)} - \sum_{l \in \mathcal{J}^{(t)}} p_l^{(t)}.$$

Let z_2 record the number of errors corrected from Pass $t+1$ onwards. With n'', $p^{(t)}$, and z_2 we can estimate the average bit error rate $p_{[z_2]}$, given that z_2 ($z_2 = 0, 1, \ldots, \text{Rem}_{\text{err}}$) errors are removed during Pass $t+1$, according to (2.38).

We shall now describe Pass $t+1$ in greater detail. Let $L_{w_t}^{(\text{Pass}> t)}[z_1]$ record the number of bits discarded after Pass t given that z_1 errors were corrected in Pass t. Let U record the number of bits which were involved in Pass $t+1$. U starts with 0, and increases during Pass $t+1$ until $U = n''$, i.e., all bits are involved in some row vectors in $H^{(t+1)}$.

Pass $t+1$ begins with $n'' = n' - L_{w_t}^{(\text{Pass } t)}[z_1]$, $L_{w_t}^{(\text{Pass}> t)}[z_1] = 0$, $U = 0$, $z_2 = 0$ and $p_{[z_2]} = p_{[0]} = p^{(t)}$.

Determine the optimal weight $w_{t+1}^{\text{opt}}[z_2]$ with $p_{[z_2]}$ according to (2.28). The probability that an odd number of errors occur in a vector of weight $w_{t+1}^{\text{opt}}[z_2]$ in $H^{(t+1)}$ is given by

$$P_{\text{odd}}[z_2] = \frac{1 - \left(1 - 2p_{[z_2]}\right)^{w_{t+1}^{\text{opt}}[z_2]}}{2}.$$

Hence one expects to have created $1/P_{\text{odd}}[z_2]$ vectors of weight $w_{t+1}^{\text{opt}}[z_2]$ created before a vector catches an odd number of errors (which has to be split). On average, the number of bits consumed by the $1/P_{\text{odd}}[z_2]$ vectors is about $w_{t+1}^{\text{opt}}[z_2]/P_{\text{odd}}[z_2]$, so

$$U \leftarrow U + w_{t+1}^{\text{opt}}[z_2]/P_{\text{odd}}[z_2].$$

Next, a process of correcting a pair of errors begins. The vector with an odd number of errors is split about $\log\left(w_{t+1}^{\text{opt}}[z_2]\right)$ times to locate and correct an error. Another error is corrected by splitting some vector of Pass t about $\log(w_t)$ times. The number of bits that have to be discarded in this process is about

$$D = 1/P_{\text{odd}}[z_2] + \log\left(w_{t+1}^{\text{opt}}[z_2]\right) + \log(w_t),$$

and

$$L_{w_t}^{(\text{Pass}> t)}[z_1] \leftarrow L_{w_t}^{(\text{Pass}> t)}[z_1] + D.$$

There are usually two other explicit bits showing up during the process that are known to be correct. More precisely, there are usually 4 explicit bits, two are in error and the other two are correct, among the D discarded bits. On the other hand, all the errors among the $D-4$ discarded bits (excluding the 4 explicit bits) also

disappear due to discarding. We estimate this number of errors to be $(D-4)\cdot p_{[z_2]}$. Update $\mathrm{Rem}_{\mathrm{err}}$ and z_2 by

$$\mathrm{Rem}_{\mathrm{err}} \leftarrow \mathrm{Rem}_{\mathrm{err}} - (D-4)\cdot p_{[z_2]} \tag{2.43}$$

and $z_2 \leftarrow z_2 + 2$.

Repeat the above procedure until all bits in the working strings are involved in some vectors. The number of bits contributing the last vector (before splitting) is less than or equal to the current optimal weight. Pass $t+1$ ends with Alice's and Bob's working strings of length $n'' - L_{w_t}^{(\mathrm{Pass}>\, t)}[z_1]$.

Generally, the number of the remaining errors after Pass $t+1$ (hence the average bit error rate $p^{(i)}$ in Pass i, $i > t+1$) is so small that the optimal weight w_i^{opt} for Pass i is half the length of the working string, i.e., $w_i^{\mathrm{opt}} = \left(n'' - L_{w_t}^{(\mathrm{Pass}>\, t)}[z_1]\right)/2$. Alice and Bob construct one vector of weight w_i^{opt} for error correction. Therefore, passes after $t+1$ turn out to be BICONF passes. A BICONF pass can detect an error with probability about 1/2, so we expect an average of two BICONF passes necessary to detect and remove one error. This error correction will also provoke other error corrections in the previous passes.

When the vector constructed in Pass i detects and removes an error, in each of the previous $i-1$ passes, there is a vector containing this error and another error is detected in this vector. There may be more than i errors eradicated since each of the i errors may lead to additional error corrections. It is also possible that less than i errors will be corrected but this happens with very small probability (unless the number of remaining errors is less than i, in which case Pass i may remove all the remaining errors). Nevertheless, we conservatively estimate that i errors will be removed during Pass i in this case. The expected number of discarded bits is about

$$D = 1 + \log(w_t) + \frac{\sum_{l=0}^{z'} \left(\log w_{t+1}^{\mathrm{opt}}[l]/P_{\mathrm{odd}}[l]\right)}{\sum_{l=0}^{z'} 1/P_{\mathrm{odd}}[l]} + \sum_{l=t+2}^{i+1} \log\left(w_l^{\mathrm{opt}}\right),$$

where z' denotes the number of errors corrected during Pass $t+1$. When the vector constructed in Pass i detects no error, Alice and Bob only have to discard 1 bit.

Combining the above two cases, each case with probability 1/2, we have

$$D = 1 + 0.5\cdot\left(\log(w_t) + \frac{\sum_{l=0}^{z'} \left(\log w_{t+1}^{\mathrm{opt}}[l]/P_{\mathrm{odd}}[l]\right)}{\sum_{l=0}^{z'} 1/P_{\mathrm{odd}}[l]} + \sum_{l=t+2}^{i+1} \log\left(w_l^{\mathrm{opt}}\right)\right).$$

We put

$$L_{w_t}^{(\mathrm{Pass}>\, t)}[z_1] \leftarrow L_{w_t}^{(\mathrm{Pass}>\, t)}[z_1] + D.$$

Update $\mathrm{Rem}_{\mathrm{err}}$ and z_2 by

$$\mathrm{Rem}_{\mathrm{err}} \leftarrow \mathrm{Rem}_{\mathrm{err}} - (D-2i)\cdot p_{[z_2]}, \tag{2.44}$$

and

$$z_2 \leftarrow z_2 + i.$$

Continue with BICONF passes until there is no error left ($z_2 = \mathrm{Rem}_{\mathrm{err}}$) or all bits are discarded ($L_{w_t}^{(\mathrm{Pass}>\, t)}[z_1] = n''$).

If we consider the probabilistic error correction behavior of Pass t, the total amount of information leaked to Eve, i.e., the expected number of bits discarded, from Pass t onwards, is given by

$$g\left(n', p^{(t-1)}, w_t\right) = \sum_{z_1=0}^{\max\left\{n'p^{(t-1)},\ n'/w_t\right\}} \frac{\Pr\left[Z_1 = z_1\right] \cdot \left(L_{w_t}^{(\mathrm{Pass}\ t)}[z_1] + L_{w_t}^{(\mathrm{Pass}>\, t)}[z_1]\right)}{\sum_{l=0}^{\max\left\{n'p^{(t-1)},\ n'/w_t\right\}} \Pr\left[Z_1 = l\right]}.$$

Therefore, the optimal weight w_t^{opt} for Pass t is determined by

$$w_t^{\mathrm{opt}} = \left\{ w \mid g\left(n', p^{(t-1)}, w\right) = \min_{x\in\mathbb{N},\ 2<x\le n'/2} g\left(n', p^{(t-1)}, x\right) \right\}.$$

The above analysis for the optimal w_t assumes that n' is divided by w_t. This assumption is not very realistic. Let $w' \equiv n' \pmod{w_t}$, then w' is the weight of the last vector of Pass t. Since w' may vary from 0 to $w_t - 1$, it is not wise to neglect the last vector and take $\left\lfloor \frac{n'}{w_t} \right\rfloor$ as the number of vectors (some of them might be split afterwards) created during Pass t, nor is it wise to take it for granted that the last vector behaves like a vector of weight w_t, and take $\left\lceil \frac{n'}{w_t} \right\rceil$ as the number of vectors created during Pass t. It is better to consider the last vector of weight w' independently.

The probability that an error may be corrected in the last vector of Pass t is determined by

$$\mathrm{Pe} = \frac{1 - \left(1 - 2p^{(t-1)}\right)^{w'}}{2}.$$

The probability that z_1 errors are removed during Pass t is therefore given by

$$\mathrm{Qr}[z_1] = \begin{cases} (1-\mathrm{Pe}) \cdot \Pr\left[z_1\right] & z_1 = 0 \\ \mathrm{Pe} \cdot \Pr\left[z_1 - 1\right] + (1-\mathrm{Pe}) \cdot \Pr\left[z_1\right] & z_1 = 1, 2, \ldots, \left\lfloor \frac{n'}{w_t} \right\rfloor \\ \mathrm{Pe} \cdot \Pr\left[z_1 - 1\right] & z_1 = \left\lfloor \frac{n'}{w_t} \right\rfloor + 1, \text{ if } w' \neq 0 \end{cases}. \quad (2.45)$$

Here $\Pr\left[z_1\right]$ is determined by (2.39).

Equation (2.42) should also be modified correspondingly to include the effect of w'. We will show the modification later in Algorithm 2.5.2, which describes how to determine w_t^{opt}.

Let $w_l = \left\lfloor \frac{w}{2} \right\rfloor$ and $w_r = \left\lceil \frac{w}{2} \right\rceil$. Some functions are defined first.

- $f(x,p) = \log(x) + \frac{2}{1-(1-2p)^x}$, where $x \in \mathbb{N}$ and $0 < p < 0.5$.
- $w^{\mathrm{opt}} = \min\left\{\max\left\{2, \{w | f(w,p) = \min_{x\in\mathbb{N}} f(x,p)\}\right\}, \frac{n}{2}\right\}$, the optimal weight determined by the bit error rate p and the length n of Bob's working string.

- $$\mathrm{SPLIT}(w) = 1 + \frac{w_l}{w} \cdot \mathrm{SPLIT}(w_l) + \frac{w_r}{w} \cdot \mathrm{SPLIT}(w_r)$$

  with initial value $\mathrm{SPLIT}(1) = 0$. This function is used to evaluate the number of splittings in a binary search to find an error in a vector of weight w. It is more precise than $\log(w)$ since a splitting always results in 2 integers.

- $$\mathrm{NUM}(w) = \frac{w_l}{w} \cdot [(w_r - 1) + \mathrm{NUM}(w_l)] + \frac{w_r}{w} \cdot [(w_l - 1) + \mathrm{NUM}(w_r)]$$

  with initial value $\mathrm{NUM}(1) = 0$. It counts the number of remaining bits after a vector of weight w is split to locate an error and the corresponding bits are discarded.

- $$\mathrm{BER}\,(p, w) = \frac{p \cdot \frac{1-(1-2p)^{w-1}}{2}}{p \cdot \frac{1-(1-2p)^{w-1}}{2} + (1-p) \cdot \frac{1+(1-2p)^{w-1}}{2}}.$$

  It estimates the a posteriori error probability for each bit in a vector of weight w, given that the a priori error probability of each bit is p and the corresponding syndrome for this vector is 0.

- $$\begin{aligned}\mathrm{PROB}\,(w, p) =& \frac{w_l}{w} \cdot [(w_r - 1) \cdot \mathrm{BER}\,(p, w_r) + \mathrm{PROB}\,(w_l, p)] \\ &+ \frac{w_r}{w} \cdot [(w_l - 1) \cdot \mathrm{BER}\,(p, w_l) + \mathrm{PROB}\,(w_r, p)]\end{aligned}$$

  with initial value $\mathrm{PROB}\,(1, p) = 0$. The sum of the bit error probabilities of the remaining bits, after a vector of weight w is split to locate an error and the corresponding bits are discarded.

- $$\begin{aligned}\mathrm{DISC}\,(w, p) =& \frac{w_l}{w} \cdot [\mathrm{BER}\,(p, w_r) + \mathrm{DISC}\,(w_l, p)] \\ &+ \frac{w_r}{w} \cdot [\mathrm{BER}\,(p, w_l) + \mathrm{DISC}\,(w_r, p)]\end{aligned}$$

  with initial value $\mathrm{DISC}\,(1, p) = 0$. The sum of the error probabilities of the discarded bits, with explicit bits excluded, after a vector of weight w has been split. If we include the explicit bits in the sum, it should be $1 + \mathrm{DISC}\,(w, p)$.

The notation used in Algorithm 2.5.2 is given below.

- n': the length of Bob's working string after Pass $t-1$, i.e., the number of bits to be involved in Pass t.

- $p^{(t-1)}$: Bob's average bit error probability after Pass $t-1$.
- n'': the length of Bob's working string after Pass t, i.e., the number of bits to be involved in Pass $t+1$.
- w_t: the weight of vectors used during Pass t.
- z_1: the number of errors corrected in Pass t.
- $\mathrm{Qr}[z_1]$: the probability that z_1 errors are removed during Pass t.
- z_2: a counter recording the number of errors corrected from Pass $t+1$ onwards.
- z': the number of errors corrected during Pass $t+1$.
- $p_{[z_2]}$: Bob's average bit error probability when z_2 errors have been corrected since Pass $t+1$.
- $w_{t+1}^{\mathrm{opt}}[z_2]$: the optimal weight of Pass $t+1$ when z_2 errors have been removed since Pass $t+1$.
- $w_i^{\mathrm{opt}},\ i > t+1$: the optimal weight of a new vector created during BICONF Pass i.
- $L_{w_t}^{(\mathrm{Pass}\ t)}[z_1]$: the number of bits discarded during Pass t given that z_1 errors are eliminated in Pass t.
- $L_{w_t}^{(\mathrm{Pass}>t)}[z_1]$: the number of bits discarded from Pass $t+1$ onwards until all remaining errors are corrected, given that z_1 errors are eliminated in Pass t.
- $g\left(n', p^{(t-1)}, w_t\right)$: the number of bits discarded from Pass t onwards until all remaining errors are corrected, given that the two working strings are of length n' and with bit error probability $p^{(t-1)}$, and w_t is chosen as the weight of vectors in Pass t.

The algorithm to determine the optimal weight w_t^{opt} is described as follows.

Algorithm 2.5.2

(1) **Determine** $\mathrm{Qr}[z_1]$, $z_1 = 0, 1, \ldots, \left\lceil \frac{n'}{w_t} \right\rceil$.

(a) $P_{\mathrm{odd}} = \frac{1-\left(1-2p^{(t-1)}\right)^{w_t}}{2}$;

$Pr[Z_1 = z_1] = \binom{\left\lfloor \frac{n'}{w_t} \right\rfloor}{z_1} {P_{\mathrm{odd}}}^{z_1} (1 - P_{\mathrm{odd}})^{\left\lfloor \frac{n'}{w_t} \right\rfloor - z_1}$, *for* $z_1 = 0, 1, \ldots, \left\lfloor \frac{n'}{w_t} \right\rfloor$.

(b) $w' \equiv n' \pmod{w_t}$; $\mathrm{Pe} = \frac{1-\left(1-2p^{(t-1)}\right)^{w'}}{2}$.

(c) $$\mathrm{Qr}[z_1] = \begin{cases} (1-\mathrm{Pe})\cdot Pr[z_1] & z_1 = 0 \\ \mathrm{Pe}\cdot Pr[z_1-1] + (1-\mathrm{Pe})\cdot Pr[z_1]) & z_1 = 1,2,\ldots,\left\lfloor \frac{n'}{w_t} \right\rfloor \\ \mathrm{Pe}\cdot Pr[z_1-1] & z_1 = \left\lfloor \frac{n'}{w_t} \right\rfloor + 1, \text{ and } w' \neq 0 \end{cases}$$

(2) $\mathrm{maxZ}_1 = \min\left\{n'p^{(t-1)}, \left\lceil \frac{n'}{w_t} \right\rceil\right\}$; $\mathrm{Q}_{\mathrm{sum}} = \sum_{z_1=0}^{\mathrm{maxZ}_1} \mathrm{Qr}[z_1]$.

For $(z_1 = 0;\ z_1 \leq \mathrm{maxZ}_1;\ z_1 \leftarrow z_1 + 1)$

{

(a) **If** $(w' = 0)${

$$L_{w_t}^{(\text{Pass } t)}[z_1] = \frac{n'}{w_t} + z_1 \cdot \mathrm{SPLIT}(w_t).$$

}

else{

$$L_{w_t}^{(\text{Pass } t)}[z_1] = \left\lceil \frac{n'}{w_t} \right\rceil + (z_1 - 1)\cdot\mathrm{SPLIT}(w_t) + (1-\mathrm{Pe})\cdot\mathrm{SPLIT}(w_t) + \mathrm{Pe}\cdot\mathrm{SPLIT}(w').$$

}

(b) **If** $(w' = 0)${

$$p^{(t)} = \frac{\left(\frac{n'}{w_t} - z_1\right)\cdot(w_t - 1)\cdot\mathrm{BER}\left(p^{(t-1)}, w_t\right) + z_1\cdot\mathrm{PROB}\left(w_t, p^{(t-1)}\right)}{\left(\frac{n'}{w_t} - z_1\right)\cdot(w_t - 1) + z_1\cdot\mathrm{NUM}\left(w_t\right)}.$$

}

else{

$$\begin{aligned}
c_1 &= (1-\mathrm{Pe})\cdot\left[\left(\left\lfloor\frac{n'}{w_t}\right\rfloor - z_1\right)\cdot(w_t-1)\cdot\mathrm{BER}\left(p^{(t-1)}, w_t\right)\right] \\
&+ (1-\mathrm{Pe})\cdot\left[(w'-1)\cdot\mathrm{BER}\left(p^{(t-1)}, w'\right)\right] \\
&+ (1-\mathrm{Pe})\cdot\left[z_1\cdot\mathrm{PROB}\left(w_t, p^{(t-1)}\right)\right]; \\
c_2 &= \mathrm{Pe}\cdot\left[\left(\left\lfloor\frac{n'}{w_t}\right\rfloor - z_1 + 1\right)\cdot(w_t-1)\cdot\mathrm{BER}\left(p^{(t-1)}, w_t\right)\right] \\
&+ \mathrm{Pe}\cdot\left[(z_1-1)\cdot\mathrm{PROB}\left(w_t, p^{(t-1)}\right) + \mathrm{PROB}\left(w', p^{(t-1)}\right)\right]; \\
d_1 &= (1-\mathrm{Pe})\cdot\left[\left(\left\lfloor\frac{n'}{w_t}\right\rfloor - z_1\right)\cdot(w_t-1) + z_1\cdot\mathrm{NUM}\left(w_t\right) + (w'-1)\right]; \\
d_2 &= \mathrm{Pe}\cdot\left(\left\lfloor\frac{n'}{w_t}\right\rfloor - z_1 + 1\right)\cdot(w_t-1) \\
&+ \mathrm{Pe}\cdot\left[(z_1-1)\cdot\mathrm{NUM}\left(w_t\right) + \mathrm{NUM}(w')\right]; \\
p^{(t)} &= \frac{c_1 + c_2}{d_1 + d_2}.
\end{aligned}$$

}

(c) **Determine** $p_{[z_2]}$.

$$\mathrm{Rem}_{\mathrm{err}} = n' \cdot p^{(t-1)} - \left\lfloor \frac{n'}{w_t} \right\rfloor \cdot \mathrm{BER}\left(p^{(t-1)}, w_t\right) - z_1 \cdot \left[1 + \mathrm{DISC}\left(w_t, p^{(t)}\right)\right]; \tag{2.46}$$

if $(w' \neq 0)\{$

$$\mathrm{Rem}_{\mathrm{err}} \leftarrow \mathrm{Rem}_{\mathrm{err}} - (1 - \mathrm{Pe}) \cdot \mathrm{BER}\left(p^{(t-1)}, w'\right) - \mathrm{Pe} \cdot \mathrm{DISC}\left(w', p^{(t)}\right);$$

}

$$\begin{aligned}
n'' &= n' - L_{w_t}^{(\mathrm{Pass}\ t)}[z_1]; \\
Pr[l] &= \binom{n''}{l} \left(p^{(t)}\right)^l \left(1 - p^{(t)}\right)^{n''-l}; \\
Pr[l| \geq z_2] &= \frac{Pr[l]}{\sum_{j \geq z_2} Pr[j]}; \\
p_{[z_2]} &= \sum_{l \geq z_2} Pr[l| \geq z_2] \, \frac{l - z_2}{n'' - 2z_2}, \quad z_2 = 0, 1, \ldots, \mathrm{Rem}_{\mathrm{err}}.
\end{aligned}$$

(d) **Determine** $L_{w_t}^{(\mathrm{Pass}>t)}[z_1]$.

$z_2 = 0$; $U = 0$; $L_{w_t}^{(\mathrm{Pass}>t)}[z_1] = 0$; $i = t + 2$;

While $(z_2 \leq \mathrm{Rem}_{\mathrm{err}})$ {

If $\left(L_{w_t}^{(\mathrm{Pass}>t)}[z_1] \geq n''\right)$ **break**;

If $(U < n'')$ {

$$\begin{aligned}
w_{t+1}^{\mathrm{opt}}[z_2] &= \min\left\{\max\left\{2, \left\{w | f(w, p_{[z_2]}) = \min_{x \in \mathbb{N}} f(x, p_{[z_2]})\right\}\right\}, \frac{n''}{2}\right\}; \\
P_{\mathrm{odd}}[z_2] &= \frac{1 - \left(1 - 2p_{[z_2]}\right)^{w_{t+1}^{\mathrm{opt}}[z_2]}}{2}; \\
D &= \left(\frac{1}{P_{\mathrm{odd}}[z_2]} + \mathrm{SPLIT}\left(w_{t+1}^{\mathrm{opt}}[z_2]\right) + \mathrm{SPLIT}(w_t)\right); \\
L_{w_t}^{(\mathrm{Pass}>t)}[z_1] &\leftarrow L_{w_t}^{(\mathrm{Pass}>t)}[z_1] + D; \\
U &\leftarrow U + \frac{w_{t+1}^{\mathrm{opt}}[z_2]}{P_{\mathrm{odd}}[z_2]}; \\
\mathrm{Rem}_{\mathrm{err}} &\leftarrow \mathrm{Rem}_{\mathrm{err}} - (D - 4) \cdot p_{[z_2]}; \\
z_2 &\leftarrow z_2 + 2; \\
z' &= z_2;
\end{aligned} \tag{2.47}$$

}

else {

$$
\begin{aligned}
w_i^{\text{opt}} &= \frac{n'' - L_{w_t}^{(\text{Pass}>t)}[z_1]}{2}; \\
D &= 1 + \frac{1}{2}\text{SPLIT}(w_t) + \frac{1}{2}\frac{\sum_{l=0}^{z'} \left(\text{SPLIT}\left(w_{t+1}^{\text{opt}}[l]\right)/P_{\text{odd}}[l]\right)}{\sum_{l=0}^{z'} 1/P_{\text{odd}}[l]} \\
&+ \frac{1}{2}\sum_{l=t+2}^{i+2} \text{SPLIT}\big(w_l^{\text{opt}}\big); \\
L_{w_t}^{(\text{Pass}>t)}[z_1] &\leftarrow L_{w_t}^{(\text{Pass}>t)}[z_1] + D; \\
\text{Rem}_{\text{err}} &\leftarrow \text{Rem}_{\text{err}} - [D - 2i] \cdot p_{[z_2]}; \\
z_2 &\leftarrow z_2 + i; \\
i &\leftarrow i + 1;
\end{aligned}
$$

} } }

(3) **Determine** $g\left(n', p^{(t-1)}, w_t\right)$.

$$
g\left(n', p^{(t-1)}, w_t\right) = \sum_{z_1=0}^{\text{maxZ}_1} \frac{\text{Qr}[z_1]}{\text{Q}_{\text{sum}}} \cdot \left(L_{w_t}^{(\text{Pass } t)}[z_1] + L_{w_t}^{(\text{Pass}>t)}[z_1]\right) \tag{2.48}
$$

(4) **Determine the optimal weight for Pass** t

$$
w_t^{\text{opt}} = \left\{ w \mid g\left(n', p^{(t-1)}, w\right) = \min_{x\in\mathbb{N},\ 2<x\le n'/2} g\left(n', p^{(t-1)}, x\right) \right\}. \tag{2.49}
$$

Through the algorithm, we assume that the total number of errors in Bob's original string is $n'p^{(t-1)}$. Hence the computational overload is of order about $n'^2p^{(t-1)}/2$ and memory overload about n'. We may consider the number of errors in Bob's original n'-bit string as a random variable, say Z. The probability that z errors exist in Bob's n'-bit string is

$$
\Pr\left[Z = z\right] = \binom{n'}{z}\left(p^{(t-1)}\right)^z\left(1 - p^{(t-1)}\right)^{n'-z},
$$

where $z = 0, 1, \ldots, n'$.

All places where $n'p^{(t-1)}$ appears in the above algorithm are replaced by z. Therefore,

$$
\text{maxZ}_1 = \min\left\{z, \left\lceil \frac{n'}{w_t} \right\rceil\right\},
$$

and

$$
\text{Rem}_{\text{err}} = z - \left\lfloor \frac{n'}{w_t} \right\rfloor \cdot \text{BER}\left(p^{(t-1)}, w_t\right) - z_1 \cdot \left[1 + \text{DISC}\left(w, p^{(t)}\right)\right]
$$

should replace (2.46). Equation (2.48) is replaced by

$$g\left(n', p^{(t-1)}, w_t\right) = \sum_{z=0}^{n'} \Pr\left[Z = z\right] \sum_{z_1=0}^{\max Z_1} \frac{\mathrm{Qr}[z_1]}{\mathrm{Q}_{\mathrm{sum}}} \cdot \left(L_{w_t}^{(\mathrm{Pass}\ t)}[z_1] + L_{w_t}^{(\mathrm{Pass}>t)}[z_1]\right).$$

However, this way will increase the complexity of the algorithm up to n'^2.

2.5.4 Simulation Results

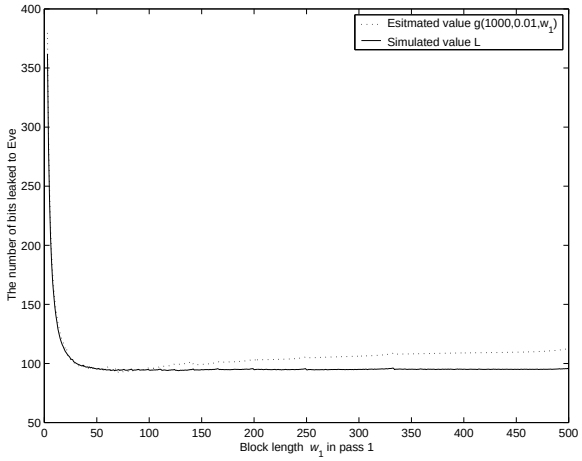

Figure 2.6: Estimated and simulated values of the amount of leaked bits as a function of the weight of vectors w_1

Let $n = 1000$, $p = 0.01$, and w_1 vary from 3 to 500. Then $t = 1$ and $n = n'$ since $p < 0.20$. With Algorithm 2.5.2, we can estimate the amount of leaked information $g\left(n, p, w_1\right)$ for different values of w_1. The value of $g\left(n, p, w_1\right)$ as a function of w_1 is shown by the dotted line in Figure 2.6. The solid line shows the corresponding simulation results of the amount of bits, denoted by L, leaked to Eve in our protocol. We stopped our protocol when 10 successive BICONF passes detected no error. It turns out that no error is left after our protocol stops. Therefore, the gap between L and $g\left(n, p, w_1\right)$ should approximately be 10 if $g\left(n, p, w_1\right)$ is properly estimated. In Figure 2.6, however, $L - g\left(n, p, w_1\right) \leq 10$. We see the following two explanations. Errors are corrected by pairs in Pass 2, one in some vector of Pass 2, the other in some vector of Pass 1. We assume that the number of splittings in a vector of Pass 1 is $\log(w_1)$. However, the vector might have already been split during Pass 1, so the number of further splittings is less than $\log(w_1)$. The other reason is given by our pessimistic estimation of the bit error rate from Pass 2 onwards and our conservative estimate of number of errors which disappeared due to discarding them rather than correcting them (see (2.43) and (2.44)).

The minimal value 93.871 of L is achieved at $w_1 = 67$ according to the simulation results. We do observe that L does not change much in value when $50 \leq w_1 \leq 500$. Our estimated optimal value $w_1 = 72$ comes up with $L = 94.08$, which is very close to the minimal value 93.871. The simulated value of the information rate of our protocol is given by $R_0(0.01) = 1 - L/n = 1 - 94.08/1000 = 0.9059$ when $w_1 = 72$.

In Table 2.3 we compare values of w_1^{opt} obtained by (2.49) for $p = 0.01$, 0.05, 0.10, 0.15 and $n = 1000$, 10000, with those used in $\texttt{Cascade}^{opt}$. The $\texttt{Cascade}^{opt}$ protocol also stops after 10 successive BICONF passes detect no error. The rule for determining optimal weights in $\texttt{Cascade}^{opt}$ is independent of the length of the working string. But in our protocol the string length also has some influence on the optimal weights. Since the optimal block lengths in Pass 2 in our protocol are varying, we determine the average value $\overline{w}_2^{\text{opt}}[z_2]$ of the simulation results and give the range for the optimal value of w_2, namely $\left(\overline{w}_2^{\text{opt}}[0] \ \sim \ \overline{w}_2^{\text{opt}}[z']\right)$, where z' is the number of errors corrected during Pass 2.

protocol	$p = 0.01$	$p = 0.05$	$p = 0.10$	$p = 0.15$
$\texttt{Cascade}^{opt}$ for w_1	70	14	7	4
$\texttt{Cascade}^{opt}$ for w_2	301	61	31	25
Ours for w_1 and $n = 1000$	72	16	8	6
Ours for w_2 and $n = 1000$	$128 \sim 293$	$22 \sim 126$	$10 \sim 76$	$8 \sim 55$
Ours for w_1 and $n = 10000$	64	16	8	5
Ours for w_2 and $n = 10000$	$136 \sim 939$	$22 \sim 268$	$10 \sim 129$	$6 \sim 102$

Table 2.3: Optimal weights of the first two passes in $\texttt{Cascade}^{opt}$ and our protocol

In Table 2.4, we show our simulation results of the information rate $R_0(p)$ for our protocol and the $\texttt{Cascade}^{opt}$ protocol for $n = 1000$ and $n = 10000$ as the length of the working strings.

protocol	$R_0(0.01)$	$R_0(0.05)$	$R_0(0.10)$	$R_0(0.15)$
$\texttt{Cascade}^{\text{opt}}$ $n = 1000$	0.904614	0.686502	0.486143	0.328317
$\texttt{Cascade}^{\text{opt}}$ $n = 10000$	0.915799	0.699131	0.499266	0.341535
Ours $n = 1000$	0.905915	0.693869	0.505117	0.352883
Ours $n = 10000$	0.916442	0.705208	0.516097	0.364690
Upper bound for $R_0(p)$	0.919200	0.713600	0.531100	0.390200

Table 2.4: Information rates of $\texttt{Cascade}^{opt}$ and our protocol

It is easy to see that the performance of our protocol is better than $\texttt{Cascade}^{opt}$ especially for large n and p.

2.5.5 Why Discarding Bits from Pass to Pass

We have two choices to execute our protocol. First, we can discard $k^{(i)}$ bits (each is part of a parity check set) during Pass i. These bits will not play a role in the next passes and they will not contribute to the final reconciled strings. The other choice is to keep those bits, let them take part in the next passes and let them contribute to the final reconciled strings (in fact, this is what `Cascade` and `Cascade`^{opt} did). The first strategy shrinks strings from pass to pass, while the second keeps the length of strings unchanged. The problem now is to find out which strategy is better.

To get some idea, we first discuss a simple case. Suppose that a parity check set with syndrome 1 contains only two bits. Then a binary search will split the two bits into two individual bits, one in error and the other correct. It is obvious that it is not wise to let these two bits take part in next passes since they are known exactly by Alice, Bob, and Eve, and will not play a role in the protocol. The following theorem considers a general case. It shows that if we discard a bit from each parity check set in the protocol, we do not change the marginal probability distribution of the error pattern given the syndrome vector.

Theorem 2.5.3 *Let $H_{k\times n}, k \leq n$, be a rank k matrix. Let $\underline{E} = (E_1, E_2, \ldots, E_n)$ denote a row vector of n random variables each defined on $\{0,1\}$ and let $\underline{S}^T = H\underline{E}^T$. So, $\underline{S}$ is a random variable defined on $\{0,1\}^k$.*

For any $\underline{e} \in \{0,1\}^n$ and for any $k \times k$ non-singular submatrix H' of H with columns indexed by $1 \leq j_1 < j_2 < \cdots < j_k \leq n$

$$Pr\left[\underline{E} = \underline{e} \mid \underline{S}^T = H\underline{e}^T\right] = Pr\left[\widetilde{\underline{E}} = \widetilde{\underline{e}} \mid \underline{S}^T = H\underline{e}^T\right],$$

where $\widetilde{\underline{E}}$ and $\widetilde{\underline{e}}$ are $n-k$ dimensional vectors formed from $\underline{E}$ resp. $\underline{e}$ by removing the coordinates $j_1, j_2, \ldots, j_k$.

Proof: Since H' is non-singular, there exist non-singular matrices $P_{k\times k}$ and $Q_{n\times n}$ such that $HQ = (H'||H'')$ and $PHQ = (I||A)$, where $||$ denotes a concatenation. Let $\underline{S'}^T = P\underline{S}^T$, $\underline{E'}^T = Q^{-1}\underline{E}^T = \left(\underline{E_1}'||\underline{E_2}'\right)^T$. We know that $(I||A)\,\underline{E'}^T = (I||A)\left(\underline{E_1}'||\underline{E_2}'\right)^T = \underline{S'}^T$, i.e., $\underline{E_1}'^T + A\underline{E_2}'^T = \underline{S'}^T$. Hence

$$\begin{aligned}
&\Pr\left[\underline{E} = \underline{e} \mid \underline{S}^T = H\underline{e}^T\right] = \Pr\left[Q^{-1}\underline{E}^T = Q^{-1}\underline{e}^T \mid P\underline{S}^T = PHQQ^{-1}\underline{e}^T\right] \\
&= \Pr\left[\underline{E'} = \underline{e'} \mid \underline{S'}^T = (I||A)\,\underline{e'}^T\right] = \Pr\left[(\underline{E_1}'||\underline{E_2}') = (\underline{e_1}'||\underline{e_2}') \mid \underline{S'}^T = (I||A)\left(\underline{e_1}'||\underline{e_2}'\right)^T\right] \\
&= \Pr\left[\underline{E_1}' = \underline{e_1}', \underline{E_2}' = \underline{e_2}' \mid \underline{S'}^T = \underline{e_1}'^T + A\underline{e_2}'^T\right] \\
&= \Pr\left[\underline{E_2}' = \underline{e_2}' \mid \underline{S'}^T = \underline{e_1}'^T + A\underline{e_2}'^T\right] \cdot \Pr\left[\underline{E_1}' = \underline{e_1}' \mid \underline{E_2}' = \underline{e_2}', \underline{S'}^T = \underline{e_1}'^T + A\underline{e_2}'^T\right] \\
&= \Pr\left[\underline{E_2}' = \underline{e_2}' \mid \underline{S'}^T = \underline{e_1}'^T + A\underline{e_2}'^T\right] = \Pr\left[\underline{E_2}' = \underline{e_2}' \mid \underline{S}^T = H\underline{e}^T\right] \\
&= \Pr\left[\widetilde{\underline{E}} = \widetilde{\underline{e}} \mid \underline{S}^T = H\underline{e}^T\right].
\end{aligned}$$

In the last step but two, we use the fact that $\underline{E_1}'^T = \underline{S'}^T - A\underline{E_2}'^T$, which implies that

$$\Pr\left[\underline{E_1}' = \underline{e_1}' \mid \underline{E_2}' = \underline{e_2}', \underline{S'}^T = \underline{e_1}'^T + A\underline{e_2}'^T\right] = 1.$$

□

The above theorem shows that if we choose the proper k bits to discard it makes no difference for Bob's marginal probability distribution of the error pattern given the syndrome vector and the parity matrix H. In our protocol, from the index set of each parity check vector $\underline{h}_m$, we discard its first bit $(\underline{h}_m)_{l_m}$, so $(\underline{h}_r)_{l_m} = 0$ for $r > m$. If we collect the k column vectors obtained in this way, we get a $k \times k$ matrix,

$$H'_{k\times k} = \begin{pmatrix} 1 & * & \dots & * \\ 0 & 1 & \dots & * \\ \vdots & \vdots & \dots & * \\ 0 & 0 & \dots & 1 \end{pmatrix},$$

which is obviously of full rank. According to Theorem 2.5.3, Bob's marginal probability distribution of the error pattern remains unchanged. On the other hand, if we discard bits from each pass to the next, in the way explained above, the protocol is more efficient since the strings shrink and the discarding bits may contain errors.

2.5.6 Eve's Strategy

At the beginning of our protocol, Eve may have more information about Alice's string than Bob has. However, with the protocol being executed, Bob gains more and more advantage over Eve and finally he arrives at a common string with Alice, while still leaving Eve with some uncertainty about it. The question is what Eve can do with the parity check information exchanged by Alice and Bob over the public channel.

Eve's a priori bit error probability is $p' = p_A + p_E - 2p_Ap_E$ in the satellite scenario. Since she knows H, she can construct the belief network just as Bob can. She also knows her syndrome vector $\underline{s}'^T = H\underline{e}'^T$ and Bob's syndrome vector $\underline{s}^T = H\underline{e}^T$. Here $\underline{e}'$ and $\underline{e}$ are the error patterns between Alice's and Eve's strings and between Alice's and Bob's.

Since the weight of vectors is 2 up to Pass $t-1$, Eve knows that Bob's error pattern for a particular vector is $(0,0)$ or $(1,1)$ if Bob's corresponding syndrome is 0. On the other hand, the belief network has no loop. As shown in Subsection 2.5.2, our protocol in the first $t-1$ passes is just the same as a $[2,\ t-1]$ iteration protocol. So Eve can use her syndrome information $\underline{s}'$ and Bob's $\underline{s}$ to compute her marginal, a posteriori probabilities on the individual bits according to (2.9). Let these probabilities be denoted by $p'^{(t-1)}_l, l = 1, 2, \dots, n$. Eve does the corresponding error correction which results in the optimal strategy for her.

In Pass t, there is still no loop in the belief network, but the weight w_t is larger than 2. Bob's error patterns become elusive and it is harder for Eve to include all possible error patterns of Bob in her error correction.

After Pass t, say Pass i with $i \geq t+1$, the topology of the belief network contains loops. The computation of the a posteriori probabilities has been shown to be intractable in [61] for belief networks that correspond to the problem $H\underline{e}'^T = \underline{s}'^T$ and that contain loops. However, as [40] shows, Eve's aim is to decode $\underline{e}'$ from $H\underline{e}'^T = \underline{s}'^T$, and the a posteriori probabilities are not required for the correct decoding. Neglecting Bob's syndrome information, Eve can indeed run the BPD $(\underline{p}'^{(t-1)}, H', \underline{s}')$ algorithm for Pass $i, i \geq t$, in the following way, where

$$H' = \left(H^{(t)^T}, H^{(t+1)^T}, \ldots, H^{(i)^T}\right)^T$$

is the parity check matrix constructed from Pass t to i, and

$$\underline{p}'^{(t-1)} = (p'^{(t-1)}_1, p'^{(t-1)}_2, \ldots, p'^{(t-1)}_n)$$

is the a priori error probabilities for Eve's bits.

Initialization. For every (l, m) such that $H_{ml} = 1$, let $q^0_{ml} = 1 - p'^{(t-1)}_l$ and $q^1_{ml} = p'^{(t-1)}_l$.

Horizontal Step. Compute r^0_{ml} and r^1_{ml} with (2.30) and (2.31).

Vertical Step. Update q^0_{ml} and q^1_{ml} with (2.32) and (2.33).

The algorithm iterates between the horizontal step and the vertical step. For each iteration, compute the "pseudo-posterior probabilities"

$$q^0_l = \alpha_l \cdot \left(1 - p'^{(t-1)}_l\right) \cdot \prod_{m \in \mathcal{M}(l)} r^0_{ml},$$

$$q^1_l = \alpha_l \cdot p'^{(t-1)}_l \cdot \prod_{m \in \mathcal{M}(l)} r^1_{ml},$$

where the scalar α_l is chosen such that $q^0_l + q^1_l = 1$. If $q^1_l > 0.5$, set $e'_l = 1$ otherwise $e'_l = 0$. Check if $H'\underline{e}'^T = \underline{s}'^T$ is satisfied. Stop when it is but also stop when the number of the iteration reaches a threshold (for example 30).

It should be noted that Eve's strategy to decoding is not optimal because the belief network contains loops and Eve does not use Bob's syndrome information $\underline{s}^T = H\underline{e}^T$. However, since H is usually a very sparse matrix, the performance of the BPD-algorithm is much better (see [40]) than if the parity check matrix of a standard error correction code such as the BCH code or RM code is used. For example, when $p_A = p_B = 0.02566$, $p_E = 0.01512$, we get $p = p_A + p_B - 2p_Ap_B = 0.05$, and $p' = p_A + p_E - 2p_Ap_E = 0.04$. We show the simulation results for Eve compared to Bob in Figure 2.7 for the case that the probability that Bob's working string is not error free is approximately 10^{-5}.

Suppose that β is the expected bit error rate of Bob's reconciled string after discarding k bits in our protocol. Then the information rate $R_\beta(p)$ is given by $R_\beta(p) = (n-k)/n$. According to Shannon's limit theorem, $nh(p) \leq k + (n-k)h(\beta)$

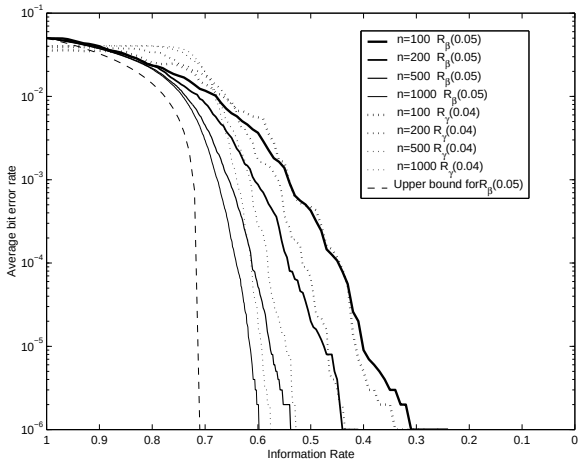

Figure 2.7: Bob's and Eve's average output bit error rate versus the information rate for $n = 100, 200, 500, 1000$, $p = 0.05$ and $p' = 0.04$.

which implies that $R_\beta(p) \leq \frac{1-h(p)}{1-h(\beta)}$. This is the upper bound on Bob's information rate.

We also compute Eve's information rate $R_\gamma(p')$ with γ the value of expected output bit error rate of Eve. From Figure 2.7, we see that Bob's information rate approaches the lower bound for increasing values of n. Eve's a priori bit error probability $p' = 0.04$ is smaller than Bob's $p = 0.05$, but Eve approaches her limit slower than Bob for increasing n. When n is large enough ($n \geq 500$ in our simulation), Bob's average a posteriori bit error rate β could be smaller than Eve's γ.

2.6 Relationship with Other Protocols

In this section, we show the connections between our protocol and other known protocols. We present the corresponding simulation results.

2.6.1 The Bit Pair Iteration Protocol

When the optimal weight (block length) determined by the bit error probability is 2, our general protocol reduces to the bit pair iteration protocol as described in Section 2.2.3. Bob can determine the a posteriori probabilities for each of his bits with the BPD-algorithm, just as we did in the analysis of the bit pair iteration protocol in Section 2.2.3. On the other hand, when the bit error probability p between Alice's and Bob's strings is smaller than 0.20, the optimal weight is larger than 2 and the bit pair iteration protocol is not the best choice any more.

For the example of Subsection 2.5.6, our protocol can achieve a much larger information rate than the bit pair iteration protocol, as illustrated in Table 2.5.

protocol	β	$R_\beta(0.05)$
bit pair iteration protocol (1 round)	0.002763	0.500
our protocol n =1000	0.002763	0.699
bit pair iteration protocol (2 round)	0.000008	0.249
our protocol n =1000	0.000008	0.613

Table 2.5: Comparison of our protocol and the bit pair iteration protocol

2.6.2 The BICONF Primitive

In our protocol, after Pass $t+1$, the average bit error rate $p^{(t+1)}$ (according to simulation) is so small that BICONF passes are enough to eliminate the remaining errors. The difference between BICONF pass and BICONF primitives is that if an error is found in some BICONF pass, other errors may be located by looking back to the parity check vectors of previous passes.

2.6.3 The Cascade^{opt} Protocol

The connection and difference between the `Cascade`^{opt} protocol and our protocol are shown below.

- The `Cascade`^{opt} protocol functions efficiently only when $p < 0.20$. When $p \geq 0.20$, the number of passes needed to remove most errors in such a reconciliation protocol is larger than 2. Our protocol is more general since it applies to $p < 0.5$. More precisely, there are $t-1$ passes in our protocol to make Bob's bit error probability less than 0.20.

- The first two passes of the `Cascade`^{opt} protocol correspond to Pass t and Pass $t+1$ of our protocol. Our protocol shares with `Cascade`^{opt} almost the same principle for determining the optimal block lengths for the two passes. The principle is to minimize the amount of information leaked during the two passes, based on the fact that almost all errors are removed in these two passes. However, the optimal block length w_{t+1} for Pass $t+1$ in our protocol is chosen dynamically with the current average bit error rate. It increases during Pass $t+1$ rather than staying constant like w_2 does in the `Cascade`^{opt} protocol.

- Our protocol needs more memory and computational overload to estimate the optimal weights in Pass t and Pass $t+1$. The main cost is Algorithm 2.5.2, which has a computational overload of order $n'^2 p^{(t-1)}/2$ and memory overload of order n' to get w_t^{opt}. Recall that the bit error rates before Pass t can be got from Algorithm 2.5.1, the bit error rate after Pass t can be computed by Equation (2.36) and (2.37), and that after Pass t by Equation (2.38). Since all the bit error rates are got with almost no computational and memory overload, the dynamic optimal weight w_{t+1}^{opt} during Pass $t+1$ can be got from Equation (2.28) almost for free.

- The $\texttt{Cascade}^{opt}$ protocol does not discard any bit. In our protocol, bits (one bit from each block) are discarded with passes going on to compensate the syndrome (parity) information leaked to Eve. So, the working strings are shrinking from pass to pass. Discarding bits also helps error correction because errors may also be discarded.

- Simulation results show that our protocol comes up with a better performance (a higher information rate) than the $\texttt{Cascade}^{opt}$ protocol.

2.6.4 Van Dijk and Koppelaar's Protocol

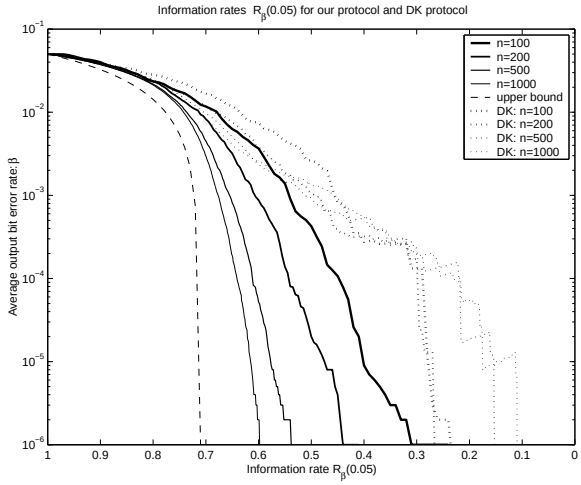

Figure 2.8: The average output bit error rate β versus the information rate $R_\beta(0.05)$ for $n = 100, 200, 500, 1000$ with the DK protocol and our protocol

In [19], Van Dijk and Koppelaar suggest that the parity check matrix H should be constructed row by row. When a new row is created, the average bit error rate is estimated by the number of errors that have been corrected and the optimal weight of the vectors is computed according to (2.28). They propose to use the number of corrected errors to estimate the average bit error rate. However, the information on the number of corrected errors is not sufficient for accurately estimating bit error probabilities.

In our protocol, we also construct H row by row. But we use the BPD-algorithm to calculate the exact a posteriori probabilities of individual bits from the syndrome information $\underline{s}^T = H\underline{e}^T$ in Pass $i, 1 \leq i \leq t$. Only when the BPD-algorithm does not work due to the loops in the belief network defined by H, do we use the number of corrected errors to estimate the bit error rate in Pass $t+1$ as they did in their protocol (which we call the DK protocol). After Pass t, there are only very few errors left, and BICONF passes are enough to eliminate the remaining errors.

The simulation results in Figure 2.8 show that our protocol is much better than the DK protocol. The information rate in the DK protocol even decreases when n is increasing, which is not logical.

2.7 Conclusion

The problem of advantage distillation and information reconciliation can be described as how Alice and Bob can construct an $[n, n-k]$ code with information rate $R = (n-k)/n$ such that Bob can correct all the errors in his codeword, while leaving errors in Eve's string. We presented a protocol to define such a code by constructing a parity check matrix $H_{k\times n}$ row by row from pass to pass with the help of a public discussion between Alice and Bob. Our aim was to get a parity check matrix that leaks as little syndrome information to Eve as possible while correcting as many errors as possible. This gave rise to (2.28), Algorithm 2.5.1, and Algorithm 2.5.2 to compute the optimal block length for each pass.

Suppose that t is the first pass in which the optimal weight (block length) w_t exceeds 2. In passes $i, i < t$, Bob can use the BPD-algorithm introduced in [40] to determine the individual a posteriori bit error probabilities because of the tree shape of the belief network and the sparsity of the parity check matrix. Therefore, the optimal block length for passes $i,\ i \leq t$, can always be determined. But Pass t is an exception to the optimal weight rule defined by (2.28). Since parity check vectors of Pass t are also involved in error corrections in Pass $t+1$ and almost all errors are removed during Pass t and Pass $t+1$, the optimal weight w_t for Pass t should be chosen such that the amount of information exchanged over the public channel to correct all the errors is as small as possible. For passes $i,\ i > t$, we could not determine the exact a posteriori bit error probabilities. However, with the help of the number of errors corrected during Pass $t+1$, a rough estimate of the average bit error rate is determined and used to determine the optimal weight w_{t+1}. It turns out that w_{t+1} is increasing during Pass $t+1$ due to error corrections by parity check vectors of Pass t. For Pass $i,\ i > t+1$, simulation results show that the bit error rate is so small that it is enough to set w_i to half the length of the working string. Therefore, BICONF passes are enough to eliminate the remaining few errors.

We also prove that there is no influence on the marginal probability of Bob's error pattern if Alice and Bob discard one bit from each parity check set.

Simulation results compared our protocol and other known AD/IR protocols. They show that our protocol can achieve a much higher information rate. We also discussed connections between our protocol and other AD/IR protocols. Our protocol turns out to reduce to the bit pair iteration protocol for Pass $i < t$. For Pass t and $t+1$, it proceeds like `Cascade`^{opt} protocol, but more optimized, while for passes i with $i > t+1$ it runs BICONF primitives.

Finally, we conclude that the authenticity of the public channel can be changed to an advantage between Alice and Bob over Eve, in terms of the mutual information between them.

Chapter 3

Privacy Amplification

3.1 Introduction

Privacy amplification is necessary when two parties want to distill a secret key from a large amount of common but partially secret strings. This idea was first proposed by Bennett et al. [4], further studied in [5, 3]. It serves as an important building block in information-theoretic secret key agreement protocols and quantum key agreement protocols.

Suppose that Alice and Bob share an n-bit string S about which the adversary, Eve, has some information, which will be denoted by $Z = z$. The standard way to realize privacy amplification is that Alice randomly chooses a function from a proper class of hash functions and sends the description of this function to Bob over a public channel. Alice and Bob take the hash value of S as their secret key, while Eve's partial uncertainty about S develops to almost full uncertainty about the final secret key.

Like information reconciliation, privacy amplification also involves a public discussion between Alice and Bob. A considerable portion of the methods proposed in the literature makes use of a perfect *authentic* public channel (for instance, the perfection made possible by error correction code techniques). As a result, everybody, including Eve, can learn the entire content of the transmissions over this public channel. On the other hand, the transmissions over the public channel cannot be modified or suppressed by Eve without detection. With such a channel, Eve can only carry out a passive attack. The length r of the final distilled secret key is upper bounded by Eve's Rényi entropy about S, i.e., $r \leq H_2(S|Z = z)$, as we shall show later in this chapter.

The authenticity can be ensured by the physical properties of some channels, for instance newspapers are often proposed in public key cryptosystems and digital signature systems for disclosing public keys. However, using newspapers for a public discussion is too inefficient to be practical.

A more reasonable assumption is that the public channel is *non-authentic*, i.e.,

when Eve can also perform an active attack. Then a problem arises, that is, how to achieve authenticity over this non-authentic public channel. Two approaches to realize privacy amplification over a non-authentic channel have been proposed, each in its own setting.

Approach 1 Alice and Bob share an extra (but shorter) secret key besides the partially secret string S. Alice and Bob use the secret key for unconditionally secure authentication during the privacy amplification process.

Approach 2 Only a partially secret string S is available to Alice and Bob. They use one part of S for unconditionally secure authentication and the other part for distilling a secret.

The first approach is suggested in [3] and is based on the theory of unconditional authentication. A shorter secret key between Alice and Bob is used besides the partially secret string. Like with the one-time pad, the secret key can only be used once to ensure unconditional authenticity for privacy amplification. This privacy amplification may result in a longer secret key, part of which can then be used to replace the used-up short secret during a subsequent authentication. However, it is possible that a malicious opponent may repeatedly interfere with the public discussion between Alice and Bob so that Alice and Bob exhaust their entire supply of authentication keys without obtaining any new secret key. This observation implies that Approach 1 can not appropriately refill the authentication key when Eve performs a continuous active attack.

The second approach is proposed by Maurer et al. in [52]. Authentication is now achieved by sacrificing part of the partially secret string shared by Alice and Bob. Results show that a one-way transmission protocol for privacy amplification against active attacks is possible as long as Eve's min-entropy $H_\infty(S|Z=z)$ (see Definition 1.2.7 in Chapter 1) about S is larger than $2n/3$. Later, Wolf [62] used an interactive protocol to prove that privacy amplification against active attacks with strong robustness (see Definition 3.4.1 in Section 3.4) is possible as long as Eve's Rényi entropy about S, so $H_2(S|Z=z)$, is larger than $2n/3$ and n is large enough.

In this chapter, we study the two approaches, and also present a third approach to the problem of privacy amplification over a non-authentic public channel.

Approach 3 Alice and Bob share two independent partially secret strings, S_I and S_{II}. One is used for authentication and the other is for the distillation of a secret.

The three different approaches apply to different settings. Approach 3 can be viewed as a kind of compromise between the two other approaches. Notice that Approach 3 is not impractical since Alice and Bob can always run information-theoretic secret key agreement protocols [48], each one independent of the others, to obtain independent partially secret common strings.

In the next section, we first review the known results about privacy amplification over an authentic channel. To handle non-authentic channels we need authentication codes to provide authenticity for the channel. Authentication codes (A-codes)

constructed by means of the so-called ϵ-*almost strongly universal*$_2$ class of hash functions are given and the relationship between the cardinality of source states, encoding rules, authenticators, and the probability of Eve's successful attacks are shown in Section 3.3. We shall deal with A-codes under the assumption that the encoding rules are uniformly distributed to Eve, i.e., Eve knows nothing about them, and under the further assumption that the encoding rules are partially secret to Eve. In Section 3.4 we shall show how in different settings of privacy amplification protocols one can use different A-codes to achieve authentication. In the final section, some conclusions will be presented.

This chapter is mainly based on [37].

3.2 Privacy Amplification over an Authentic Channel

After the first two phases in an information-theoretic secret key protocol (see Chapter 1), namely the *advantage distillation* phase and the *information reconciliation* phase, Alice and Bob arrive at a common string. However, Eve has gained some information about it due to the side information exchanged by Alice and Bob over the public channel. As an example, consider a quantum key distribution scheme in which the secret bits are encoded in non-orthogonal states of a quantum system. Eve is prevented from getting complete information about the secret bits due to the uncertainty principle of quantum mechanics, but she does gain some partial information by specific measurements. She also gets more information when Alice and Bob reconcile their quantum bits over a public channel.

Eve's information about Alice's and Bob's common string S can be classified into two types, namely *deterministic information* and *probabilistic information.* Let the random variable Z denote Eve's information about S.

- In the deterministic case, we shall say that Eve knows t bits determined from the n-bit string S. So, $Z = e(S)$, where $e : \{0,1\}^n \to \{0,1\}^t$ is some given function.
- In the probabilistic case Eve knows $H(S|Z)$ Shannon bits about S.

We can see that deterministic information is a specific kind of probabilistic information, therefore probabilistic information is more general to characterize Eve's information.

In the satellite scenario (see Section 1.4 in Chapter 1), Alice, Bob, and Eve receive noisy versions of the same binary, satellite output. Eve gets some probabilistic information about Alice's (or Bob's) string.

In the quantum setting, Eve can perform canonical measurements to some intercepted or split light pulses that have been sent by Alice to Bob. She obtains deterministic bits (physical bits in fact) after the public announcement of the canonical bases since she will definitely get a specific quantum bit if her measurement for it coincides with the right measurement basis. If Eve uses the Breidbart basis to measure the light pulses, she will get the bit with error probability 0.15, so she obtains

probabilistic information in this case. During the advantage distillation or information reconciliation phase, Alice and Bob will use the public channel to exchange some parity check bits about their own strings. These parity check bits are also available to Eve as deterministic bits.

When we say that Eve knows t deterministic bits of the n-bit string S, it may be the case that Eve knows t physical bits of S but Alice and Bob do not know which t positions, or that Eve knows t parity check bits of S, or even that Eve has access to the output of an arbitrary function $e : \{0,1\}^n \to \{0,1\}^t$ of her choice, which is not known to Alice and Bob. In this case, Alice and Bob can distill about $r = n-t$ *highly secret* bits from S, as long as they choose randomly a function g from a universal$_2$ class of hash functions from $\{0,1\}^n$ to $\{0,1\}^r$ and take the hash value $g(S)$ as the final secret string (here "highly secret" means that Eve has negligible information about the finally distilled secret bits) . Such a universal$_2$ class of functions is a set of hash functions from $\{0,1\}^n$ to $\{0,1\}^r$ such that the hash values of two different elements from $\{0,1\}^n$ collides with a probability at most 2^{-r} (see Definition 3.3.3 in Subsection 3.3.1 for more details). The above is summarized in the following theorem by Bennett, Brassard, and Robert [5].

Theorem 3.2.1 *Let S be a random n-bit string with uniform distribution over $\{0,1\}^n$ and let $Z = e(S)$ for an arbitrary eavesdropping function $e : \{0,1\}^n \to \{0,1\}^t$ for some $t < n$. Further, let $s < n - t$ be a positive security parameter and put $r = n - t - s$. Suppose that Alice and Bob choose $S' = G(S)$ as their secret key, where G denotes a random choice from a universal class of hash functions from $\{0,1\}^n$ to $\{0,1\}^r$. Then Eve's expected information about the secret string S', given G and Z, satisfies*

$$I(S'; G, Z) \leq 2^{-s} / \ln 2.$$

In the more general case that Eve gets probabilistic information Z about S, Eve obtains $t = H(S|Z)$ Shannon bits. However, unlike in Theorem 3.2.1, Alice and Bob can not generate $n - t - s$ secret bits. The length of secret string that Alice and Bob can distill from S is measured by Eve's Rényi entropy of order 2, as shown in the following theorem from [3].

Theorem 3.2.2 *Let P_{SZ} be the joint probability distribution of S and Z and let z be the particular value of Z observed by Eve. If Eve's Rényi entropy $H_2(S|Z = z)$ about S is known to be at least c and Alice and Bob choose $S' = G(S)$ as their secret key, where G is chosen at random from a universal class of hash functions from $\mathcal{S}$ to $\{0,1\}^r$, then*

$$H(S|G, Z = z) \geq r - \log\left(1 + 2^{r-c}\right) \geq r - \frac{2^{r-c}}{\ln 2}.$$

It should be pointed out that Theorem 3.2.2 holds only for $H_2(S|Z = z) \geq c$, but can not be generalized to Rényi entropy conditioned on a random variable, i.e.,

$$H(S|G, Z) \geq r - \frac{2^{r-H_2(S|Z)}}{\ln 2}$$

is not true in general when $H_2(S|Z) \geq c$.

We shall now describe how to achieve privacy amplification over an authentic public channel (see also Figure 3.1). The notation $g \in_R \mathcal{G}$ denotes a random choice of an element g from a set $\mathcal{G}$, and $\mathrm{LSB}_r(x)$ stands for the r least significant bits of a string x. Alice's and Bob's initial states are called **reject**.

Let $\cdot$ denote the multiplication of two elements in the finite field $GF(2^n)$.

Alice		Bob
$S \in GF(2^n)$		$S \in GF(2^n)$
$g \in_R GF(2^n)$	$\xrightarrow{\quad g \quad}$	
$S'_A = \mathrm{LSB}_r(g \cdot S)$		$S'_B = \mathrm{LSB}_r(g \cdot S)$
accept		**accept**

Figure 3.1: Protocol for privacy amplification over an *authentic channel*

The class of hash functions $f_g : \{0,1\}^n \to \{0,1\}^r$ with $f_g(s) = \mathrm{LSB}_r(g \cdot s)$ is a universal$_2$ hash function (see Example 3.3.5 in Subsection 3.3.1).

Another way to implement privacy amplification is to use *extractors*. The idea of extractors is to use a small number of truly random bits to distill randomness from some partially secret string. The advantage of an extractor is that the number of truly random bits used for distillation is only an asymptotically small fraction of the total number of partially secret bits. However, the number of final secret bits is measured by the *min*-entropy and extractors exist only theoretically.

Below is a formal definition of an extractor.

Definition 3.2.3 *A function $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^r$ is called a (δ', ϵ')-extractor if for any distribution P on $\{0,1\}^n$ with min-entropy $H_\infty(P) \geq \delta' n$, the variational distance of the distribution of $[V, E(X,V)]$ to the uniform distribution over $\{0,1\}^{d+r}$ is at most ϵ' when choosing X according to P and V independently according to the uniform distribution over $\{0,1\}^d$.*

It was shown in [78] that for every choice of n, $0 < \delta' < 1$, and $\epsilon' > 0$, there exists a (δ', ϵ')-extractor

$$E : \{0,1\}^n \times \{0,1\}^{O((\log(n/\epsilon'))^2 \log(\delta' n))} \to \{0,1\}^{\delta' n - 2\log(1/\epsilon') - O(1)}.$$

The following theorem was proved in [64] as a consequence.

Theorem 3.2.4 *Let $\delta', \triangle_1$, and $\triangle_2 > 0$ be constants. Then for all sufficiently large n, a function*

$$E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^r,$$

exists with $d \leq \triangle_1 n$ and $r \geq (\delta' - \triangle_2)n$, such that for all random variables $T \in_R \mathcal{T}$ with $\mathcal{T} \subseteq \{0,1\}^n$ and with

$$H_\infty(T) > \delta' n$$

we have

$$H(E(T,V)|V) \geq r - 2^{-n^{1/2-o(1)}}.$$

3.3 Authentication Codes

Before we can discuss privacy amplification over a non-authentic public channel, authentication codes need to be studied, since these codes can provide unconditional authenticity for a public channel.

First we introduce the terminology of authentication codes. A general *authentication code* is a triple $(\mathcal{G}, \mathcal{S}, \mathcal{M})$ of finite sets and a map $e : \mathcal{G} \times \mathcal{S} \to \mathcal{M}$. Here $\mathcal{G}$ is called the set of *source states*, which are pieces of information to be transmitted. For any specific $s, s \in \mathcal{S}$, which is called an *encoding rule*, the map e maps $\mathcal{G}$ to a subset of $\mathcal{M}$, the set of *messages*, of size $|\mathcal{G}|$. An encoding rule is also called an *authentication key*, and it is the common information shared by Alice and Bob beforehand, but Eve has no access to it. There are two kinds of attacks. In an *impersonation attack*, Eve inserts a message on the channel and impersonates the sender. In a *substitution attack*, Eve replaces a correct message over the channel by a false one. The success probabilities for Eve when trying these attacks are denoted by P_I and P_S, respectively. The probability of *deception* is defined by $P_D = \max(P_I, P_S)$. For any authentication codes, we have the trivial inequalities $P_I \geq \frac{|\mathcal{G}|}{|\mathcal{M}|}$ and $P_S \geq \frac{|\mathcal{G}|-1}{|\mathcal{M}|-1}$.

In the context of privacy amplification, Alice randomly chooses a hash function, denoted by the random variable G, from a class of hash functions, and transfers it to Bob. As shown in Figure 3.1, the random variable G has the same length as S, the partially secret string used to generate the final secret. Since G does not necessarily need to be secret, authentication codes without secrecy are enough for privacy amplification against Eve's active attacks. Therefore, we take interest in authentication codes where the messages carry all information about the source states. These kinds of authentication codes are called *Cartesian codes*, or *systematic authentication codes*, in analogy to systematic codes in coding theory. A Cartesian code is a triple $(\mathcal{G}, \mathcal{S}, \mathcal{X})$ of finite sets and a map $f : \mathcal{G} \times \mathcal{S} \to \mathcal{X}$, where for any $s \in \mathcal{S}$ and $g \in \mathcal{G}$, any message is of the form $m = (g, x)$ with $x = f(g, s)$. The value $x \in \mathcal{X}$ is called the *tag* or *authenticator* of g. According to Theorem 1 in [29], one has

$$P_S \geq P_I$$

for Cartesian codes. Authentication codes with $P_I = \frac{|\mathcal{G}|}{|\mathcal{M}|}$ are optimal against the impersonation attack, and also called *I-equitable authentication codes*. It is easy to see that any Cartesian I-equitable authentication code satisfies $P_D = P_S \geq 1/|\mathcal{X}|$.

3.3.1 Authentication Codes with Totally Secret Keys

Results on authentication codes, A-codes for short, are usually based on the assumption that the authentication key is totally secret to Eve (or the distribution of the authentication key is a uniform one to Eve). The first to consider the prob-

lem of authentication are Gilbert, MacWilliams and Sloane in [28]. Wegman and Carter proposed in [74] to use *ϵ-almost strongly universal$_2$* hash functions for authentication. Stinson obtained new bounds and gave general constructions for these kinds of hash functions in [72]. From these, he got the corresponding A-codes without secrecy. Multi-round protocols were suggested in [27]. These may reduce the amount of secret information needed for authentication keys, when compared with one-round protocols. Relationships between A-codes and error-correcting codes are suggested by Johanson, Kabatianskii and Smeets in [32] and [29].

We shall first properly define several notions of universal hashing and then show constructions for A-codes based on some of these classes of universal hash functions.

Let $\mathcal{A}$ and $\mathcal{B}$ be finite sets. For a hash function $h : \mathcal{A} \to \mathcal{B}$, for $a_1, a_2 \in \mathcal{A}, a_1 \neq a_2$, define

$$\delta_h(a_1, a_2) = \begin{cases} 1, & \text{if } h(a_1) = h(a_2), \\ 0, & \text{otherwise.} \end{cases}$$

For a finite set $\mathcal{H}$ of hash functions, all from $\mathcal{A}$ to $\mathcal{B}$, define

$$\delta_{\mathcal{H}}(a_1, a_2) = \sum_{h \in \mathcal{H}} \delta_h(a_1, a_2).$$

So, $\delta_{\mathcal{H}}(a_1, a_2)$ counts the number of hash functions, for which a_1 and a_2 collide.

Definition 3.3.1 *Let $\epsilon > 0$, $\mathcal{H}$ is ϵ-almost universal$_2$ (or ϵ-AU$_2$) if $\delta_{\mathcal{H}}(a_1, a_2) \leq \epsilon|\mathcal{H}|$ for all $a_1, a_2 \in \mathcal{A}, a_1 \neq a_2$.*

It was shown in [68] that for any ϵ-AU$_2$ class from $\mathcal{A}$ to $\mathcal{B}$,

$$\epsilon \geq \frac{|\mathcal{A}| - |\mathcal{B}|}{|\mathcal{B}|(|\mathcal{A}| - 1)}.$$

Definition 3.3.2 *Let $\epsilon > 0$, $\mathcal{H}$ is ϵ-almost strongly-universal$_2$ (or ϵ-ASU$_2$) if*

(a) *for every $a \in \mathcal{A}$ and for every $b \in \mathcal{B}$, $|\{h \in \mathcal{H} : h(a) = b\}| = |\mathcal{H}|/|\mathcal{B}|$;*

(b) *for every $a_1, a_2 \in \mathcal{A}$ $(a_1 \neq a_2)$ and for every $b_1, b_2 \in \mathcal{B}$, $|\{h \in \mathcal{H} : h(a_1) = b_1,\ h(a_2) = b_2\}| \leq \epsilon|\mathcal{H}|/|\mathcal{B}|$.*

The lower bound on ϵ in any ϵ-ASU$_2$ class from $\mathcal{A}$ to $\mathcal{B}$ is $1/|B|$.

Definition 3.3.3 *$\mathcal{H}$ is called universal$_2$ (or just universal) if it is $(1/|\mathcal{B}|)$-AU$_2$.*

Example 3.3.4 *Let $\mathcal{A} = \{0,1\}^n$ and $\mathcal{B} = \{0,1\}^r$. The set of all linear functions from $\mathcal{A}$ to $\mathcal{B}$ is universal.*

A linear function from $\mathcal{A} = \{0,1\}^n$ to $\mathcal{B} = \{0,1\}^r$ can be described by an $n \times r$ matrix M. The element a in $\mathcal{A}$ is mapped to $b = a \cdot M,\ b \in \mathcal{B}$. Hence the number of such functions is $2^{n \cdot r}$ and the description of such a function involves $n \cdot r$ bits.

The following example also gives a universal$_2$ class of hash functions, but it needs only n bits to describe a hash function from this class.

Example 3.3.5 *The set of functions* $f_h : \{0,1\}^n \to \{0,1\}^r$, *defined by* $f_h(a) = LSB_r(h \cdot a)$ *where* $(\cdot)$ *denotes the field multiplication in* $GF(2^n)$ *and* $LSB_r(x)$ *stands for the* r *least significant bits of a string* x, *is a universal class with cardinality* 2^n.

Definition 3.3.6 *If* $\mathcal{H}$ *is* ϵ-ASU_2 *with* $\epsilon = 1/|\mathcal{B}|$, *then* $\mathcal{H}$ *is called strongly universal*$_2$ *(or strongly universal,* SU_2 *for short).*

We denote strongly universal$_2$ by SU_2 and universal$_2$ by U_2 for convenience. The value $1/|\mathcal{B}|$ is the minimal value of ϵ for any ϵ-AU$_2$ and ϵ-ASU$_2$.

Example 3.3.7 *The set of functions* $\mathcal{H} = \{h_{s_1 s_2} : s_1, s_2 \in GF(2^n)\}$ *with* $h_{s_1 s_2}(g) = s_1 \cdot g + s_2$ *is an* SU_2 *class of hash functions from* $\{0,1\}^n \to \{0,1\}^n$ *of cardinality* 2^{2n}.

For any ϵ-ASU$_2$ class of hash functions from $\mathcal{A}$ to $\mathcal{B}$, a corresponding A-code can be constructed as follows. Take the elements of $\mathcal{A}$ as source states, those of $\mathcal{B}$ as authenticators and each hash function from ϵ-ASU$_2$ as an encoding rule (choose according to a uniform distribution). This idea is summarized in the following theorem.

Theorem 3.3.8 (Stinson, [72]) *If there exists an* ϵ-ASU_2 *class of hash functions* $\mathcal{H}$ *from* $\mathcal{A}$ *to* $\mathcal{B}$, *then there exists an authentication code for* $|\mathcal{A}|$ *source states, having* $|\mathcal{B}|$ *authenticators and* $|\mathcal{H}|$ *encoding rules, such that* $P_I = 1/|\mathcal{B}|$ *and* $P_S \leq \epsilon$.

The A-codes we constructed from ϵ-ASU$_2$ are Cartesian A-codes, which means that the transmitted message is the source state concatenated with the corresponding authenticator. Since A-codes made from ϵ-ASU$_2$ classes of hash functions have $P_I = \frac{1}{|\mathcal{B}|} = \frac{|\mathcal{A}|}{|\mathcal{M}|}$, we know that such A-codes are also I-equitable A-codes. Therefore $P_D = P_S \leq \epsilon$ with $\epsilon \geq 1/|\mathcal{B}|$.

Here is an example of a construction from SU_2.

Construction 3.3.9 *For some positive integer* k, *let* $\mathcal{A} = \{a = (a_1, a_2, \ldots, a_k); a_i \in GF(q), i = 1, 2, \ldots, k\}$ *be the set of source states. Let* $\mathcal{H} = \{h = (h_1, h_2, \ldots, h_{k+1}); h_j \in GF(q), j = 1, 2, \ldots, k+1\}$ *be the set of encoding rules and let* $\mathcal{B} = GF(q)$ *be the set of authenticators. Then a message* m *is given by a pair* $m = (a, b)$ *with*

$$b = h_1 \cdot a_1 + h_2 \cdot a_2 + \cdots + h_k \cdot a_k + h_{k+1}.$$

Construction 3.3.9 gives parameters $|\mathcal{A}| = q^k$, $|\mathcal{B}| = q$, and $|\mathcal{H}| = q^{k+1}$.

Since encoding rules are secret information between Alice and Bob, we would like to use a small number of encoding rules to authenticate as many source states as possible. However, there is a conflict between minimizing the number of encoding rules, maximizing the number of source states, and minimizing P_D. Based on SU_2, both P_I and P_S achieve minimal values, $1/|\mathcal{B}|$, but in [72] it was shown that

$$(|\mathcal{B}| - 1)|\mathcal{A}| \leq |\mathcal{H}| - 1,$$

which means that the number of source states is at best linearly bounded by the number of possible encoding rules. On the other hand, with ϵ-ASU$_2$, the number of encoding rules can be reduced significantly for a fixed number of source states by increasing P_S. More precisely, if we don't require P_S to be minimal, then for a fixed probability of substitution $P_S \leq \epsilon, \epsilon > 1/|\mathcal{B}|$, the number of source states in the best A-code(s) increases exponentially with the number of encoding rules. Below are two constructions. Construction 3.3.10 was proposed in [6] (see also [29], [32]). It was also applied in different settings in [60] and [16]. Construction 3.3.11 was presented by Stinson in [72].

Construction 3.3.10 *Let $\mathcal{A} = (GF(q))^k$ be the set of source states and write $a = (a_1, a_2, \ldots, a_k)$ for $a \in \mathcal{A}$. Define the source state polynomial to be $a(x) = a_1 x + a_2 x^2 + \cdots + a_k x^k$. It is a mapping from $GF(q)$ to itself. The set $\mathcal{H} = \{(h_1, h_2); h_1, h_2 \in GF(q)\}$ describes the set of the encoding rules as follows: a message m is given by the pair*

$$m = (a, h_1 + a(h_2)).$$

Construction 3.3.11 *Let $\mathcal{A} = \{a = (a_1, a_2, \ldots, a_{2^k}); a_i \in GF(q),\ i = 1, 2, \ldots, 2^k\}$ be the set of source states. Let $\mathcal{H} = \{h = (h_1, h_2, \ldots, h_{k+2}); h_j \in GF(q), j = 1, 2, \ldots, k+2\}$ be the set of encoding rules and let $\mathcal{B} = GF(q)$ be the set of authenticators. Then a message m is given by a pair $m = (a, b)$ with*

$$b = h_{k+1} + a_1^{(k)} \cdot h_{k+2},$$

where $a_1^{(k)}$ follows recursively from $a^{(l)} = (a_1^{(l)}, a_2^{(l)}, \ldots, a_{2^{k-l}}^{(l)})$, $a_j^{(l)} = a_{2j-1}^{(l-1)} + a_{2j}^{(l-1)} \cdot h_l$, for $j = 1, 2, \ldots, 2^{k-l}$ and $l = 1, 2, \ldots, k$ with initial value $a^{(0)} = a$.

We summarize the parameters of the A-codes from the above three constructions in Table 3.1.

A-codes	$\log \lvert\mathcal{H}\rvert$	$\log \lvert\mathcal{A}\rvert$	$\frac{\log \lvert\mathcal{A}\rvert}{\log \lvert\mathcal{H}\rvert}$	$\log \lvert\mathcal{B}\rvert$	P_D
Construction 3.3.9	$(k+1)\log q$	$k \log q$	$\frac{k}{k+1}$	$\log q$	$\leq \frac{1}{q}$
Construction 3.3.10	$2 \log q$	$k \log q$	$\frac{k}{2}$	$\log q$	$\leq \frac{k}{q}$
Construction 3.3.11	$(k+2)\log q$	$2^k \log q$	$\frac{2^k}{(k+2)}$	$\log q$	$\leq \frac{k+1}{q}$

Table 3.1: Parameters for A-codes with a totally secret authentication key

In Table 3.1 we have listed $\log |\mathcal{H}|$, the number of bits needed to describe an encoding rule (i.e., the length of the authentication key), and $\log |\mathcal{A}|$, the length of a source state which can be authenticated with the authentication key. We also listed $\log |\mathcal{A}|/\log |\mathcal{H}|$ because this rate indicates how many bits of source state can be authenticated by one bit of the authentication key given an upper bound on P_D. On the other hand, one can always choose q sufficiently large to ensure that P_D is less than a given upper bound.

3.3.2 Authentication Codes with Partially Secret Keys

In the previous subsection, we assume that encoding rules are chosen according to a uniform distribution, in other words, the authentication key is completely unknown to Eve. This subsection will mainly focus on the study of A-codes with only a partially secret authentication key. This partial secrecy is characterized by the fact that Eve's information about S, denoted by her Rényi entropy $H_2(S|Z=z)$, is less than the length of S. Here we use S to denote the partially secret string between Alice and Bob (as opposed to H in Subsection 3.3.1 which represents a totally secret authentication key).

For the constructions presented in Subsection 3.3.1, we shall derive upper bounds for P_I and P_S under the condition that $H_2(S|Z=z) \geq t \log |\mathcal{S}|$, $0 < t < 1$. First, we quote a lemma that gives a probabilistic upper bound on the reduction of Rényi entropy of a random variable S by obtaining additional side information A and B, where it is assumed that the mutual information between S and A is zero.

Lemma 3.3.12 (Wolf, [62]) *Let S, A and B be random variables with $I(S;A) = 0$. Then*

$$Pr[H_2(S|A=a, B=b) \geq H_2(S) - \log|\mathcal{B}| - s] \geq 1 - 2^{-(s/2-1)}$$

for all $s > 2$, and

$$Pr[H_\infty(S|A=a, B=b) \geq H_\infty(S) - \log|\mathcal{B}| - s] \geq 1 - 2^{-s}$$

for $s > 0$.

This lemma shows that with high probability the Rényi entropy decreases at most by $\log|\mathcal{B}|$. In the following theorem, we will derive the upper bounds on P_I and P_S, given an lower bound on Eve's information about the authentication key.

Theorem 3.3.13 *Suppose that the authentication key S in an ϵ-ASU_2 class of hash functions from $\mathcal{A}$ to $\mathcal{B}$ is only partially secret and let Eve's information about S be characterized by $H_2(S|Z=z) \geq t \log|\mathcal{S}|$. Then the construction described in Theorem 3.3.8 results in an A-code with parameters P_I and P_S, satisfying*

$$P_I \leq 2^{-\left(\frac{t-1}{2}\log|\mathcal{S}| + \frac{1}{2}\log|\mathcal{B}|\right)} = |\mathcal{S}|^{-\frac{t-1}{2}} \cdot |\mathcal{B}|^{-\frac{1}{2}},$$

$$\begin{aligned} P_S &\leq \left(\sqrt{\epsilon \cdot |\mathcal{B}|} + 2\right) \cdot 2^{-\left(\frac{t-1}{4}\log|\mathcal{S}| + \frac{1}{4}\log|\mathcal{B}|\right)} \\ &= \left(\sqrt{\epsilon \cdot |\mathcal{B}|} + 2\right) \cdot |\mathcal{S}|^{-\frac{t-1}{4}} \cdot |\mathcal{B}|^{-\frac{1}{4}}. \end{aligned}$$

Proof: Let $\mathcal{S} : \mathcal{A} \to \mathcal{B}$ be the set of hash functions from ϵ-ASU_2. First, we want to estimate the probability of a successful impersonation attack by Eve, i.e, to determine the probability that Eve, who has partial knowledge about the encoding rule, can successfully guess a pair (a, b) such that $b = s(a)$ for some s shared between

Alice and Bob. From the properties of ϵ-ASU$_2$, we note that for a given pair (a, b), the number of encoding rules satisfying $b = s(a)$ is $|\mathcal{S}|/|\mathcal{B}|$, in formula,

$$|\{s \in \mathcal{S} : s(a) = b\}| = \frac{|\mathcal{S}|}{|\mathcal{B}|}. \tag{3.1}$$

Let $Z = z$ denote all knowledge Eve knows about S before she sees any valid pair. Let p_i, $1 \leq i \leq |\mathcal{S}|$, denote the probabilities that Eve has assigned to all encoding rules and let the first $|\mathcal{S}|/|\mathcal{B}|$ of these coincide with the encoding rules that give a valid pair (a, b). Then

$$\sum_{i=1}^{|\mathcal{S}|/|\mathcal{B}|} p_i^2 \leq \sum_{i=1}^{|\mathcal{S}|} p_i^2 = P_c(S|Z = z) = 2^{-H_2(S|Z=z)} \leq 2^{-t \log |\mathcal{S}|}, \tag{3.2}$$

and

$$P_I = \sum_{i=1}^{|\mathcal{S}|/|\mathcal{B}|} p_i.$$

Since $\sum_{i=1}^{|\mathcal{S}|/|\mathcal{B}|} p_i$ achieves its maximal value when $p_1 = p_2 = \ldots = p_{|\mathcal{S}|/|\mathcal{B}|}$, it follows that

$$P_I \leq \left(\frac{|\mathcal{S}|}{|\mathcal{B}|}\right)^{1/2} \cdot 2^{-\frac{t}{2} \log |\mathcal{S}|} = 2^{-\left(\frac{t-1}{2} \log |\mathcal{S}| + \frac{1}{2} \log |\mathcal{B}|\right)}.$$

A successful substitution attack means that Eve has guessed a correct pair (a', b') after having seen a valid pair (a, b), where $a' \neq a$. The number of encoding rules that give rise to both pairs is upper bounded by $\epsilon \cdot |\mathcal{S}|/|\mathcal{B}|$, i.e.,

$$\{s : b = s(a),\ b' = s(a')\} \leq \epsilon \cdot |\mathcal{S}|/|\mathcal{B}|. \tag{3.3}$$

Eve's Rényi entropy about S decreases after observing (a, b) from $H_2(S|Z = z)$ to $H_2(S|Z = z, A = a, B = b)$. According to Lemma 3.3.12, we know that

$$H_2(S|Z = z, A = a, B = b) \geq \left(\frac{t}{2} + \frac{1}{2} - \frac{3}{2}\frac{\log |\mathcal{B}|}{\log |\mathcal{S}|}\right) \log |\mathcal{S}| \tag{3.4}$$

holds with probability $1 - 2^{-((t-1)\log |\mathcal{S}| + \log |\mathcal{B}|)/4+1}$. Here $\Pr[S|Z = z]$ is used in Lemma 3.3.12 instead of $\Pr[S]$. Just like in the impersonation attack, Eve gets the best result if there are exactly $\epsilon \cdot |\mathcal{S}|/|\mathcal{B}|$ encoding rules matching (a', b') and (a, b) and these $\epsilon \cdot |\mathcal{S}|/|\mathcal{B}|$ encoding rules all have the same probability, say p. If (3.4) holds, it follows that

$$\left(\epsilon \cdot \frac{|\mathcal{S}|}{|\mathcal{B}|}\right) \cdot p^2 \leq P_c(S|Z = z, A = a, B = b) \leq 2^{-\left(\frac{t}{2} + \frac{1}{2} - \frac{3}{2}\frac{\log |\mathcal{B}|}{\log |\mathcal{S}|}\right) \log |\mathcal{S}|}, \tag{3.5}$$

and we get

$$P_S \leq \left(\epsilon \cdot \frac{|\mathcal{S}|}{|\mathcal{B}|}\right) \cdot p \leq \sqrt{\epsilon \cdot |\mathcal{B}|} \cdot 2^{-\left(\frac{t-1}{4} \log |\mathcal{S}| + \frac{1}{4} \log |\mathcal{B}|\right)}.$$

When (3.4) does not hold, which occurs with probability at most

$$2^{-\left(\frac{t-1}{4}\log|\mathcal{S}|+\frac{1}{4}\log|\mathcal{B}|\right)+1},$$

we conservatively assume $P_S = 1$. Taking the two cases into account, we get

$$\begin{aligned} P_S &\le 2^{-\left(\frac{t-1}{4}\log|\mathcal{S}|+\frac{1}{4}\log|\mathcal{B}|\right)+1} + \sqrt{\epsilon\cdot|\mathcal{B}|}\cdot 2^{-\left(\frac{t-1}{4}\log|\mathcal{S}|+\frac{1}{4}\log|\mathcal{B}|\right)} \\ &\le \left(\sqrt{\epsilon\cdot|\mathcal{B}|}+2\right)\cdot 2^{-\left(\frac{t-1}{4}\log|\mathcal{S}|+\frac{1}{4}\log|\mathcal{B}|\right)}. \end{aligned}$$

□

Remark. We claim that the upper bounds for P_I and P_S are generally not tight. For example, when $t = 1$, we know that $P_I = 1/|\mathcal{B}|$, and $P_S \le \epsilon$, but the above theorem gives $P_I \le (|\mathcal{B}|)^{-\frac{1}{2}}$ and $P_S \le \left(\sqrt{\epsilon\cdot|\mathcal{B}|}+2\right)\cdot(|\mathcal{B}|)^{-\frac{1}{4}}$. The loose upper bounds for P_I and P_S are due to (3.1), (3.3), and (3.4). For instance, when we estimate the upper bound for P_I, we use the fact that $\sum_{i=1}^{|\mathcal{S}|/|\mathcal{B}|} p_i^2 \le \sum_{i=1}^{|\mathcal{S}|} p_i^2 = P_c(S|Z=z)$, and equality holds only when $p_1 = p_2 = \cdots = p_{|\mathcal{S}|/|\mathcal{B}|} = |\mathcal{B}|/|\mathcal{S}|$, in which case $P_I = 1$. In other words, we neglect the nonnegative part, $\sum_{i=|\mathcal{S}|/|\mathcal{B}|+1}^{|\mathcal{S}|} p_i^2$, in the left part of (3.1) while its contribution can be as high as $(1 - 1/|\mathcal{B}|)$. However, when $t < 1$, it is hard to say anything about the exact value of $\sum_{i=|\mathcal{S}|/|\mathcal{B}|+1}^{|\mathcal{S}|} p_i^2$, given only that $\sum_{i=1}^{|\mathcal{S}|} p_i^2 \le 2^{-t\log|\mathcal{S}|}$. That the upper bound for P_S is loose is for the same reason, see (3.3) and also due to the probabilistic upper bound of the reduction of Rényi entropy, see (3.4).

When Eve's partial knowledge about the authentication key S is characterized by the min-entropy, instead of Rényi entropy, we get the following corollary. We omit the proof because it is very similar to the proof of Theorem 3.3.13.

Corollary 3.3.14 *If one replaces the assumption in Theorem 3.3.13 on Eve's knowledge about S by $H_\infty(S|Z=z) \ge t\log|\mathcal{S}|$, an A-code results with parameters*

$$P_I \le 2^{-[(t-1)\log|\mathcal{S}|+\log|\mathcal{B}|]},\ P_S \le (\epsilon\cdot|\mathcal{B}|+1)\,2^{-\left(\frac{t-1}{2}\log|\mathcal{S}|+\frac{1}{2}\log|\mathcal{B}|\right)}.$$

The above results suggest that a substitution attack is more powerful than an impersonation attack, in which case $P_D = P_S$. To make sure that the upper bounds for P_I and P_S are less than 1, one needs $t > 1 - \log|\mathcal{B}|/\log|\mathcal{S}|$.

Applying Theorem 3.3.13 and Corollary 3.3.14 to the constructions in the previous subsection for A-codes from ϵ-ASU$_2$, we get the following corollary.

Corollary 3.3.15 *Suppose that Eve's information $Z = z$ about the authentication key S satisfies $H_\alpha(S|Z=z) \ge t\log|\mathcal{S}|$, where $\alpha = \{2,\infty\}$. Then the authentication codes of Construction 3.3.9, 3.3.10, and 3.3.11 have parameters given in Table 3.2.*

Table 3.1 shows the trade-off between the number of encoding rules, the number of source states, and P_D. Now that the encoding rules are partially known to Eve,

A-codes	$\log \lvert\mathcal{S}\rvert$	$\frac{\log \lvert\mathcal{A}\rvert}{\log \lvert\mathcal{S}\rvert}$	$\log \lvert\mathcal{B}\rvert$
Construction 3.3.9	$(k+1)\log q$	$k/(k+1)$	$\log q$
Construction 3.3.10	$2\log q$	$k/2$	$\log q$
Construction 3.3.11	$(k+2)\log q$	$2^k/(k+2)$	$\log q$
A-codes $(\boldsymbol{H_2 \geq t \log \lvert\mathcal{S}\rvert})$	$\boldsymbol{P_I}$	$\boldsymbol{P_S}$	$\boldsymbol{t}$
Construction 3.3.9	$\leq q^{-\left(\frac{k+1}{2}t-\frac{k}{2}\right)}$	$\leq 3 \cdot q^{-\left(\frac{k+1}{4}t-\frac{k}{4}\right)}$	$> \frac{k}{k+1}$
Construction 3.3.10	$\leq q^{-(t-1/2)}$	$\leq (\sqrt{k}+2)q^{-(t/2-1/4)}$	$> 1/2$
Construction 3.3.11	$\leq q^{-(\frac{k+2}{2}t-\frac{k+1}{2})}$	$\leq (\sqrt{k+1}+2)q^{-(\frac{k+2}{4}t-\frac{k+1}{4})}$	$> \frac{k+1}{k+2}$
A-codes $(\boldsymbol{H_\infty \geq t \log \lvert\mathcal{S}\rvert})$	$\boldsymbol{P_I}$	$\boldsymbol{P_S}$	$\boldsymbol{t}$
Construction 3.3.9	$\leq q^{-(k+1)t+k}$	$\leq 2 \cdot q^{-\left(\frac{k+1}{2}t-\frac{k}{2}\right)}$	$> \frac{k}{k+1}$
Construction 3.3.10	$\leq q^{-(2t-1)}$	$\leq (k+1)q^{-(t-1/2)}$	$> 1/2$
Construction 3.3.11	$\leq q^{-(k+2)t+k+1}$	$\leq (k+2)q^{-(\frac{k+2}{2}t-\frac{k+1}{2})}$	$> \frac{k+1}{k+2}$

Table 3.2: Parameters for A-codes with a partially secret authentication key of $H_\alpha(S|Z=z) \geq t \log |\mathcal{S}|$

Eve's knowledge about the encoding rules also plays a role in the comparison. This can be seen in Table 3.2. To make the upper bounds for P_S less than 1, different constructions lead to different requirements for t.

A-codes based on Construction 3.3.9 authenticate a source state with a longer authentication key. For example, when $k = 1$, the length of the source states $\log |\mathcal{A}|$ is only half of $\log |\mathcal{S}|$, the length of authentication key. For increasing k, $\log |\mathcal{A}|$ approaches $\log |\mathcal{S}|$, but $t = k/(k+1)$ approaches to 1. That means a stronger requirement about Eve's knowledge about $\mathcal{S}$: she should know almost nothing about $\mathcal{S}$, since $H_\alpha(S|Z=z) \geq \frac{k}{k+1} \log |\mathcal{S}|$ is required.

Construction 3.3.10 only requires $t > 1/2$ and can authenticate a source state which is $k/2$ times as long as the partially secret authentication key S.

To authenticate a longer source state, one may consider the A-codes based on Construction 3.3.11. When k increases, $\log |\mathcal{A}|$ increases exponentially faster than $\log |\mathcal{S}|$ with as price a larger P_S than that in Construction 3.3.9. However, the same problem arises as in Construction 3.3.9. When k is large, S should be highly secret to Eve, namely, $H_\alpha(S|Z=z) \geq \frac{k+1}{k+2} \log |\mathcal{S}|$. This assumption may not be very realistic in the context of privacy amplification.

3.4 Application of A-codes to Privacy Amplification

First we give a formal definition of privacy amplification over a non-authentic channel and then we discuss three approaches in detail for three different settings.

Definition 3.4.1 ([62]) *Suppose that Alice and Bob share a mutual n-bit random variable S, and let the random variable Z denote Eve's knowledge about S. Let $\mathcal{D}$ be a subset of all probability distributions on the set of n-bit strings, let r be an integer, and let $0 < \epsilon, \delta < 1$. Alice and Bob communicate over an insecure and non-authentic channel.*

A weak $(n, \mathcal{D}, r, \epsilon, \delta)$-protocol for privacy amplification *or a* weak $(n, \mathcal{D}, r, \epsilon, \delta)$ *PA protocol for short, satisfies properties (i) and (ii) listed below, while a* strong *PA protocol satisfies (i) and (iii).*

(i) Correctness and privacy: If Eve is a passive wire-tapper, Alice and Bob compute and accept S'_A, respectively S'_B, at the end of the protocol with $S' = S'_A = S'_B$. Further $H(S'|C, Z = z) \geq r - \epsilon$, where C stands for the entire communication held over the public channel.

(ii) Weak robustness: For every possible strategy of Eve, the probability that at least one party, Alice or Bob, rejects the outcome of the protocol or that both Alice and Bob accept the outcome (in which case the privacy amplification protocol has been successful), must be at least $1 - \delta$.

(iii) Strong robustness: For every possible strategy of Eve, the probability that both Alice and Bob reject the outcome of the protocol or that privacy amplification has been successful must be at least $1 - \delta$.

Remark. Here, we assume that Alice and Bob are honest players, always following the rules of the protocol.

In this chapter, $\mathcal{D}_{n,\alpha,tn}$ denotes the set of probability distributions $\Pr[S|Z = z]$ such that $H_\alpha(S|Z = z) > tn$.

By saying that privacy amplification has been successful we mean that both Alice and Bob in fact have obtained $S' = S'_A = S'_B$. This covers two cases: one is that both Alice and Bob accept, the other is that one accepts and the other rejects the outcome, although it was obtained correctly. The latter can occur if Eve deletes the last message in the PA-protocol. To enable privacy amplification against active attacks over a non-authentic channel, the only thing Alice and Bob need to do is to authenticate the description of g, the hash function, over the non-authentic channel. An alternative is to use some random bits for the input of privacy amplification extractors, see Section 3.2. Below we shall use the traditional universal hash technique. To authenticate g, Alice and Bob need two strings, S_I for authentication and S_{II} from which they will distill the final secret.

It is easy to see that the *one-way transmission protocol* depicted in Figure 3.2 can accomplish weak robustness by using an A-code with S_I as authentication key. This ensures that $P_D \leq \delta$.

For strong robustness, an extra thing needs to be done. Alice adds a random challenge to Bob, Bob then computes and returns the corresponding authenticator to Alice, and Alice won't accept the result unless she gets the correct authenticator for her challenge. We call this the *interactive protocol*.

The interactive protocol (Figure 3.3) can avoid the case that one party accepts while privacy amplification is not successful. This may occur in the one-way transmission protocol. Therefore, the interactive protocol can accomplish strong robust-

Alice		Bob
$S_I \in GF(2^{n_1})$,		$S_I \in GF(2^{n_1})$,
$S_{II} \in GF(2^{n_2})$		$S_{II} \in GF(2^{n_2})$
$g \in_R GF(2^{n_2}), x = S_I(g)$		
$S'_A = \text{LSB}_r(g \cdot S_{II})$		
accept	$\xrightarrow{(g,x)}$	
		accept if $x = S_I(g)$ **stop** otherwise
		$S'_B = \text{LSB}_r(g \cdot S_{II})$

Figure 3.2: One-way transmission protocol

ness (Property (iii) in Definition 3.4.1). It still ensures $P_D \leq \delta$, as did the one-way transmission protocol. The reason is the following. Eve can not replace (g, x) by (g', x') without Bob finding out with probability more than δ. This is due to the properties of the A-code. She cannot either, with probability more than δ, guess a correct response v to the challenge u from Alice in order to convince Alice that Bob had accepted the result. The remaining thing Eve can do is to replace u in (g, x, u) with another $u', u' \neq u$, or replace Bob's response v with $v', v' \neq v$, or block v altogether to prevent Alice from receiving it. But this does not make any sense any more because privacy amplification is already successful as soon as Bob has received the correct pair (g, x). We will consider different settings for the authentication

Alice		Bob
$S_I \in GF(2^{n_1})$,		$S_I \in GF(2^{n_1})$,
$S_{II} \in GF(2^{n_2})$		$S_{II} \in GF(2^{n_2})$
$g \in_R GF(2^{n_2}), x = S_I(g), u \in_R GF(2^{n_2})$		
$S'_A = \text{LSB}_r(g \cdot S_{II})$		
	$\xrightarrow{(g,x,u)}$	
		accept if $x = S_I(g)$
		stop otherwise
		$S'_B = \text{LSB}_r(g \cdot S_{II})$
		$v = S_I(u)$
	$\xleftarrow{v}$	
accept if $v = S_I(u)$		

Figure 3.3: Interactive protocol

problem in the context of privacy amplification in the following subsections. Alice and Bob will have access to different resources in the different settings, as will be illustrated below.

- Alice and Bob share two strings: S_I is a secret string of length n_1 and S_{II} is a partially secret string of length n_2 . Eve's knowledge about these strings is characterized by $H_2(S_I|Z_1 = z_1) = n_1$ (which means that for her the Rényi entropy of S_I is equal to its Shannon entropy, which is n_1) and $H_2(S_{II}|Z_2 = z_2) \geq t_2 n_2$. This setting will be discussed in Subsection 3.4.1;
- Both S_I and S_{II} are partially secret and their concatenation forms a string S of length n with $H_2(S|Z = z) > tn$. We will study this case in Subsection 3.4.2;
- Both S_I and S_{II} are partially secret but S_I and S_{II} are independent strings, with $H_2(S_I|Z_1 = z_1) \geq t_1 n_1$ and $H_2(S_{II}|Z_2 = z_2) \geq t_1 n_2$. This case will be analyzed in Subsection 3.4.3.

3.4.1 Application of Authentication Codes to Privacy Amplification When a Totally Secret Key Is Available

Suppose that Eve has no knowledge at all about S_I (except its length). Alice and Bob can use A-codes based on ϵ-ASU$_2$ with $\delta = \epsilon$ to convert the non-authentic channel to a channel that can authenticate messages with probability at least $1 - \delta$. Suppose that the A-code maps $\{0,1\}^{n_2}$ to $\{0,1\}^m$. Then Alice and Bob can always, in case of a passive attack by Eve, distill a secret key S' of length $r = H_2(S_{II}|Z_2 = z_2) - s$ from an n_2-bit partially secret string S_{II} such that Eve's information about S' is at most $2^{-s}/\ln 2$, where $s > 0$ is a security parameter (see Theorem 3.2.2). This is exactly Approach 1 described in Section 3.1.

We would like to use only a small number of secret bits (S_I) for authentication purposes during the privacy amplification process, because this results in a longer S_{II} and thus in more secret bits (S') obtained through privacy amplification. The longer string the secret string S_I can authenticate, the more bits privacy amplification may generate. Hence, the problem becomes the classical question in unconditional authentication: given $P_D < \delta$, how many secret bits do Alice and Bob need to share to authenticate a source state of given length n?

Wegman and Carter [74] describe a protocol for ϵ-almost strongly universal$_2$ which needs approximately $O(\log(n)\log(1/\delta))$ ($\epsilon = \delta$) secret bits, while Stinson [72] improves this result to $(2\log(n) + 3 - 2\log\log(1/\delta))(\log(1/\delta))$ (according to Construction 3.3.11) secret bits. All the results hold for the single-round protocols. Gemmell et al. [27] prove that a tighter lower bound on the number of shared secret bits is between $\log(n) + \log(1/\delta)$ and $\log(n) + 2\log(1/\delta)$. They also propose a multi-round protocol that requires $2\log(1/\delta) + O(1)$ bits. Note that this number is independent of the length of source states. When $\log(1/\delta) < \log(n)$, a multi-round protocol helps to decrease the number of secret bits. However, no protocol with a fixed number of rounds can achieve $2\log(1/\delta) + O(1)$ bits. In fact, any multi-round protocol needs $O(\log^*(n))$ rounds. See [27] for more details.

The above discussion can be summarized by the following theorem.

Theorem 3.4.2 *Let $t > 0, s > 0$. Suppose that there exists a δ-ASU_2 from $\{0,1\}^{n_2}$ to $\{0,1\}^m, \delta \geq 2^{-m}$, with cardinality $|\mathcal{S}|$, and Alice and Bob share a secret of length $n_1, n_1 \geq \log|\mathcal{S}|$. Then there exists a strong $(n_2, \mathcal{D}_{n_2,2,tn_2}, tn_2, 2^{-s}/\ln 2, \delta)$ PA-protocol.*

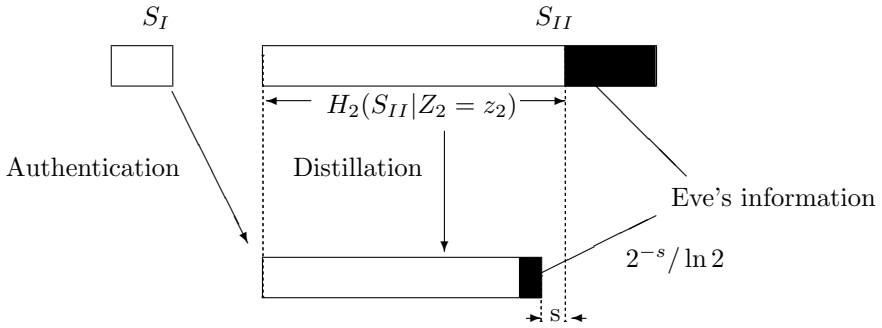

Figure 3.4: Privacy amplification with a totally secret authentication key

Another technique to implement privacy amplification, other than the traditional method with universal$_2$ functions, is to use extractors, as shown in Theorem 3.2.4. Similar to the traditional privacy amplification technique, some random bits are also needed to be transmitted from Alice to Bob if using extractors. As we pointed out earlier, the advantage of extractors is that the length of the random bits is only an asymptotically arbitrarily small fraction of S_{II}, but the final distilled secret key is measured by $H_\infty(S|Z = z)$. When considering the non-authentic case, we need a secret S_I for authentication that is again an asymptotically small fraction of S_{II}. We get the following theorem when extractors are used for privacy amplification.

Theorem 3.4.3 *Let $s > 0,\ 1 > t > 0,\ \triangle, \delta > 0$. Suppose that Alice and Bob share a secret of length n_1. Then for sufficiently large n_1 there exists a strong $(n_2, \mathcal{D}_{n_2,\infty,tn_2}, tn_2, 2^{-n_2^{1/2-o(1)}}, \delta)$ PA-protocol with $n_2 = n_1/\triangle$.*

We note that the traditional technique is practical while the extractor technique is non-constructive. On the other hand, the extractor technique can provide proof of the existence of protocols.

3.4.2 Application of Authentication Codes to Privacy Amplification when One Partially Secret String Is Available

We shall now consider the case that Alice and Bob have only a partially secret string S available to privacy amplification. As was suggested in Section 3.1, privacy

amplification can be realized by means of Approach 2. Suppose that S has length n. Alice and Bob split S into two parts, using one substring S_I for authentication and the remaining part S_{II} for distillation. Eve's information $Z = z$ about S is assumed to satisfy $H_2(S|Z = z) \geq tn$. Eve's information about S's substrings can be estimated by the following lemma.

Lemma 3.4.4 (Wolf, [64]) *Let $S = (S_1, S_2, \ldots, S_n)$ be a sequence of n binary, random variables. For any k-tuple $\underline{i} = (i_1, i_2, \ldots, i_k)$, where $1 \leq i_1 < i_2 < \cdots < i_k \leq n$, let $S_{\underline{i}}$ be the substring $(S_{i_1}, S_{i_2}, \ldots, S_{i_k})$. Then $H_\alpha(S_{\underline{i}}) \geq H_\alpha(S) - (n - k)$ for $\alpha = 2$ and $\alpha = \infty$.*

Theorem 3.4.5 *Suppose that in the setting of one partially secret string and Approach 2 an $(n, \mathcal{D}_{n,\alpha,tn}, r, \epsilon, \delta)$ PA-protocol (see Definition 3.4.1) exists with $\alpha \in \{2, \infty\}$, $0 < r < tn$, $\epsilon, \delta > 0$. Then $t > 1/2$.*

Proof: Let S_I be of length n_1 and S_{II} of length n_2, where $S = S_I || S_{II}$ and $n = n_1 + n_2$. From Lemma 3.4.4, we know that

$$H_\alpha(S_I|Z = z) \geq tn - n_2 = tn_1 - (1 - t)n_2 \tag{3.6}$$

$$H_\alpha(S_{II}|Z = z) \geq tn - n_1 = tn_2 - (1 - t)n_1 \tag{3.7}$$

To accomplish authentication and distillation, the right hand sides in Equations (3.6) and (3.7) need to be positive, which results in $t > 1/2$. □

Theorem 3.4.5 states that $1/2$ is a lower bound for parameter t when Eve's information satisfies $H_\alpha(S|Z = z) \geq tn$. Wolf proved the existence of a PA-protocol when $t = 2/3$ for $\alpha = \{2, \infty\}$. In [34], Liu et al. [34] claim that $t = 1/2$ is also achievable for $\alpha = 2$, while Maurer et al. in [52] claim this for $\alpha = \infty$. Both proofs use the same partially secret string S for authentication of a random string W, and as input of an extractor function (denoted by $E(W, S)$) as well. The idea of the extractor is to distill a secret key from the partially secret string S with the help of a small number of truly random bits W. It assumes that W and S are independent of each other. If part of S is used to authenticate W, however, the authenticator for W is public, so W and S are not independent of each other any more. Therefore, we claim that the problem whether privacy amplification against an active attack (with strong or weak robustness) is possible when $H_\alpha(S|Z = z) > n/2$ remains open. The best bound up to now is still $H_\alpha(S|Z = z) > 2n/3$.

Here, we summarize the best known results for the case that Alice and Bob share only a partially secret string S (See [52, 62]).

Theorem 3.4.6 *Let $2/3 < t < 1$ and $\triangle > 0$ be constants, $\alpha = \{2, \infty\}$. Then for sufficiently large n, strong $(n, \mathcal{D}_{n,\alpha,tn}, (t - 2/3 - \triangle)n, 2^{-\Omega(n)}, 2^{-\Omega(n)})$ PA-protocols exist.*

Since a strong protocol is also a weak one, the above theorem also implies the existence of weak protocols.

If $t > 2/3$, Alice and Bob have to separate S into S_I of length $2n/3$ and S_{II} of length $n/3$. They use S_I for authentication purposes, employing an A-code from Construction 3.3.9 with $k = 2$. They use S_{II} for the distillation of the secret key. The length of the final secret key is at most $(t - 2/3)n$ bits. Compared with the length of a secret key distilled by a privacy amplification protocol designed only against passive attacks, which is almost tn, we see that Alice and Bob pay the price of $2n/3$ bits of S to detect active attacks, so the final secret key also shrinks by $2n/3$. This price is really too high.

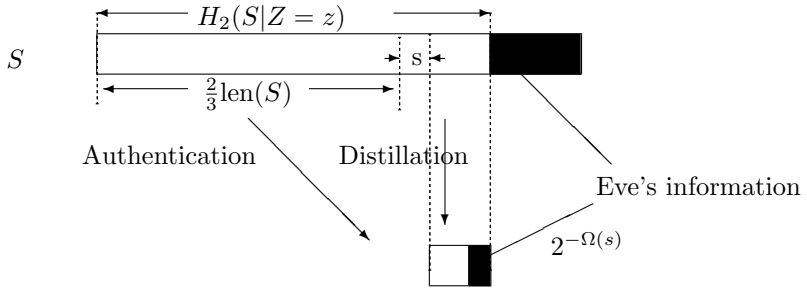

Figure 3.5: Privacy amplification with one partially secret authentication string

3.4.3 Application of Authentication Codes with Independent Partially Secret Strings to Privacy Amplification

We now consider the setting that Alice and Bob have two independent partially secret strings, say S_I and S_{II}, and assume that Eve's information about these are $Z_1 = z_1$ resp. $Z_2 = z_2$. Alice and Bob use S_I to authenticate the hash function G (in a one-way transmission protocol or in an interactive protocol), and take the hash value $G(S_{II})$ as the final secret. The problem is to find a lower bound on $H_2(S_I|Z = z)$, given that the probability that Eve can perform a successful active attack is upper bounded by a security parameter δ, $0 < \delta < 1$. What is the maximal length of S_{II} (that will be used to distill a secret key) for a fixed length of S_I and fixed value of δ.

We shall use Theorem 3.3.13 to give answers to these questions. As Table 3.2 suggests, Construction 3.3.10 is more suitable for privacy amplification against active attacks in this setting since it only requires $t > 1/2$. Suppose that Alice and Bob share two independent partially secret strings, say S_I of length n_1 and S_{II} of length $n_2 = kn_1/2$. Without loss of generality, we assume that n_1 is even. We show in Figure 3.6 how Alice and Bob can construct a strong PA-protocol by using an interactive transmission. Then Theorem 3.4.7 follows.

Theorem 3.4.7 *Let* $0 < \delta < 1$, $\frac{1}{2} < t_1 < 1$, $0 < t_2 < 1$, $\triangle_1, \triangle_2 > 0$, $n_1, n_2 \in \mathbb{N}$ *and consider the setting of two independent, partially secret strings, say* S_I *of length* n_1

and S_{II} of length n_2. Then, both strong and weak $(n_2, \mathcal{D}_{n_2,2,n_2t_2}, n_2t_2 - s, 2^{-s}/\ln 2, \delta)$ PA-protocols exist if the following two conditions are met:

(i) *Eve's Rényi entropy about S_I given her knowledge $Z_1 = z_1$ is lower bounded by n_1t_1, i.e., $H_2(S_I|Z_1 = z_1) \geq n_1t_1$,*

(ii) $n_2 \leq \frac{n_1}{2}\left(\delta 2^{(t_1/2-1/4)n_1} - 2\right)^2$.

Strong and weak $\left(n_2, \mathcal{D}_{n_2,\infty,n_2t_2}, n_2(t_2 - \triangle_1), 2^{-n_2^{1/2-o(1)}}, \delta\right)$ PA-protocols exist, for sufficiently large n_1, if

(1) *Eve's min-entropy given her knowledge $Z_1 = z_1$ is lower bounded by n_1t_1, i.e., $H_\infty(S_I|Z_1 = z_1) \geq n_1t_1$,*

(2) $n_2 \leq \frac{n_1}{2\triangle_2}\left(\delta 2^{(t_1-1/2)n_1} - 1\right)$.

Proof: We only need to show that the interactive protocol depicted in Figure 3.6 is a strong privacy amplification protocol. It is trivial to check that the one-way transmission protocol depicted in Figure 3.2 is weak.

Alice		Bob
$S_I = (S_1 \| \| S_2)$,		$S_I = (S_1 \| \| S_2)$,
$S_1, S_2 \in \{0,1\}^{n_1/2}$		$S_1, S_2 \in \{0,1\}^{n_1/2}$
$S_{II} \in \{0,1\}^{kn_1/2}$		$S_{II} \in \{0,1\}^{kn_1/2}$
$g \in_R GF(2^{kn_1/2}), x = S_1 + g(S_2)$		
$u \in_R GF(2^{kn_1/2})$		
$S'_A = \text{LSB}_r(g \cdot S_{II})$		
	$\xrightarrow{(g,x,u)}$	
		accept if $x = S_1 + g(S_2)$
		stop otherwise
		$S'_B = \text{LSB}_r(g \cdot S_{II})$
		$v = S_1 + u(S_2)$
	$\xleftarrow{v}$	
accept if $v = S_1 + u(S_2)$		

Figure 3.6: Interactive protocol using A-codes from Construction 3.3.10, where $g(\cdot)$ and $u(\cdot)$ denote two source state polynomials

For the case of a passive attack by Eve, if $H_2(S_{II}|Z_2 = z_2) \geq n_2t_2$, according to Theorem 3.2.2, Alice and Bob will get a final string $S' = S'_A = S'_B$ of length $n_2t_2 - s$ while $H(S'|G, Z = z) \geq n_2t_2 - s - \frac{2^{-s}}{\ln 2}$. This implies that Eve's knowledge about S' is not more than $2^{-s}/\ln 2$.

If $H_\infty(S_{II}|Z_2 = z_2) \geq n_2t_2$, then for constants $t_2, \triangle_1, \triangle_2 > 0$, there exists, according to Corollary 3.2.4, for sufficiently large n_2, an extractor $E : \{0,1\}^{n_2} \times$

$\{0,1\}^d \to \{0,1\}^r$, where $d \approx \triangle_1 n_2$, and $r \geq (\delta' - \triangle_2)n_2$, such that $H(E(S_{II}, W)|W) \leq r - 2^{n_2^{1/2-o(1)}}$.

The aim of an active attack by Eve is to make Alice (or Bob) accept S'_A (or S'_B) although the protocol does not run successfully. There are two possible active attacks for Eve. One possibility for Eve is to try to impersonate Alice to get Bob to accept while Alice rejects. Eve can forge a pair (g', x') to get Bob to accept it. She can do it with or without the help of Alice's valid pair (g, x), which corresponds to an impersonation attack resp. substitution attack. The other possibility is that Eve prevents Alice's valid pair (g, x) from reaching Bob after which she tries to give a correct response v' to Alice's challenge u hoping to convince Alice that she is Bob (this is just a substitution attack).

It follows from Corollary 3.3.15 that when $H_2(S_I|Z_1 = z_1) \geq n_1 t_1$, $t_1 > 1/2$, one can authenticate a source state of length $k/2$ times longer than S_I and the probability of a successful active attack by Eve for the interactive protocol is upper bounded by δ, as long as $k \leq \left(\delta 2^{(t_1/2-1/4)n_1} - 2\right)^2$, which means $n_2 \leq \frac{n_1}{2}\left(\delta 2^{(t_1/2-1/4)n_1} - 2\right)^2$.

In the case that an extractor is used, one needs to authenticate a d-bit random variable W with $d \leq \triangle_1 n_2$. If $H_\infty(S_I|Z_1 = z_1) \geq n_1 t_1$, then, according to Corollary 3.3.14, S_I can authenticate the string W of length $d = kn_1/2$ such that the probability of a successful active attack can be upper bounded by δ if $k \leq \left(\delta 2^{(t_1-1/2)n_1} - 1\right)$. On the other hand, W can serve as input to the extractor, together with S_{II}, to distill a secret. W has length d with $d \approx \triangle_1 n_2$. Hence $n_2 \leq \frac{n_1}{2\triangle_1}\left(\delta 2^{(t_1-1/2)n_1} - 1\right)$. □

We notice that when n_1 is large enough, for any n_2 and δ, a strong (and weak) PA-protocol exists.

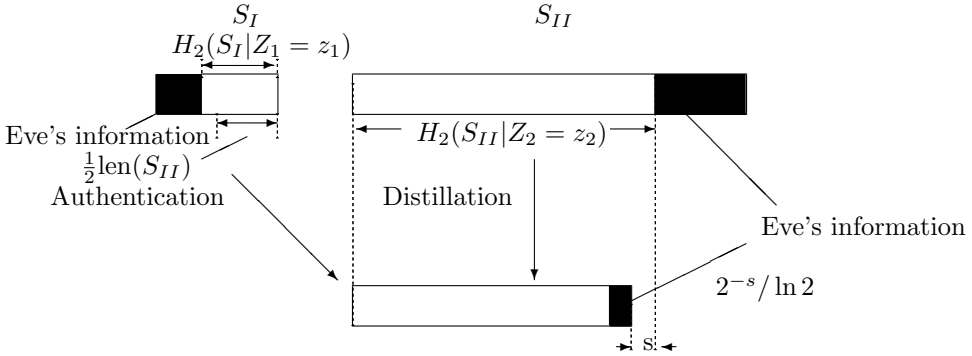

Figure 3.7: Privacy amplification with independent partially secret strings

3.5 Conclusion

Authentication codes have been developed to provide protection against a malicious adversary who has access to the communication channel and can insert fraudulent

messages or modify the messages over the channel. In this chapter, we reviewed some results of A-codes and showed how authentication codes provide unconditional authentication. Earlier results were usually obtained under the assumption that the sender and receiver have a common authentication key about which Eve knows nothing. In the context of privacy amplification, it is more reasonable to assume that Alice and Bob share a partially secret string instead of a totally secret one.

We studied the performance of A-codes from the ϵ-ASU$_2$ class with partially secret authentication key. Here, we characterized the partial secrecy by the fact that the Rényi entropy of order 2 about the secret string is less than its length. The application of authentication codes to privacy amplification is explored in three different settings, for which different approaches are presented. Approach 1 employs A-codes with a totally secret authentication key and is suitable for the case that Alice and Bob share an additional, totally secret string. Approaches 2 and 3 use A-codes with a partially secret authentication key. Approach 2 is for when Alice and Bob share only one partially secret string, while Approach 3 can be used when Alice and Bob share two independent, partially secret strings.

We claim that Approach 3 is more practical and more reasonable because

- In each approach, one has to solve the problem of how to renew the authentication key when Eve performs frequent active attacks. This is so because the authentication key can only be used once. Approach 1 requires that Alice and Bob share a shorter but totally secret string for unconditional authentication, but the disadvantage is that getting a fresh secret string is expensive. Especially when Eve performs continuous active attacks, Alice and Bob may sacrifice more secret bits than what they can get from privacy amplification. Then an unconditional secure secret key agreement can never be accomplished. On the other hand, getting partially secret strings is much cheaper: Alice and Bob can always run independent, unconditionally secure secret key agreement protocols [48] over an insecure and non-authentic channel to get partially secret strings to compensate for previously used partially secret strings. That is the advantage of Approach 2 and 3 over Approach 1.

- Approach 2 assumes that Alice and Bob only share one partially secret string S. However, this is very unattractive since it implies the stronger requirement that $H_2(S|Z = z) \geq tn$ with $t > 2/3$. When $t \leq 2/3$, Approach 2 will not work. Meanwhile, the length of the final secret key is only at most $(t - 2/3)n$. Further, in case of an active attack by Eve, Approach 3 only needs a short, independent, partially secret string S_I, satisfying $H_2(S_I|Z_1 = z_1) \geq t_1 n_1$ and $t_1 > 1/2$, for authentication of hash function g. This hash function will be applied to the other string, S_{II}, of length $k/2$ times longer than the length of S_I, during the privacy amplification phase to distill a final secret key of length about $H_2(S_{II}|Z_2 = z_2)$.

Chapter 4

Secret Key Agreement over a Non-Authentic Public Channel

4.1 Introduction

A secret key agreement protocol starts with an *initialization phase*, which enables Alice, Bob, and Eve to receive random variables X, Y, and Z, respectively. The three variables are jointly distributed according to some probability distribution P_{XYZ}. Next, in the *communication phase* Alice and Bob alternately send each other messages over a public channel. This process is known as *public discussion*. Through public discussion, Alice and Bob implement advantage distillation, information reconciliation, and privacy amplification. Finally, in the *decision phase* each of Alice and Bob either accepts or rejects the protocol execution, depending on whether they believe to be able to generate a secret key.

When the public channel between Alice and Bob is non-authentic, Eve may introduce fraudulent messages or modify messages over the public channel without detection, thus thwarting the secret key agreement. A problem arises: how do Alice and Bob authenticate the messages exchanged over the public channel during the communication phase? If Alice and Bob share an authentication key, they can employ authentication codes to detect Eve's active attacks. However, the aim of a secret key agreement by Alice and Bob is to get a secret string. So, it is not reasonable to assume some secret information between Alice and Bob in this context. On the other hand, even if a short authentication key is assumed to be shared between Alice and Bob, Eve's continuous active attacks may use up the common authentication key.

In Chapter 3, we studied how Alice and Bob achieve authenticity with a partially secret string as the authentication key. A necessary condition for Alice and Bob to

accomplish authenticity, namely, the upper bound for Eve's Rényi entropy of the authentication key, was derived. This result can be applied to privacy amplification since common strings are usually assumed between Alice and Bob in the context of privacy amplification. Privacy amplification can be seen as a special case of the general secret key agreement protocol where Alice and Bob share a common string $X = Y$, about which Eve has some knowledge.

In the general case of secret key agreement, there is no common string between Alice and Bob like there is in the privacy amplification stage. They have to use their correlated strings that were obtained during the initialization phase to accomplish authenticity for their communication over the public channel. However, not all probability distributions P_{XYZ} imply that authenticity can be achieved. In fact, only for those distributions where Alice and Bob have some advantage over Eve, authenticity can be accomplished (between Alice and Bob).

In [48], a so-called *simulatability condition* characterizes those distributions for which no authenticity can be achieved. The simulatability condition will be introduced in Section 4.2. In the subsequent sections, the satellite scenario is again considered in the initialization phase, where we assume that both Alice and Bob have better (in terms of lower bit error probabilities) binary symmetric channels than Eve. Then we study how Alice and Bob use their correlated strings obtained in the initialization phase of the secret key agreement protocol to authenticate the messages over the public channel based on error correction techniques. The messages exchanged during the *communication phase* do not serve advantage distillation since we already assume an advantage (in terms of mutual information) between Alice and Bob over Eve. For the satellite scenario, the assumption of more mutual information between Alice and Bob is equivalent to the non-simulatability condition. Since we have already solved the authentication problem for privacy amplification, this chapter actually deals with the problem how to authenticate messages for information reconciliation.

This chapter is mainly based on [35].

4.2 A Necessary Condition for Secret Key Agreement against Active Adversaries

In this section, we define a class of probability distributions P_{XYZ}, for which secret key agreement against active adversaries is impossible. This class of distributions is characterized by the *simulatability condition*. This concept was first proposed in [48].

Definition 4.2.1 *Let X, Y, and Z be random variables with joint probability distribution P_{XYZ}. If there exists a conditional probability distribution $P_{\overline{X}|Z}$ such that $P_{\overline{X}Y}$ $(= \sum_X \sum_Z P_{XYZ} P_{\overline{X}|Z}) = P_{XY}$, then X is simulatable by Z with respect to Y, denoted by* $\mathrm{sim}_Y(Z \to X)$.

In words, $\mathrm{sim}_Y(Z \to X)$ means that it is possible for Eve to change her random variable Z to another variable $\overline{X}$ such that Bob cannot tell the difference between Eve's forged variable $\overline{X}$ and Alice's variable X. Hence Alice cannot send any authenticated message to Bob.

In fact, $\mathrm{sim}_Y(Z \to X)$ implies that $XY \to Z \to \overline{X}$ is a Markov chain and $P_{\overline{X}Y} = P_{XY}$. Another way to explain $\mathrm{sim}_Y(Z \to X)$ is the following. There exists a channel, characterized by $P_{\overline{X}|Z}$, to which Eve can input Z and that outputs $\overline{X}$, such that $P_{\overline{X}Y} = P_{XY}$. Eve can successfully implement active attacks with $\overline{X}$.

Similarly, we can define Y *is simulatable by* Z *with respect to* X, which is denoted by $\mathrm{sim}_X(Z \to Y)$.

It was proved that if either $\mathrm{sim}_Y(Z \to X)$ or $\mathrm{sim}_X(Z \to Y)$, secret key agreement against active adversaries is not possible (see Theorem 6.3 in [64]).

The following theorem, which was proved in [64], shows that $\mathrm{sim}_Y(Z \to X)$ and $\mathrm{sim}_X(Z \to Y)$ hold if secret key agreement against passive adversaries is impossible.

Theorem 4.2.2 *Let* P_{XYZ} *be a probability distribution such that* $I(X; Y \downarrow Z) = 0$. *Then* $\mathrm{sim}_Y(Z \to X)$ *and* $\mathrm{sim}_X(Z \to Y)$ *hold.*

The question how to determine whether $\mathrm{sim}_Y(Z \to X)$ or $\mathrm{sim}_X(Z \to Y)$ holds and how to find a channel $P_{\overline{X}|Z}$ (or $P_{\overline{Y}|Z}$) for a given distribution P_{XYZ} such that $P_{\overline{X}Y} = P_{XY}$ (or $P_{X\overline{Y}} = P_{XY}$) was studied in [64].

4.3 Authentication with Correlated Strings between Alice and Bob

Let us deal with the problem of authentication for the satellite scenario introduced in Chapter 1. We still use the terminology of authentication codes in this context, for example, source states, encoding rules, messages, impersonation attack, substitution attack, etc. Suppose that Alice, Bob, and Eve get random variables $\underline{X} = (X_1, X_2, \ldots, X_N)$, $\underline{Y} = (Y_1, Y_2, \ldots, Y_N)$, and $\underline{Z} = (Z_1, Z_2, \ldots, Z_N)$ respectively, through three independent binary symmetric channels with bit error probabilities p_A, p_B, and p_E. Since X_i, Y_i, and Z_i have the same joint probability distribution for $i = 1, 2, \ldots, N$, we use X, Y, and Z to denote the binary random variables that Alice Bob and Eve get from the BSC channels. Then

$$P_{\underline{XYZ}}[x_1, x_2, \ldots, x_N, y_1, y_2, \ldots, y_N, z_1, z_2, \ldots, z_N] = \prod_{i=1}^{N} P_{XYZ}[x_i, y_i, z_i].$$

The simulatability condition is related to p_A, p_B, and p_E.

Lemma 4.3.1 *In the satellite scenario,* $p_A \geq p_E$ *is equivalent to* $\mathrm{sim}_{\underline{Y}}(\underline{Z} \to \underline{X})$.

Proof: The bit error rate between Alice's string $\underline{X}$ and Bob's string $\underline{Y}$ is given by $\epsilon_{AB} = p_A + p_B - 2p_Ap_B$. The bit error rate between Eve's string $\underline{Z}$ and Bob's string

$\underline{Y}$ is given by $\epsilon_{BE} = p_B + p_E - 2p_Bp_E$. We get $\epsilon_{AB} \geq \epsilon_{BE}$ from $p_A \geq p_E$. Then Eve can always input Z to a binary symmetric channel with bit error probability $P_{\overline{X}|Z} = (\epsilon_{AB} - \epsilon_{BE})/(1 - 2\epsilon_{BE})$, and get $\overline{X}$ such that

$$P_{\overline{X}|Y} = P_{X|Y} = \begin{cases} \epsilon_{AB} & \text{if } \overline{X} \neq Y \\ 1 - \epsilon_{AB} & \text{if } \overline{X} = Y \end{cases} .$$

It follows that $P_{\overline{X}\underline{Y}} = P_{\underline{X}\underline{Y}}$.

Reversing the order in the above proof, we can derive $p_A \geq p_E$ from $\mathrm{sim}_{\underline{Y}}(\underline{Z} \rightarrow \underline{X})$. □

Meanwhile $p_A \geq p_E$ is equivalent to $I(X;Y) \leq I(Y;Z)$, hence we have the following theorem.

Theorem 4.3.2 *In the satellite scenario,* $\mathrm{sim}_{\underline{Y}}(\underline{Z} \rightarrow \underline{X})$ *is equivalent to* $I(\underline{X};\underline{Y}) \leq I(\underline{Y};\underline{Z})$.

The above theorem was also proved in [64]. It means that the simulatability condition is equivalent to Eve having an advantage, in terms of mutual information, between her and Bob over Alice and Bob (or between her and Alice over Bob and Alice).

Secret key agreement against passive adversaries is possible even if X and Y are both simulatable. This has been verified by the protocol proposed in Chapter 2 that combines advantage distillation and information reconciliation. In the case of active adversaries, however, it is necessary to assume that $p_A < p_E$ and $p_B < p_E$.

As we pointed out earlier, traditional authentication techniques cannot be employed to prevent Eve's active attacks, because there is usually no authentication key shared between Alice and Bob (generally not even a partially secret string between them). What Alice and Bob try to generate is a secret string from their correlated strings $\underline{X}$ and $\underline{Y}$ that have bit error rate $\epsilon_{AB} = p_A + p_B - 2p_Ap_B$. The assumption that $p_A < p_E$ and $p_B < p_E$ implies that the bit error rate between Eve's and Alice's strings, $\epsilon_{AE} = p_A + p_E - 2p_Ap_E$, is larger than ϵ_{AB}, and so is the bit error rate $\epsilon_{BE} = p_B + p_E - 2p_Bp_E$ between Bob and Eve. Then, $p_A < p_E$ implies that Alice shares more common bits with Bob than with Eve. Similarly, $p_B < p_E$ implies that Bob shares more common bits with Alice than with Eve. The idea of authentication is that Alice attaches part of her string $\underline{X}$ as an authenticator when she transmits a source state $\underline{s}$ to Bob over the public channel. After Bob gets the message, he creates from $\underline{Y}$ his own authenticator of $\underline{s}$, and compares it with the received authenticator. If the message is from Alice, then Bob expects a fraction about ϵ_{AB} of inconsistent bits between the two authenticators. If the message is forged by Eve, either by an impersonation attack or a substitution attack, there will be more inconsistent bits.

In case of an impersonation attack, Eve introduces a fraudulent message to Bob with her string $\underline{Z}$. Then a fraction about ϵ_{BE} of inconsistent bits is expected between Bob's calculated and received authenticators. As long as the authenticator is long enough, there will be a great gap between the number of inconsistent bits introduced

by Alice and that by Eve. Bob will detect this and thwart Eve's impersonation attack.

In case of a substitution attack, Eve intercepts a message, i.e., a source state together with its authenticator. She wants to create a new authenticator for a different source state, hoping that it will be accepted by Bob. Her best strategy is to copy those bits from the intercepted authenticator that are also contained in the new authenticator, introducing errors with probability ϵ_{AB}, and to take as guesses for the remaining bits her copies of the bits (from $\underline{Z}$), introducing errors in those bits with probability ϵ_{BE}. To thwart Eve's substitution attack, the encoding rule should be chosen such that a different source state leads to as many bits as possible for Eve to guess with her own string $\underline{Z}$.

The principles for the encoding rules are as follows. With the same encoding rule,

- a source state uniquely determines the bit positions of Alice's (Bob's) initial string that contribute to the authenticator;
- different source states result in as many different bit positions in Alice's (Bob's) initial string as possible.

Traditional authentication codes assume that the encoding rules are secret information (called the authentication key) between Alice and Bob. If the message is from the legitimate sender, the receiver never rejects it. In the context of secret key agreement, however, Alice and Bob have no common string at all. Authenticity is implemented in a different way.

- The encoding rule is public.
- The receiver may reject the sender's messages with nonzero probability.

The aim of authenticity is to enable the receiver both to accept legitimate messages and to reject fraudulent messages with high probabilities. The following questions arise.

(1) What is a good encoding rule to encode the source states into messages?

(2) How to determine the threshold of the number of inconsistent bits so that Bob accepts Alice's messages and rejects Eve's messages with given probabilities?

Given an encoding rule, suppose that a source state leads to a binary codeword, and that the binary codeword uniquely determines an authenticator such that the indices of the 1-entries of the binary codeword determine the positions of bits from Alice's (Bob's) string that make up the authenticator. Then the number of different bit positions between two authenticators can be measured by the so-called *0-1 distance*.

Definition 4.3.3 *The 0-1 distance from a codeword $\underline{c}_1$ to another codeword $\underline{c}_2$, denoted by $d(\underline{c}_1 \to \underline{c}_2)$, is defined as the number of transitions from 0 to 1 when going from $\underline{c}_1$ to $\underline{c}_2$, not counting the transitions from 1 to 0.*

The 0-1 distance of two codewords is different from the Hamming distance and it is not symmetric, i.e., $d(\underline{c}_1 \to \underline{c}_2) = d(\underline{c}_2 \to \underline{c}_1)$ does not hold in general.

Definition 4.3.4 *The minimum 0-1 distance of a code $\mathcal{C}$, denoted by $d_{0\to1}(\mathcal{C})$, is defined as the smallest value among the 0-1 distances between any two different codewords in $\mathcal{C}$, i.e.,*

$$d_{0\to1}(\mathcal{C}) = \min_{i,j,i\neq j} d(\underline{c}_i \to \underline{c}_j),$$

where $\underline{c}_i, \underline{c}_j \in \mathcal{C}$.

The minimum 0-1 distance of any conventional linear code is 0 since the zero codeword always lies in the code. The following theorem shows how to change a conventional linear code of Hamming distance d into a code with 0-1 distance d.

Theorem 4.3.5 *Every conventional linear code of length n with minimum Hamming distance d can be converted to a code of length $2n$ with minimum 0-1 distance d by replacing every bit in the original codewords by a pair of bits, namely by replacing 0 by 01 and 1 by 10.*

We omit the proof since it is obvious. The code obtained from the linear code with the method of Theorem 4.3.5 is called a *0-1 code*, and its codewords are called *0-1 codewords*. Note that the number of 1's is equal to that of 0's in any 0-1 codeword.

Alice and Bob can agree on a linear code. For any source state, a corresponding codeword is obtained according to the encoding rule of the linear code. Subsequently, the codeword can be changed into a 0-1 codeword. Then both Alice and Bob are able to construct the authenticator for the source state by taking together some bits from her or his initial string, where the positions of the bits are determined by the indices of 1-entries of the 0-1 codeword.

More precisely, the authentication scheme can be described as follows. Let Alice be the sender and Bob the receiver.

Scheme 4.3.6 Prerequisite:

(1) Alice, Bob, and Eve obtain initial strings $\underline{x} = (x_1, x_2, \dots)$, $\underline{y} = (y_1, y_2, \dots)$ and $\underline{z} = (z_1, z_2, \dots)$ from the satellite's binary, random broadcast, over independent binary symmetric channels with bit error probabilities p_A, p_B, and p_E, respectively, where $p_A < p_E$ and $p_B < p_E$. Let $\epsilon_{AB} = p_A + p_B - 2p_Ap_B$, $\epsilon_{BE} = p_E + p_B - 2p_Ep_B$, and $\epsilon_{AE} = p_A + p_E - 2p_Ap_E$.

(2) Alice and Bob agree on an $[n, k, d]$ linear code and r, a threshold value. This can be accomplished over an authentic channel, for instance a voice channel (but it is not practical to use such an authentic channel to authenticate a large amount of data, which is why this authentication scheme is developed).

Authentication scheme:

(1) Let $\underline{s} = (s_1, s_2, \dots, s_k)$ be the k-bit source state. Suppose that $G_{k\times n}$ is the generation matrix of the $[n, k, d]$ linear code.

(a) *Alice determines the linear codeword* $\underline{c} = \underline{s} \cdot G_{k \times n}$. *She converts the linear codeword to the corresponding 0-1 codeword* $\underline{c'} = (c_1', c_2', \ldots, c_{2n}')$.

(b) *Suppose* c_{i_1}', c_{i_2}', ..., *and* c_{i_n}' *equal to 1 in* $\underline{c'}$. *Alice constructs the authenticator* $(x_{i_1}, x_{i_2}, \ldots, x_{i_n})$ *for* $\underline{s}$.

(c) *Alice sends the message* $(s_1, s_2, \ldots, s_k)||(x_{i_1}, x_{i_2}, \ldots, x_{i_n})$ *to Bob.*

(2) *After Bob gets the message, he determines* $\underline{c'}$ *from the source state in the same way and gets his own authenticator* $(y_{i_1}, y_{i_2}, \ldots, y_{i_n})$.

(a) *Bob determines the number of inconsistent bits between the received and created authenticators. Let*

$$v = \sum_{j=1}^{n} |x_{i_j} - y_{i_j}|.$$

(b) *Bob only accepts the received message when* v *is less than the threshold value* r, *otherwise he rejects it.*

(3) *Alice and Bob discard* $(x_1, x_2, \ldots, x_{2n})$ *and* $(y_1, y_2, \ldots, y_{2n})$ *respectively from their strings.*

The following parameters are defined to evaluate the performance of Scheme 4.3.6.

Definition 4.3.7 *The code rate of an authentication scheme, denoted by* R, *is defined as the number of bits that can be authenticated by Alice and Bob with one bit of their initial, correlated strings.*

For traditional authentication codes, the code rate R is determined by the length of the source states divided by that of the encoding rules (authentication keys).

Definition 4.3.8 *The failure probability of an authentication scheme, denoted by* P_F, *is defined as the probability that the receiver rejects a legitimate message from the sender.*

The failure probability of any traditional authentication code is 0.

Definition 4.3.9 *The deception probability of an authentication scheme, denoted by* P_D, *is defined as the probability that the adversary successfully carries out an active attack, i.e., the receiver accepts a fraudulent message sent by the adversary.*

The deception probability is generally determined by the impersonation probability and the substitution probability, i.e., $P_D = \max\{P_I, P_S\}$.

The performance of the above authentication scheme is summarized in the following theorem.

Theorem 4.3.10 *For $s, s' > 1$, let $r = n \cdot \epsilon_{AB} + s \cdot \sqrt{n \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})}$. Suppose that the $[n, k, d]$ code satisfies*

$$s' \cdot \frac{1 - \epsilon_{AB} - \epsilon_{BE}}{2} \leq \sqrt{n \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})} \tag{4.1}$$

and

$$\begin{aligned} &s' \cdot \sqrt{(n-d) \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB}) + d \cdot \epsilon_{BE} \cdot (1 - \epsilon_{BE})} + s \cdot \sqrt{n \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})} \\ &\leq d \cdot (\epsilon_{BE} - \epsilon_{AB}). \end{aligned} \tag{4.2}$$

Then Scheme 4.3.6 has code rate

$$R = \frac{k}{2n},$$

failure probability

$$P_F \leq 1/s^2,$$

and deception probability

$$P_D \leq 1/s'^2.$$

Proof: In Scheme 4.3.6, when the message comes from Alice, the subscripts of 1-entries, $(i_1, i_2, \ldots, i_n)$, of the 0-1 codeword determined by Bob should be consistent with those determined by Alice. Let the random variable V denote the number of different bits between $(x_{i_1}, x_{i_2}, \ldots, x_{i_n})$ and $(y_{i_1}, y_{i_2}, \ldots, y_{i_n})$. Then the expected value and the standard deviation of V are

$$\mu = n \cdot \epsilon_{AB}$$

and

$$\sigma = \sqrt{n \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})}.$$

In the authentication scheme, Bob accepts a message only when

$$V < r = n \cdot \epsilon_{AB} + s \cdot \sqrt{n \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})}.$$

From Chebyshev's inequality, we have

$$\Pr\left[|V - \mu| < s \cdot \sigma\right] > 1 - \frac{\sigma^2}{(s \cdot \sigma)^2}.$$

It follows that

$$\Pr\left[|V - n \cdot \epsilon_{AB}| < s \cdot \sqrt{n \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})}\right] > 1 - \frac{1}{s^2},$$

hence

$$\Pr\left[V < n \cdot \epsilon_{AB} + s \cdot \sqrt{n \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})}\right] > 1 - \frac{1}{s^2}.$$

It indicates that Bob accepts Alice's messages with probability at least $1 - 1/s^2$.

When Eve has intercepted a message from Alice, as we pointed out earlier, her best strategy for creating a new authenticator for a different message (hoping that it will be accepted by Bob), is to copy those bits from the received authenticator that are also contained in the new authenticator. This introduces an inconsistent bit in one of those bits with probability ϵ_{AB}. She also takes as guesses for the remaining bits her copies of the bits (from $\underline{Z}$), introducing an error in one of those bits with probability ϵ_{BE}. The probability of successful deception is hence determined by the number l of bits that Eve must guess and the length n of the authenticator. Let V' denote the number of different bits between the two authenticators, one is forged by Eve and the other generated by Bob. The expected value and the standard deviation of V' are

$$\mu' = (n-l) \cdot \epsilon_{AB} + l \cdot \epsilon_{BE}$$

and

$$\sigma' = \sqrt{(n-l) \cdot \epsilon_{AB} \cdot (1-\epsilon_{AB}) + l \cdot \epsilon_{BE} \cdot (1-\epsilon_{BE})}.$$

In fact, the 0-1 distance from a codeword $\underline{c'_1}$ to another codeword $\underline{c'_2}$ is the number of bits that Eve must guess when trying to convert the authenticator corresponding to $\underline{c'_1}$ into another authenticator corresponding to $\underline{c'_2}$. Since the minimum 0-1 distance of the 0-1 code generated from the $[n, k, d]$ code is d, it is obvious that $l \geq d$ holds. Therefore Eve must guess at least d bits to forge the authenticator. Let

$$\begin{aligned} f(x) =& (\epsilon_{BE} - \epsilon_{AB})x - s \cdot \sqrt{n \cdot \epsilon_{AB} \cdot (1-\epsilon_{AB})} \\ & - s' \cdot \sqrt{(n-x) \cdot \epsilon_{AB} \cdot (1-\epsilon_{AB}) + x \cdot \epsilon_{BE} \cdot (1-\epsilon_{BE})}. \end{aligned}$$

It is easy to test that when

$$s' \cdot \frac{1 - \epsilon_{AB} - \epsilon_{BE}}{2} \leq \sqrt{n \cdot \epsilon_{AB} \cdot (1-\epsilon_{AB})}$$

holds, the derivative function $f'(x) \geq 0$ for $x > 0$. If (4.2) holds, it follows that

$$\begin{aligned} & s' \cdot \sqrt{(n-l) \cdot \epsilon_{AB} \cdot (1-\epsilon_{AB}) + l \cdot \epsilon_{BE} \cdot (1-\epsilon_{BE})} + s \cdot \sqrt{n \cdot \epsilon_{AB} \cdot (1-\epsilon_{AB})} \\ & \leq l \cdot (\epsilon_{BE} - \epsilon_{AB}), \end{aligned}$$

which implies that $\mu' - \mu \geq s \cdot \sigma + s' \cdot \sigma'$.

From Chebyshev's inequality we know

$$\Pr\left[|V' - \mu'| < s' \cdot \sigma'\right] > 1 - \frac{\sigma'^2}{(s'\sigma')^2},$$

hence

$$\Pr\left[V' > \mu' - s' \cdot \sigma'\right] > 1 - \frac{1}{s'^2}.$$

Since $\mu' - s' \cdot \sigma' \geq \mu + s \cdot \sigma$, we have

$$\Pr\left[V' > \mu + s \cdot \sigma\right] = \Pr\left[V' > r\right] > 1 - \frac{1}{s'^2},$$

which means that Bob rejects fraudulent messages with a probability at least $1 - 1/{s'}^2$. □

Remark. When Bob is the sender and Alice the receiver, ϵ_{BE} is replaced by ϵ_{AE} in Theorem 4.3.10 and the corresponding proof.

4.4 Authentication with the Extended Reed-Solomon Codes

In this section, we show with an example that as long as $p_A < p_E$ and $p_B < p_E$, it is possible to find a proper linear code to implement Scheme 4.3.6 with the required authentication performance.

We take as an example the $[N, K, d]$ extended Reed-Solomon code over a finite field $GF(2^m)$. The code has a length of $N = 2^m$. As a binary code, it has a length of $n = m \cdot 2^m$, and the source state consists of $k = m \cdot K$ bits. The minimum Hamming distance of the $[N, K, d]$ extended Reed-Solomon code satisfies $d = N - K + 1$. Consequently, the corresponding 0-1 code has also a 0-1 distance of d. Substituting $d = 2^m - K + 1$ in (4.1) and (4.2) yields

$$s' \cdot \frac{1 - \epsilon_{AB} - \epsilon_{BE}}{2} \leq \sqrt{m \cdot 2^m \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})} \tag{4.3}$$

and

$$\begin{aligned} &s' \cdot \sqrt{((m-1) \cdot 2^m + K - 1) \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB}) + (2^m - K + 1) \cdot \epsilon_{BE} \cdot (1 - \epsilon_{BE})} \\ &+ s \cdot \sqrt{m \cdot 2^m \cdot \epsilon_{AB} \cdot (1 - \epsilon_{AB})} \leq (2^m - K + 1) \cdot (\epsilon_{BE} - \epsilon_{AB}). \end{aligned} \tag{4.4}$$

For any $k \in \mathbb{N}$, $s, s' > 1$, there exists an integer m_0 such that for any $m \geq m_0$, the above two inequalities with $K = \lceil k/m \rceil$ hold. In other words, as long as the code length $n = m \cdot 2^m$ is large enough, Alice can always send a k-bit source state to Bob with $P_F \leq 1/s^2$ and $P_D \leq 1/{s'}^2$. Further Alice and Bob have to sacrifice $2n$ bits to authenticate the k bits of the source state. So, $R = k/(2n) = K/(2N)$ is the *code rate* of the authentication scheme. Alice and Bob would like to use an authentication scheme with high code rate R and low values of P_F and P_D. But for a fixed k, there is a trade-off between n, P_F and P_D. To ensure P_F and P_D to be small, n has to be large enough, which implies a low code rate R.

Let $p_A = 0.01$, $p_B = 0.02$, and $p_E = 0.3$. When the code rate of the authentication scheme is fixed, an upper bound on the minimal length of the authenticator, say n_0, is determined for $P_D = P_F \leq 1/s^2$, where $s = 2, 3, 4, 5$. The dotted lines in Figure 4.1 illustrate how P_D and P_F behave as a function of n_0 for code rate R=1/4, 1/16. The solid lines show the bounds for $p_A = 0.1$ instead of 0.01.

Scheme 4.3.6 with the extended RS codes is able to achieve any given code rate R, $0 < R < 1$, and failure and deception probabilities P_D and P_F ($0 < P_D, P_F < 1$) as long as n (k as well) is long enough.

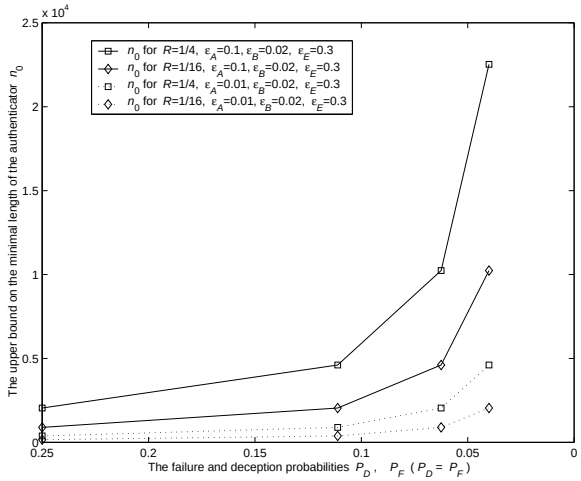

Figure 4.1: The upper bound on the minimal length of the authenticator as a function of the probability P_F $(=P_D)$ when code rate $R = k/(2n_0) = 1/4$ and $1/16$ for the case when $p_A = 0.01$, $p_B = 0.02$, and $p_E = 0.3$ and the case when $p_A = 0.1$, $p_B = 0.02$, and $p_E = 0.3$

4.5 Further Analysis of the Authentication Scheme

In Section 4.3, we analyzed the performance of Scheme 4.3.6 by means of the Chebyshev's inequality. Using other inequalities, the performance of the authentication scheme may be proved to be better than what Theorem 4.3.10 claims. In this section, we will analyze Scheme 4.3.6 with the inequality introduced in Lemma 1.2.2. The corresponding result is shown in the following theorem.

Theorem 4.5.1 *Suppose that an $[n, k, d]$ linear code is employed in Scheme 4.3.6 by Alice and Bob, and Bob chooses r, $r \in \mathbb{N}$, as the threshold (he accepts when the number of inconsistent bits between the received and calculated authenticators is less than r and rejects otherwise). Let $\alpha = ((1-\epsilon_{AB})r)/(\epsilon_{AB}(n-r))$, $a = (n-r)\epsilon_{AB}\epsilon_{BE}$, $b = n\epsilon_{AB}(1-\epsilon_{BE}) - r(\epsilon_{AB}+\epsilon_{BE}-2\epsilon_{AB}\epsilon_{BE}) + d(\epsilon_{BE}-\epsilon_{AB})$, $c = -r(1-\epsilon_{AB})(1-\epsilon_{BE})$, and $\beta = (-b+\sqrt{b^2-4ac})/(2a)$. If n, k, d, and r are chosen such that*

$$n\epsilon_{AB} + d(\epsilon_{BE} - \epsilon_{AB}) \geq r \geq n\epsilon_{AB},$$

then the authentication scheme has a code rate of

$$R = \frac{k}{2n},$$

Bob's failure probability satisfies

$$P_F \leq \frac{(\epsilon_{AB}\cdot\alpha + 1 - \epsilon_{AB})^n}{\alpha^r}, \tag{4.5}$$

and Eve's deception probability satisfies

$$P_D \leq \frac{(\epsilon_{AB} \cdot \beta + 1 - \epsilon_{AB})^{n-d} (\epsilon_{BE} \cdot \beta + 1 - \epsilon_{BE})^d}{\beta^r}. \tag{4.6}$$

Proof: Let the random variable V denote the number of different bits between the authenticator Alice sent and the one Bob calculated for some source state. It is binomially distributed with n and ϵ_{AB} as parameters. Let $\alpha = ((1-\epsilon_{AB}) \cdot r)/(\epsilon_{AB} \cdot (n-r))$. According to Theorem 1.2.3,

$$\Pr[V \geq r] \leq \frac{(\epsilon_{AB} \cdot \alpha + 1 - \epsilon_{AB})^n}{\alpha^r}$$

holds when

$$r \geq n \cdot \epsilon_{AB}. \tag{4.7}$$

For Eve's active attack, we only have to analyze her best strategy for the substitution attack. Let the random variable V' denote the number of different bits between the authenticator Eve forged and the one Bob calculated for some source state. Suppose that Eve has to guess l bits with her own string, and can copy $n-l$ bits from the intercepted authenticator. We know that $l \geq d$. The number of different bits in the l bits is binomially distributed with parameters l and ϵ_{BE}, while the number of different bits in the other $n-l$ bits is binomially distributed with parameters $n-l$ and ϵ_{AB}. From Lemma 1.2.2, it follows that

$$\Pr[V' \leq r] \leq E[e^{(r-V')t}]$$

for any $t \geq 0$.

$$\begin{aligned} E[e^{(r-V')t}] =& \sum_{i=0}^{n-l} \binom{n-l}{i} {\epsilon_{AB}}^i (1-\epsilon_{AB})^{n-l-i} \sum_{j=0}^{l} \binom{l}{j} {\epsilon_{BE}}^j (1-\epsilon_{BE})^{l-j} \cdot e^{(r-(i+j))t} \\ =& e^{rt} (\epsilon_{AB} \cdot e^{-t} + 1 - \epsilon_{AB})^{n-l} (\epsilon_{BE} \cdot e^{-t} + 1 - \epsilon_{BE})^l. \end{aligned} \tag{4.8}$$

Let $\beta = e^{-t}$, then

$$\begin{aligned} \Pr[V' \leq r] \leq& \frac{(\epsilon_{AB} \cdot \beta + 1 - \epsilon_{AB})^{n-l} (\epsilon_{BE} \cdot \beta + 1 - \epsilon_{BE})^l}{\beta^r} \\ \leq& \frac{(\epsilon_{AB} \cdot \beta + 1 - \epsilon_{AB})^{n-d} (\epsilon_{BE} \cdot \beta + 1 - \epsilon_{BE})^d}{\beta^r}. \end{aligned} \tag{4.9}$$

The last inequality comes from the fact that $\epsilon_{AB} < \epsilon_{BE}$ and $\beta \leq 1$.

Define

$$g(\beta) = \frac{(\epsilon_{AB} \cdot \beta + 1 - \epsilon_{AB})^{n-d} (\epsilon_{BE} \cdot \beta + 1 - \epsilon_{BE})^d}{\beta^r}.$$

The corresponding derivative is given by

$$g'(\beta) = (a \cdot \beta^2 + b \cdot \beta + c)\,(\epsilon_{AB} \cdot \beta + 1 - \epsilon_{AB})^{n-d-1}\,(\epsilon_{BE} \cdot \beta + 1 - \epsilon_{BE})^{d-1} \cdot \beta^{-r-1},$$

where $a = (n-r) \cdot \epsilon_{AB} \cdot \epsilon_{BE}$, $b = n \cdot \epsilon_{AB} \cdot (1-\epsilon_{BE}) - r \cdot (\epsilon_{AB} + \epsilon_{BE} - 2\epsilon_{AB}\epsilon_{BE}) + d \cdot (\epsilon_{BE} - \epsilon_{AB})$, and $c = -r \cdot (1-\epsilon_{AB}) \cdot (1-\epsilon_{BE})$. Since $a > 0$ and $b^2 - 4ac > 0$, we have

$$g'(\beta) < 0, \text{ if } (-b - \sqrt{b^2-4ac})/(2a) < \beta < (-b + \sqrt{b^2-4ac})/(2a),$$
$$g'(\beta) > 0, \text{ if } \beta > (-b + \sqrt{b^2-4ac})/(2a).$$

Consequently, $g(\beta)$ achieves its minimum for $\beta = (-b + \sqrt{b^2-4ac})/(2a)$ since $g'(\beta) = 0$. To guarantee that $t \geq 0$ (i.e., $\beta = e^{-t} \leq 1$),

$$\frac{-b + \sqrt{b^2-4ac}}{2a} \leq 1 \tag{4.10}$$

is required.

We shall now show that Alice and Bob can always find an integer r, $r \geq n \cdot \epsilon_{AB}$, which satisfies (4.10), to achieve (4.5) and (4.6). The reason is the following. Statement (4.10) is equivalent to $-c \leq a + b$, i.e.,

$$r \cdot (1-\epsilon_{AB}) \cdot (1-\epsilon_{BE}) \leq$$

$$(n-r) \cdot \epsilon_{AB} \cdot \epsilon_{BE} + n \cdot \epsilon_{AB} \cdot (1-\epsilon_{BE}) - r \cdot (\epsilon_{AB} + \epsilon_{BE} - 2\epsilon_{AB}\epsilon_{BE}) + d \cdot (\epsilon_{BE} - \epsilon_{AB}).$$

Simplify the above formula, we get $r \leq n \cdot \epsilon_{AB} + d \cdot (\epsilon_{BE} - \epsilon_{AB})$.

□

From (4.5) and (4.6), we see that the upper bound for P_F decreases with increasing n, and the upper bound for P_D decreases with increasing n and d. The parameter r also plays a role on P_F and P_D. For fixed n and d, an increase of r decreases P_F but increases P_D, while a decrease of r decreases P_D but increases P_F. On the other hand, when n and d are large enough, Alice and Bob can always choose a threshold r such that P_F and P_D are less than any given small value.

As for the example in Section 4.4, Table 4.1 tabulates P_F and P_D determined by Theorem 4.3.10 with those determined by Theorem 4.5.1 for different n_0 when $R = k/(2n_0) = 1/4$ and $p_A = 0.1$, $p_B = 0.02$, and $p_E = 0.3$. We can see that the failure and deception probabilities determined by Theorem 4.5.1 are generally smaller than those determined by Theorem 4.3.10, especially when n is large.

4.6 Conclusion

In this chapter, a so-called simulatability condition was introduced. The simulatability condition characterizes a class of probability distributions P_{XYZ}, where X, Y, and Z are correlated random variables obtained by Alice, Bob, and Eve in the initialization phase respectively. If X is simulatable by Z with respect to Y (or Y is

$n_0$bits	$P_F = P_D$ (Theorem 4.3.10)	P_F (Theorem 4.5.1)	P_D (Theorem 4.5.1)
2048	0.2500	0.1605	0.0617
4608	0.1111	0.0166	0.0066
10240	0.0625	$6.64 \cdot 10^{-4}$	$1.76 \cdot 10^{-5}$
22528	0.0400	$9.72 \cdot 10^{-6}$	$6.47 \cdot 10^{-12}$

Table 4.1: Upper bounds on P_F and P_D determined by Theorem 4.3.10 and Theorem 4.5.1 for the given lengths of the authenticator with the code rate $R = k/(2n_0) = 1/4$ and $p_A = 0.1$, $p_B = 0.02$, and $p_E = 0.3$

simulatable by Z with respect to X), Bob (Alice) cannot tell the difference between Alice's (Bob's) messages and Eve's. Thus secret key agreement is vulnerable to Eve's active attacks since Alice and Bob need some public discussion over a public channel during the communication phase.

We considered the special scenario, when Alice, Bob, and Eve obtain correlated information through independent binary symmetric channels from a random, binary output of a satellite. In such a scenario, the statement that X is simulatable by Z with respect to Y is equivalent to the scenario that Eve's channel is better than Alice's. To consider secret key agreement against active adversaries, it is necessary to assume that Eve's channel is noisier than both Alice's and Bob's. We showed that an authentication scheme based on coding theory can always be implemented to accomplish the required authentication performance, using Alice's and Bob's correlated strings.

The authentication scheme based on the extended RS codes was given as an example. It showed that the receiver's failure probability, and the adversary's deception probability, are related to the length of the authenticator, the code rate of the authentication scheme, and the bit error rates of the binary symmetric channels. Although a linear code satisfying the required performance can always be found for the authentication scheme, the authentication scheme might not be very practical since the authenticator may be too long and the code rate too low. So how to design a practical authentication scheme with high code rate and moderate authenticator length deserves further research. However, the existence of the authentication scheme described in this chapter implies that the advantage between Alice and Bob can be exploited to provide authenticity for the messages exchanged over the public channel. Recall that in Chapter 2 the authenticity of the public channel leads to the advantage between Alice and Bob over Eve. Therefore, the authenticity of the public channel and the advantage between Alice and Bob over Eve are convertible.

Chapter 5

Evaluating Eve's Information in a Quantum Transmission

5.1 Introduction

In this chapter, we consider the quantum key agreement method described by Bennett et al. in [2], and derive a probabilistic upper bound on the information obtained by Eve during the raw quantum transmission.

Bennett et al. considered quantum key agreement between two legal users Alice and Bob. The physical carriers of information are quantum mechanical (e.g. photons) and therefore called quantum bits, or simply qubits. It is well known that measurements on a quantum mechanical system destroy all information about the state of the system. We will explain in Section 5.2 how this can be used to detect eavesdropping.

More precisely, the quantum transmission protocol works as follows: Alice sends a sequence of polarized photons to Bob. Each photon is randomly selected to be in one of the following four canonically polarized states: horizontal, vertical (rectilinear), left-circular, or right-circular. For each received photon Bob chooses randomly whether to measure the photon's rectilinear or circular polarization. Then Bob announces publicly the sequence of measurements he has made. Alice replies publicly the used sequence of polarizations (rectilinear or circular). Alice and Bob then discard all bit positions for which Bob's measurements did not match Alice's reply and all bit positions for which Bob did not detect any photon at all. Heisenberg's uncertainty principle implies that measuring a photon's rectilinear polarization will randomize its circular polarization, and vice versa. When Bob detects all photons and there is no eavesdropping, his bit error rate will typically be 1/4 before discarding the unmatched qubits. The reason is the following. With probability 1/2, Bob chooses a correct basis to measure a photon, hence getting the qubit sent by Alice; with probability 1/2, he chooses a wrong basis and gets a randomly valued qubit. After discarding the unmatched qubits, his bit error rate should be 0. The

polarizations of the remaining photons are interpreted as bit 0 for horizontal or left-circular, and bit 1 for vertical and right-circular. These steps together are called a raw quantum transmission session.

In the practical implementation of a quantum transmission, it is very hard for the current quantum facility to produce a sequence of pure single photons. What is used is a sequence of light pulses instead. The expected number of photons per light pulse is μ, which is sufficiently smaller than 1.

Alice and Bob want to distill a secret key of a certain minimal length from each raw quantum transmission session despite eavesdropping by Eve. They remove all raw quantum transmission sessions for which it is not possible to distill a secret key of sufficient length. It is Eve's intention that Alice and Bob generate a secret key about which Eve gets some information. Hence, Eve's activities should not prevent Alice and Bob from generating a common key. Alice and Bob have to solve the following problems:

(1) Find with high probability the location of all, say e, errors after agreement of the bases. This can be done by means of information reconciliation, see [2, 7, 76, 73, 77, 38] and Chapter 2. During the information reconciliation, Alice and Bob reveal bits to Eve. The exact number b of bits revealed to Eve can be computed by Alice and Bob. For the reconciliation protocol proposed in Chapter 2, b is just the number of the rows of the parity check matrix constructed during the protocol.

(2) Given the number of qubits n that Alice and Bob get after a raw quantum transmission, the number of errors e between Alice's and Bob's qubits, and the density of the light pulses μ used in the quantum transmission, compute an upper bound $l(e, n, \mu)$, which holds with high probability, on the number of bits of information obtained by Eve during the raw quantum transmission session. This problem has been studied in [20, 21], and will be further discussed in this chapter (see Section 5.3).

(3) Generate a secret key given that Eve obtained at most $l(e, n, \mu) + b$ bits of information, and compute the secret information leaking to Eve. This problem can be solved by means of privacy amplification, see [9, 3, 4, 5, 10] and Chapter 3.

This chapter is mainly based on [22].

5.2 Eve's Strategies

Eve's possible strategies presented in [2] are introduced in this section. We will only consider the qubits which Bob measures in the right bases, since they are discarded otherwise.

5.2.1 The Intercept/Resend Strategy

When Alice transmits a message to Bob over a channel, it is quite possible that Eve sits in the middle, intercept and read the message, then resend it to Bob. This is known as the *Intercept/Resend* strategy. With a classical channel, Eve knows the exact information about the intercepted message. After Eve resends the message, Bob also gets the exact information about the message, but Alice and Bob are not able to detect whether the message has been known to Eve.

However, things are different with a quantum channel. More precisely, with the Intercept/Resend strategy during the raw quantum transmission session, Eve intercepts selected pulses and measures them in bases of her choice. Then Eve fabricates a pulse of the same polarization as she detected (the correct information on the state that Alice had prepared may have disappeared!), which she sends to Bob in the interception basis.

When Eve uses canonical bases to intercept/resend pulses, she will get deterministic information about the qubits. For each qubit that Eve intercepts, with probability 1/2 she will have chosen the right basis to measure (which agrees with the basis in which Alice polarized the pulse) and gets exact information about the qubit. She also has probability 1/2 to measure the pulse in the wrong basis. In that case, she gets random information. Therefore, her bit error probability is 1/4. As for Bob, he will also get the exact information about the qubit in the first case. In the second case, Bob interprets intercepted photons measured by Eve in the wrong basis correctly with probability 1/2. That means for each of these intercepted qubits, Bob also gets an error with probability 1/4. Eve can also use a basis, which is different from the interception basis, to fabricate a light pulse and send to Bob. But this will result in an error in Bob's qubit with a probability larger than 1/4, see [2] for details. Therefore, we assume that Eve uses the interception basis to resend the light pulse to Bob. After Alice publishes the correct bases, both Bob and Eve expect to know the polarization information on a fraction 3/4 of the pulses Eve intercepted. There is a significant difference between the channel from Alice to Bob and the channel from Alice to Eve; Eve knows which intercepted photons were measured by her in correct bases, where Bob does not know which intercepted photons were measured by Eve in correct bases. This extra knowledge of Eve gives Eve information about whether her interpretation of an intercepted photon is correct or completely arbitrary (that is independent of Alice's interpretation). We conclude that Eve's channel is superior to Bob's channel (which is not surprising since Eve is an active eavesdropper).

If Eve uses the Breidbart basis, which can be loosely described as a basis midway between the rectilinear and circular bases, her bit error probability will be minimized to $(2-\sqrt{2})/4 \approx 0.15$ (see [2] for details). But after the correct bases are announced, Eve's information will not change from a probabilistic one into a deterministic one, as is the case with a canonical measurement. In the case of Eve using a Breidbart basis, however, the final secret key generated in the subsequent privacy amplification phase is measured by Rényi entropy according to Theorem 3.2.2. More precisely, about $-\log(0.85^2+0.15^2) \approx 0.415$ bits can be distilled from each intercepted qubit. Correspondingly, we can conclude that Eve gets no more than 0.585 deterministic

bits from each intercepted pulse. On the other hand, Bob's bit error probability achieves its minimum of 1/4 as long as Eve resends the intercepted pulses in the Breidbart basis.

In the above analysis, we assume a perfect quantum setting. However, practical quantum facilities inevitably introduce noise. Therefore, errors in Bob's qubits may also come from imperfection of the quantum facilities, such as photon detector's malfunction or system misalignment. Nevertheless, Alice and Bob may conservatively suppose that all errors are caused by Eve's intercept/resend attack.

5.2.2 The Beamsplitting Strategy

In practical implementations of quantum transmissions, the light pulses Alice transmits over the quantum channel are not pure single-photon states. This fact enables Eve to implement a so-called *Beamsplitting* attack. She can set a partly-silvered mirror in the quantum channel. With the mirror, a fraction f of the original beam's intensity is diverted to herself, and the remaining fraction $1 - f$ of the intensity of Alice's original light pulse goes to Bob without any disturbance. If Bob can detect this fraction of the light pulse, then no new error is introduced in this specific qubit (recall we only consider those pulses that Bob has measured with the right bases). Eve's successful splitting of a light pulse means that both Bob and Eve detect at least one photon. If every light pulse sent by Alice consists of several photons, it is very likely that Eve is able to split it successfully. Then the quantum channel becomes a classical channel, i.e., Eve may read the message over the channel without detection, since no errors are introduced by Beamsplitting. However, if the intensity of the light pulses is very low, the probability of a successful splitting by Eve will be very small, as we will show below.

Generally, the distribution of the number of photons per light pulse is given by a Poisson distribution with mean μ ($\mu < 1$). The probability that the number of photons in a light pulse is x is

$$\Pr[x] = \frac{e^{-\mu}\mu^x}{x!}.$$

The probability of a successful splitting by Eve is determined in the following computation.

The probability that both Bob and Eve detect at least 1 photon per light pulse is given by

$$\sum_{x=2}^{\infty} \Pr[x] \sum_{i=1}^{x-1} \binom{x}{i} f^i (1-f)^{x-i} = \sum_{x=2}^{\infty} \Pr[x] \cdot [1 - f^x - (1-f)^x], \tag{5.1}$$

and the probability that Bob detects at least 1 photon is determined by

$$\sum_{x=1}^{\infty} \Pr[x] \sum_{i=1}^{x} \binom{x}{i} f^{x-i} (1-f)^i = \sum_{x=1}^{\infty} \Pr[x] \cdot [1 - f^x]. \tag{5.2}$$

The probability that Eve successfully splits a light pulse that contributes the qubits, on which Alice and Bob will agree after a raw quantum transmission session, can be determined by dividing (5.1) by (5.2), which is

$$\begin{aligned}\Pr\left[\text{A successful Beamsplitting}\right] &= \frac{\sum_{x=2}^{\infty}\Pr\left[x\right]\cdot\left[1-f^x-(1-f)^x\right]}{\sum_{x=1}^{\infty}\Pr\left[x\right]\cdot\left[1-f^x\right]} \\ &= \frac{\sum_{x=2}^{\infty}\frac{\mu^x}{x!}-\sum_{x=2}^{\infty}\frac{(f\mu)^x}{x!}-\sum_{x=2}^{\infty}\frac{[(1-f)\mu]^x}{x!}}{\sum_{x=1}^{\infty}\frac{\mu^x}{x!}-\sum_{x=1}^{\infty}\frac{(f\mu)^x}{x!}} \\ &\approx \frac{\frac{\mu^2}{2}-\frac{(f\mu)^2}{2}-\frac{(1-f)^2\mu^2}{2}}{\mu-f\mu} = f\mu.\end{aligned} \tag{5.3}$$

Unless we assume that Eve can store photons, Eve will still have probability of 1/2 of guessing the right basis for a successfully split pulse.

5.2.3 Combination of Intercept/Resend and Beamsplitting

Eve can carry out both the Intercept/Resend and Beamsplitting attacks during the raw quantum transmission session. We call this strategy the *Combination strategy*. If Eve selects some pulses for an Intercept/Resend attack and tries to split every light pulse, she will learn some qubits twice through intercept/resending a successfully split pulse. In order to count these qubits only once, we make different assumptions depending on the kind of bases Eve uses with the Combination strategy.

Suppose that Eve intercepts/resends K pulses among the n pulses. When Eve uses canonical bases, let l denote the number of pulses measured by her with correct bases (then she will learn the l qubits through intercept/resending). In this case, we shall assume that Eve splits all the $n-K$ pulses that she does not intercept/resend, and that she also splits the $K-l$ pulses that she does intercept/resend but about which she gets random information (because of the wrong bases). In other words, Eve tries to split the $n-l$ pulses. In case of Eve using the Breidbart measurement, we shall assume that Eve only splits the $n-K$ pulses that she does not intercept/resend.

5.3 Probabilistic Analysis

The analysis given here corresponds to the situation after a *raw quantum transmission* and a public *reconciliation* procedure. This refers to the situation where Alice has sent her qubits to Bob, and Alice and Bob publicly agree to the qubits that Bob received and measured with the correct bases. Some qubits may have been intercepted/resent or split by Eve. The number of errors in Bob's qubits, due to Eve's intercept/resending, is also known after a reconciliation procedure. In the ideal quantum setting, there will be no discrepancy between Alice's and Bob's qubits when Eve's interception is absent. Let n represent the number of qubits Alice and Bob get after a transmission session. Let the random variable K be the number of

qubits that have been intercepted and resent to Bob by Eve. The number of errors in Bob's qubits is denoted by E. Hence the number of correct qubits of Bob from those that were intercepted by Eve is given by $C = K - E$. Let M denote the number of light pulses that Eve successfully splits among the n light pulses sent by Alice. Let $L^{(\cdot)}$ be the number of bits of deterministic information that have leaked to Eve for some specific strategy and measurement. For instance,

- $L^{(IC)}$ for the Intercept/Resend strategy with canonical measurement;
- $L^{(IB)}$ for the Intercept/Resend strategy with the Breidbart measurement;
- $L^{(BC)}$ for the Beamsplitting strategy with canonical measurement;
- $L^{(BB)}$ for the Beamsplitting strategy with the Breidbart measurement;
- $L^{(CC)}$ for the Combination strategy with canonical measurement;
- $L^{(CB)}$ for the Combination strategy with the Breidbart measurement.

5.3.1 Analysis for the Intercept/Resend Strategy

We first talk about Eve's canonical measurement in the Intercept/Resend strategy. The number of correct qubits of Bob from those that were intercepted by Eve is given by $C = K - E$ (if there is no noise and no eavesdropping then after the raw quantum transmission, $E = 0$). A straightforward deterministic upper bound on the number of qubits that Eve obtained when the number of errors is known is given by

$$L^{(IC)} \leq C = K - E \leq n - E. \tag{5.4}$$

To obtain a probabilistic upper bound we make a statistical analysis of the related random variables.

Analysis for the random variable $L^{(IC)}$

The following statements can be proved easily.

Lemma 5.3.1 *For $0 \leq l \leq c \leq k$,*
i)

$$Pr\left[C = c | L^{(IC)} = l, K = k\right] = \binom{k-l}{c-l} \left(\frac{1}{2}\right)^{k-l},$$

ii)

$$Pr\left[L^{(IC)} = l | K = k\right] = \binom{k}{l} \left(\frac{1}{2}\right)^{k},$$

iii)

$$Pr[C = c | K = k] = \binom{k}{c} \left(\frac{3}{4}\right)^{c} \left(\frac{1}{4}\right)^{k-c},$$

iv)

$$Pr\left[L^{(IC)} = l | C = c, K = k\right] = \binom{c}{l}\left(\frac{1}{3}\right)^{c-l}\left(\frac{2}{3}\right)^{l}.$$

v)

$$Pr\left[L^{(IC)} = l | C = c\right] = \frac{1}{3}\frac{c}{c-l} Pr\left[L^{(IC)} = l | C = c-1\right].$$

Moreover $K \rightarrow C \rightarrow L^{(IC)}$ *form a Markov chain since* $Pr\left[L^{(IC)} = l | C = c, K = k\right] = Pr\left[L^{(IC)} = l | C = c\right]$.

Proof: Statements i), ii) and iii) follow immediately from some combinatorial considerations. Statement iv) follows from the following computation:

$$\begin{aligned}
\Pr\left[L^{(IC)} = l | C = c, K = k\right] &= \frac{\Pr\left[L^{(IC)} = l | K = k\right]\Pr\left[C = c | L^{(IC)} = l, K = k\right]}{\Pr\left[C = c | K = k\right]} \\
&= \frac{\binom{k}{l}\left(\frac{1}{2}\right)^{k}\binom{k-l}{c-l}\left(\frac{1}{2}\right)^{k-l}}{\binom{k}{c}\left(\frac{3}{4}\right)^{c}\left(\frac{1}{4}\right)^{k-c}} \\
&= \binom{c}{l}\left(\frac{1}{3}\right)^{c-l}\left(\frac{2}{3}\right)^{l}. \qquad (5.5)
\end{aligned}$$

From iv), it follows that $\Pr\left[L^{(IC)} = l | C = c, K = k\right]$ is independent of the numbers of qubits that were intercepted by Eve. Hence,

$$\Pr\left[L^{(IC)} = l | C = c, K = k\right] = \Pr\left[L^{(IC)} = l | C = c\right], \qquad (5.6)$$

which shows that this process generates a Markov chain.

Combining (5.5) and (5.6), Statement v) follows. □

The above lemma implies that C and $L^{(IC)}$ are binomially distributed, more precisely, $C \sim Binomial(k, 3/4)$ and $L^{(IC)} \sim Binomial(c, 2/3)$. From Statement v) of the lemma, it follows that:

$$\Pr\left[L^{(IC)} = l | C = c\right] \geq \Pr\left[L^{(IC)} = l | C = c-1\right] \quad \text{if} \quad l \geq \frac{2}{3}c. \qquad (5.7)$$

We define the following notion:

Definition 5.3.2 *For* $\epsilon \geq 0$,

$$l'_{\epsilon}(c) = \min\left\{l \in \mathbb{N} \mid \frac{2}{3}c \leq l \leq c,\ Pr\left[L^{(IC)} \geq l | C = c\right] \leq \epsilon\right\}. \qquad (5.8)$$

If the value of c, i.e., the number of correct qubits in Bob's n qubits, is known, then with probability at least $1 - \epsilon$, the information leaked to Eve is not more than $l'_{\epsilon}(c)$ bits.

Lemma 5.3.3 *The function $l'_\epsilon(c)$ is non-decreasing in c.*

Proof: Put $l'_\epsilon(c) = l$ for some l satisfying $2c/3 \leq l \leq c$ and $\Pr\left[L^{(IC)} \geq l | C = c\right] \leq \epsilon$. From Relation (5.7) it follows that:

$$\epsilon \geq \Pr\left[L^{(IC)} \geq l | C = c\right] \geq \Pr\left[L^{(IC)} \geq l | C = c - 1\right].$$

This implies that $l'_\epsilon(c-1) \leq l$.

□

Analysis of the random variable E

Similar to what we did above for the random variable $L^{(IC)}$, we can also analyze the number of errors E. By using $C = K - E$ we obtain

$$\Pr[E = e | K = k] = \Pr[C = k - e | K = k] = \left(\begin{array}{c} k \\ e \end{array} \right) \left(\frac{1}{4}\right)^e \left(\frac{3}{4}\right)^{k-e}$$

from Lemma 5.3.1 iii). This leads to the next lemma.

Lemma 5.3.4

$$Pr[E = e | K = k+1] = \frac{3}{4} \frac{k+1}{k+1-e} Pr[E = e | K = k].$$

From lemma 5.3.4, it follows immediately that:

$$\Pr[E = e | K = k+1] \leq \Pr[E = e | K = k] \text{ if } e \leq \frac{k+1}{4}. \tag{5.9}$$

Similar to $l'_\epsilon(c)$, we now define the notion $e'_\epsilon(k)$.

Definition 5.3.5 *For $\epsilon \geq 0$,*

$$e'_\epsilon(k) = \max\left\{ e \in \mathbb{N} \;\middle|\; 0 \leq e \leq \frac{k}{4}, Pr[E < e | K = k] \leq \epsilon \right\}.$$

Again, we remark that $e'_\epsilon(k)$, is well-defined since $\Pr[E < 0 | K = k] = 0$, which ensures that a maximum exists.

When the number of light pulses that Eve intercepts and resends is known, with probability at least $1 - \epsilon$, there are at most $e'_\epsilon(k)$ errors in Bob's n qubits.

Lemma 5.3.6 *The function $e'_\epsilon(k)$ is non-decreasing in k.*

Proof: Put $e'_\epsilon(k) = e$ for some e satisfying $0 \leq e \leq \frac{k}{4}$ and $\Pr[E < e | K = k] \leq \epsilon$. From relation (5.9) it follows that

$$\Pr[E < e | K = k+1] \leq \Pr[E < e | K = k] \leq \epsilon.$$

Therefore, $e'_\epsilon(k+1) \geq e$. □

Analysis of the random variables K and C

Finally, we analyze the quantities K and C.

Definition 5.3.7 *For $\epsilon > 0$, we define:*

$$k'_\epsilon(e) = \min\{k \in \mathbb{N} \mid k \geq 4e, Pr[E < e|K = k] \leq \epsilon\}.$$

This is well-defined since $\Pr[E < e|K = k] \to 0$ (when k tends to ∞) as follows from Lemma 5.3.4.

If the number of errors in Bob's n qubits is known, Eve intercepts and resends at least $k'_\epsilon(e)$ light pulses with probability at least $1 - \epsilon$.

Lemma 5.3.8 *When $e \in \mathbb{N}$ satisfies $k'_\epsilon(e) \leq k$, then*

$$e \leq e'_\epsilon(k).$$

Proof: Write $\hat{k} = k'_\epsilon(e)$. It follows from the definition of $k'_\epsilon(e)$ that $e \leq \frac{\hat{k}}{4}$ and $\Pr\left[E < e|K = \hat{k}\right] \leq \epsilon$. Therefore one has that $e'_\epsilon(\hat{k}) \geq e$ by the definition of $e'_\epsilon(\hat{k})$. Applying $\hat{k} \leq k$ and Lemma 5.3.6, it follows that:

$$e \leq e'_\epsilon(\hat{k}) \leq e'_\epsilon(k).$$

□

The relationship $C = K - E$ suggests the next definition.

Definition 5.3.9 *For $\epsilon > 0$, we define $c'_\epsilon(e) = k'_\epsilon(e) - e$.*

Probabilistic upper bound for the Intercept/Resend strategy

The following theorem gives a probabilistic upper bound on the number of bits leaked to Eve when the number of errors is given by E in Eve's canonical measurement with the Intercept/Resend strategy.

Theorem 5.3.10 *Define $\gamma'(e) = Pr[E < e|K = n]$. Then*

$$Pr\left[L^{(IC)} < \min_{\alpha, 0 \leq \alpha \leq \epsilon} l'_{\epsilon-\alpha}(c'_\alpha(E))\right] \geq 1 - \epsilon \tag{5.10}$$

for $\epsilon \geq \gamma'(E)$ and

$$Pr\left[L^{(IC)} < l'_\epsilon(n - E)\right] \geq 1 - \epsilon \tag{5.11}$$

for $0 < \epsilon < \gamma'(E)$.

Proof: The proof follows from the following computations.

For $\beta \geq 0$,

$$\begin{aligned}
&\Pr\left[L^{(IC)} \geq l'_\beta(c'_\alpha(E))\right] \\
&= \sum_{k,c} \Pr\left[L^{(IC)} \geq l'_\beta(c'_\alpha(k-c))|K=k, C=c\right] \Pr\left[K=k, C=c\right] \\
&= \sum_{k,c} \Pr\left[L^{(IC)} \geq l'_\beta(c'_\alpha(k-c))|C=c\right] \Pr\left[K=k, C=c\right], \qquad (5.12)
\end{aligned}$$

where (5.12) follows from the Markov property of Lemma 5.3.1. It is easy to see that

$$\begin{aligned}
\Pr\left[L^{(IC)} \geq l'_\beta(c'_\alpha(E))\right] \quad \leq \quad & \sum_{k,c,c'_\alpha(k-c)\geq c} \Pr\left[L^{(IC)} \geq l'_\beta(c)|C=c\right] \Pr\left[K=k, C=c\right] \\
& + \sum_{k,c,c'_\alpha(k-c)<c} \Pr\left[K=k, C=c\right]. \qquad (5.13)
\end{aligned}$$

Further,

$$\Pr\left[L^{(IC)} \geq l'_\beta(c'_\alpha(E))\right] \leq \beta + \sum_{k,c,c'_\alpha(k-c)<c} \Pr\left[K=k, C=c\right] \qquad (5.14)$$

is a direct consequence of Definition 5.3.2. And

$$\Pr\left[L^{(IC)} \geq l'_\beta(c'_\alpha(E))\right] \leq \beta + \Pr\left[c'_\alpha(E) < C\right] = \beta + \Pr\left[k'_\alpha(E) < K\right] \qquad (5.15)$$

follows from the relation $C = K - E$ and Definition 5.3.9.

For $\alpha > 0$,

$$\begin{aligned}
\Pr\left[k'_\alpha(E) < K\right] &\leq \Pr\left[E \leq e'_\alpha(K)\right]) \qquad (5.16) \\
&= \sum_k \Pr\left[E < e'_\alpha(k)|K=k\right] \Pr\left[K=k\right] \\
&\leq \alpha, \qquad (5.17)
\end{aligned}$$

where (5.16) follows from Lemma 5.3.8 and (5.17) follows from Definition 5.3.5. By substituting $\beta = \epsilon - \alpha$, we obtain (5.10).

Relation (5.9) implies that $k'_\alpha(e)$ is a decreasing function of α for fixed e. It follows that $k'_\alpha(e) \geq n$ for $\alpha \leq \gamma'(e) = \Pr\left[E < e|K=n\right]$. For such α, $\Pr\left[K > k'_\alpha(E)\right] \leq \Pr\left[K > n\right] = 0$. This observation together with (5.12)-(5.15) proves (5.11) with $\beta = \epsilon$ and $c'_\alpha(E)$ replaced by $n - E$.

When $\epsilon \geq \gamma'(E)$, it is easy to see that

$$\min_{\alpha, 0\leq\alpha\leq\epsilon} l'_{\epsilon-\alpha}(c'_\alpha(E)) \leq l'_\epsilon(c'_0(E)) = l'_\epsilon(n-E).$$

On the other hand, if $0 < \epsilon < \gamma'(E)$, we have $\alpha < \gamma'(e)$. Then $l'_\beta(c'_\alpha(E)) = l'_\beta(c'_0(E)) = l'_\beta(n-E) \geq l'_\epsilon(n-E)$ holds. The last inequality follows from the fact that $l'_\epsilon(c)$ is a decreasing function of ϵ. We conclude that $\min_{\alpha,0\leq\alpha\leq\epsilon} l'_{\epsilon-\alpha}(c'_\alpha(E)) \geq l'_\epsilon(n-E)$.

Notice that $l'_\epsilon(n-E) \leq n-E$. In combination with (5.11) we deduce that $\Pr\left[L^{(IC)} \leq n-E\right] \geq 1-\epsilon$ which is in accordance with (5.4) which states that $L^{(IC)} \leq n-E$ for all instances of random variables L and E. □

Efficient evaluation of the probabilistic upper bound

When c and l are large, the evaluation of $\Pr\left[L^{(IC)} \geq l | C=c\right]$ becomes inefficient and loses precision due to the limited register length in computers. The same problem happens to $\Pr\left[E \leq e | K=k\right]$. Since $L^{(IC)} \sim Binomial(C, 2/3)$ and $E \sim Binomial(K, 1/4)$, we can use Theorem 1.2.3 to efficiently estimate an upper bound for $\Pr\left[L^{(IC)} \geq l | C=c\right]$ and $\Pr\left[E \leq e | K=k\right]$. More precisely, if $l \geq \frac{2}{3}c$,

$$\Pr\left[L^{(IC)} \geq l \mid C=c\right] \leq \frac{\left(\frac{2}{3}x+\frac{1}{3}\right)^c}{x^l}$$

for $x = \frac{l}{2(c-l)}$. If $e \leq \frac{k}{4}$

$$\Pr\left[E \leq e \mid K=k\right] \leq \frac{\left(\frac{1}{4}x+\frac{3}{4}\right)^k}{x^e}$$

for $x = \frac{3e}{k-e}$.

Now we can redefine $l'_\epsilon(c)$, $e'_\epsilon(k)$, $k'_\epsilon(e)$, $c'_\epsilon(e)$ and $\gamma'(e)$ by

$$l_\epsilon(c) = \min\left\{l \in \mathbb{N} \mid \frac{2}{3}c \leq l < c,\ \frac{\left(\frac{2}{3}x+\frac{1}{3}\right)^c}{x^l} \leq \epsilon,\ x = \frac{l}{2(c-l)}\right\}; \tag{5.18}$$

$$e_\epsilon(k) = \max\left\{e \in \mathbb{N} \mid 0 < e \leq \frac{k}{4},\ \frac{\left(\frac{1}{4}x+\frac{3}{4}\right)^k}{x^e} \leq \epsilon,\ x = \frac{3e}{k-e}\right\}; \tag{5.19}$$

$$k_\epsilon(e) = \min\left\{k \in \mathbb{N} \mid\ k \geq 4e,\ \frac{\left(\frac{1}{4}x+\frac{3}{4}\right)^k}{x^e} \leq \epsilon,\ x = \frac{3e}{k-e}\right\}; \tag{5.20}$$

$$c_\epsilon(e) = k_\epsilon(e) - e; \tag{5.21}$$

$$\gamma(e) = (y/4+3/4)^n / y^e, \tag{5.22}$$

where $y = 3e/(n-e)$.

Lemma 5.3.3 and Lemma 5.3.6 still hold for the new definitions of $l'_\epsilon(c)$ and $e'_\epsilon(k)$.

Lemma 5.3.11 *The function $l_\epsilon(c)$ is non-decreasing in c.*

Proof: Let $f(c,l) = \ln\left((2x/3+1/3)^c/x^l\right)$ for $x = l/(2(c-l))$, i.e.,

$$f(c,l) = c\ln\frac{c}{3(c-l)} - l\ln\frac{l}{2(c-l)}.$$

Then

$$\frac{\partial f(c,l)}{\partial c} = \ln\frac{c}{3(c-l)} \geq 0, \tag{5.23}$$

and

$$\frac{\partial f(c,l)}{\partial l} = -\ln\frac{l}{2(c-l)} \leq 0 \tag{5.24}$$

for $2c/3 \leq l < c$. Let $\ln\epsilon = f(c,l)$ and $\ln\epsilon' = f(c+1,l)$. Then $l_\epsilon(c) = l_{\epsilon'}(c+1) = l$ according to (5.18), and $\epsilon' \geq \epsilon$ according to (5.23). Let $l' = l_\epsilon(c+1)$. We get

$$f(c+1,l') \leq \ln\epsilon = f(c,l) \leq f(c+1,l).$$

Then $l' \geq l$, i.e., $l_\epsilon(c+1) \geq l_\epsilon(c)$, follows from (5.24). □

Lemma 5.3.12 *The function $e_\epsilon(k)$ is non-decreasing in k.*

Proof: Let $f(k,e) = \ln\left((x/4+3/4)^k/x^e\right)$ for $x = 3e/(k-e)$, i.e.,

$$f(k,e) = k\ln\frac{3k}{4(k-e)} - e\ln\frac{3e}{k-e}.$$

Then

$$\frac{\partial f(k,e)}{\partial k} = \ln\frac{3k}{4(k-e)} \leq 0, \tag{5.25}$$

and

$$\frac{\partial f(k,e)}{\partial e} = -\ln\frac{3e}{k-e} \geq 0 \tag{5.26}$$

for $0 < e \leq \frac{k}{4}$.

Let $\ln\epsilon = f(k,e)$ and $\ln\epsilon' = f(k-1,e)$. Then $e_\epsilon(k) = e_{\epsilon'}(k-1) = e$ according to (5.19), and $\epsilon' \geq \epsilon$ according to (5.25). Let $e' = e_\epsilon(k-1)$. We get

$$f(k-1,e') \leq \ln\epsilon = f(k,e) \leq f(k-1,e).$$

Then $e' \leq e$, i.e., $e_\epsilon(k-1) \leq e_\epsilon(k)$, follows from (5.26). □

With Lemma 5.3.11 and Lemma 5.3.12, we may conclude that Lemma 5.3.8 and Theorem 5.3.10 also hold for the new definitions of $l'_\epsilon(c)$, $e'_\epsilon(k)$, $k'_\epsilon(e)$, $c'_\epsilon(e)$ and $\gamma'(e)$.

Let us consider two examples from [2] and compare their analysis with ours. Recall from the introduction that b denotes the number of (parity check) bits exchanged publicly and revealed to Eve during the reconciliation phase to locate the

e errors between Alice's and Bob's n qubits. In our proposal for an AD/IR protocol as described in Chapter 2, all e errors are located with very high probability. Furthermore b bits are indicated (including all the bits which were found to be in error) which if discarded from the n original qubits give $n-b$ remaining bits about which Eve can not obtain additional knowledge by using the b bits revealed to her. Alice and Bob will discard these b bits and they will compute a statistical upper bound on l, the number of bits revealed to Eve during the raw quantum transmission. In the privacy amplification phase they compress the $n-b$ remaining bits to $n-b-l-s$ bits, were s is a security parameter. According to Theorem 3.2.1, which was proved in [5], with probability at most $2^{-s}/\ln 2$ Eve knows one deterministic bit of the final secret key after the privacy amplification phase. For example, by taking $s=21$ this probability is $3.121 \cdot 10^{-7}$.

In [2] the exact number of errors e is not known after the reconciliation phase because during the reconciliation phase bits may already get discarded. Alice and Bob know that with very high probability all errors are among the b discarded bits. However, they have only been able to compute the locations of some of the bits in error. The locations only gives a lower bound on e. This is because they do not require reconciliation of the bits which were discarded during their protocol. Such bits have a non-neglectable probability to be in error. Continuing the reconciliation phase, Alice and Bob are incapable of finding out the exact number of errors. However, with simple interpolation techniques they obtain a very precise estimate of the number of errors e (in the examples $e+1$ and $e+2$ resp.). Therefore we assume in the examples of [2] that e is known exactly. This modifies their examples slightly.

In [2] the following method is used to obtain a statistical upper bound on l. Let $p=e/n$ be the bit error probability between Alice's and Bob's qubits. The amount of information leaked to Eve during a raw quantum transmission session exceeds $l=\rho+5\sigma$ bits with very low probability, where $\rho=(4/\sqrt{2})pn$ represents the expected value and $\sigma=\sqrt{(4+2\sqrt{2})pn}$ represents the standard deviation. An upper bound on the probability of exceeding $l=\rho+5\sigma$ is unknown. Note that in a Gaussian distribution the probability of exceeding a 5σ statistical deviation equals $2.866 \cdot 10^{-7}$. For this reason we assume in the examples that the upper bound $l=\rho+5\sigma$ is exceeded with probability $2.866 \cdot 10^{-7}$.

Example 5.3.13 *Suppose that after a raw quantum transmission session without Eve's eavesdropping, Alice and Bob got* $n=2000$ *qubits, and removed all* $e=79$ *errors by publicly exchanging* $b=621$ *parity bits during an information reconciliation protocol. With* $e=79$, *Alice and Bob estimated that* $p=0.039$. *Let* $s=21$.

By using the method of [2] at most $l=340$ *bits of information are leaked to Eve with probability at least approximately* $1-2.866 \cdot 10^{-7}$. *The length of the secret key constructed during the privacy amplification phase equals* $2000-621-340-21=1018$ *bits. With probability at most* **approximately** $2.866 \cdot 10^{-7}+3.121 \cdot 10^{-7}=5.987 \cdot 10^{-7}$ *Eve obtains one or more bits of deterministic information about the final key.*

By applying Theorem 5.3.10 and taking $\epsilon = 2.866 \cdot 10^{-7}$ *we obtain* $\gamma(79) = 8.34 \cdot 10^{-144}$. *We have* $\gamma(79) < \epsilon$, *so the upper bound is given by*

$$\min_{\alpha, 0<\alpha\leq\epsilon} l_{\epsilon-\alpha}(c_\alpha(79)) = 352.$$

The minimum is achieved for $\alpha = 1.92 \cdot 10^{-7}$. *Compared to the method in [2], our result proves that if we construct a final secret key of length* $2000 - 621 - 352 - 21 = 1006$ *bits, then with probability at most* $5.987 \cdot 10^{-7}$ *Eve obtains one or more bits of deterministic information. Note that this is not an approximation. We notice that by taking* $\epsilon = 1.309 \cdot 10^{-6}$ *and applying Theorem 5.3.10, a final secret key of length* 1018 *bits can be generated such that the probability that Eve obtains one or more bits of deterministic information is* **guaranteed** *to be at most* $1.621 \cdot 10^{-6} \approx 2.7 \cdot 5.987 \cdot 10^{-7}$.

Example 5.3.14 *Suppose that after a raw quantum transmission session with Eve's substantial eavesdropping, Alice and Bob got* $n = 2000$ *qubits, and removed all* $e = 160$ *errors by publicly exchanging* $b = 993$ *parity bits during an information reconciliation protocol. With* $e = 160$, *Alice and Bob estimated that* $p = 0.080$. *Let* $s = 21$.

By using the method of [2] at most $l = 618$ *bits of information are leaked to Eve with probability at least approximately* $1 - 2.866 \cdot 10^{-7}$. *The length of the final secret key equals* $2000 - 993 - 618 - 21 = 368$ *bits. Eve obtains one or more bits of deterministic information on the final secret with probability* **approximately** *at most* $5.987 \cdot 10^{-7}$.

By applying Theorem 5.3.10 and taking $\epsilon = 2.866 \cdot 10^{-7}$ *we obtain* $\gamma(160) = 8.301 \cdot 10^{-85}$. *We have* $\gamma(160) < \epsilon$, *so the upper bound is determined by*

$$\min_{\alpha, 0<\alpha\leq\epsilon} l_{\epsilon-\alpha}(c_\alpha(160)) = 579.$$

The minimum is achieved for $\alpha = 2.12 \cdot 10^{-7}$. *The length of the final secret key equals* $2000 - 993 - 579 - 21 = 407$ *bits. With probability* **guaranteed** *to be at most* $5.987 \cdot 10^{-7}$ *Eve has obtained one or more bits of deterministic information.*

When using the Breidbart measurement in the Intercept/Resend strategy, the amount of information Eve obtains is only related to K, the number of pulses she intercepts and resends. More precisely, $0.585K$ bits of deterministic information is supposed to leak to Eve. The probabilistic upper bound is given in the following theorem.

Theorem 5.3.15 *Let* $\gamma(e)$ *be defined as in (5.22). Define* $l'_\epsilon(E) = 0.585 \cdot k_\epsilon(E)$. *Eve's Breidbart measurement in the Intercept/Resend strategy gives*

$$Pr\left[L^{(IB)} < l'_\epsilon(E)\right] \geq 1 - \epsilon, \tag{5.27}$$

for $\epsilon > \gamma(E)$, *otherwise a deterministic upper bound is given by*

$$L^{(IB)} \leq 0.585n.$$

Proof: When $\epsilon > \gamma(E)$, we have $k_\epsilon(E) < n$. Statement (5.27) follows from $E \sim Binomial(K, 1/4)$, Definition 5.3.9 and Theorem 1.2.3.

If $\epsilon \leq \gamma(E)$, then $k_\epsilon(E) \geq n$ follows. So $L^{(IB)}$ has as deterministic upper bound $0.585n$. □

In subsequent examples, we will always take the settings of Examples 5.3.13 or 5.3.14, i.e., $n = 2000, b = 621, e = 79$ for no eavesdropping or $n = 2000, b = 993, e = 160$ with significant eavesdropping.

Example 5.3.16 $n = 2000, b = 621, e = 79$. *Then* $\gamma(79) = 8.34 \cdot 10^{-144}$, *and*

$$l'_\epsilon(79) = 305$$

with $\epsilon = 2.866 \cdot 10^{-7}$. *The length of the final secret key equals* $2000 - 621 - 305 - 21 = 1053$ *bits. With probability at most* $5.987 \cdot 10^{-7}$ *Eve obtains one or more bits of deterministic information, using the Breidbart measurement in the Intercept/Resend strategy.*

Example 5.3.17 $n = 2000, b = 993, e = 160$. *Then* $\gamma(160) = 8.301 \cdot 10^{-85}$, *and*

$$l'_\epsilon(160) = 536$$

with $\epsilon = 2.866 \cdot 10^{-7}$. *The length of the final secret key equals* $2000 - 993 - 536 - 21 = 450$ *bits. With probability at most* $5.987 \cdot 10^{-7}$ *Eve obtains one or more bits of deterministic information, using the Breidbart measurement in the Intercept/Resend strategy.*

For $\epsilon = 2.866 \cdot 10^{-7}$, Figure 5.1 shows the upper bounds for $L^{(IC)}$ and $L^{(IB)}$ as a function of e.

5.3.2 Analysis for the Beamsplitting Strategy

Suppose that μ ($\mu < 1$) is the expected number of photons per light pulse. It is also approximately the probability that a pulse would be detected by a perfectly efficient detector. If Eve diverts a fraction f of the original beam's intensity to herself, letting the remainder pass undisturbed to Bob, Eve will succeed in splitting with probability approximately $f\mu$, as shown in Section 5.2. Since Beamsplitting introduces no new errors, we will assume that Eve tries to split every light pulse sent by Alice. Eve's splitting reduces the intensity reaching Bob by a factor $1 - f$. However, Eve may supplement the fraction f she splits by a trick proposed in [2], resulting in an effective splitting ratio of about $(f + 1)/2$. Nevertheless, we conservatively assume that Eve with probability approximately μ splits a light pulse. Let M denote the number of light pulses that Eve successfully splits among the n qubits, then $M \sim Binomial(n, \mu)$. Let $L^{(BC)}$ and $L^{(BB)}$ be the amount of deterministic information Eve gets from the canonical resp. Breidbard measurement with Beamsplitting. It is easy to see that $L^{(BC)} \sim Binomial(n, \mu/2)$ for the canonical measurement and $L^{(BB)} = 0.585M$ for the Breidbart measurement.

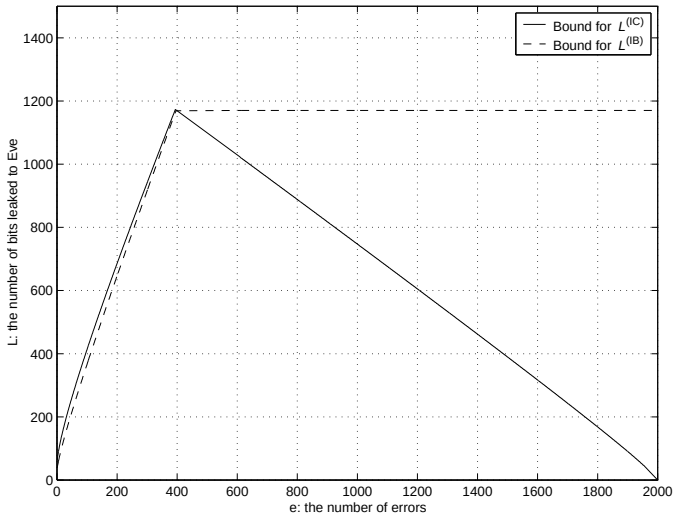

Figure 5.1: Probabilistic upper bounds for $L^{(IC)}$ and $L^{(IB)}$ as a function of e for $\epsilon = 2.866 \cdot 10^{-7}$ with Eve's Intercept/Resend strategy.

Definition 5.3.18 *For $\epsilon > 0$ we define:*

$$l_\epsilon(n,\mu) = \min\left\{ l \in \mathbb{N} \mid \frac{n\mu}{2} \leq l < n,\ \frac{(\mu x/2 + 1 - \mu/2)^n}{x^l} \leq \epsilon,\ x = \frac{l(2-\mu)}{(n-l)\mu} \right\}, \tag{5.28}$$

and

$$l'_\epsilon(n,\mu) = 0.585 \cdot \min\left\{ m \in \mathbb{N} \mid n\mu \leq m < n,\ \frac{(\mu x + 1 - \mu)^n}{x^l} \leq \epsilon,\ x = \frac{m(1-\mu)}{(n-m)\mu} \right\}. \tag{5.29}$$

Theorem 5.3.19 *For $\epsilon > 0$,*

$$Pr\left[L^{(BC)} < l_\epsilon(n,\mu) \right] \geq 1 - \epsilon,$$

for the canonical measurement, and

$$Pr\left[L^{(BB)} < l'_\epsilon(n,\mu) \right] \geq 1 - \epsilon.$$

for the Breidbard measurement in the Beamsplitting strategy.

Proof: The statements follow from the fact that

$$M \sim Binomial(n,\mu),\ L^{(BC)} \sim Binomial(n,\mu/2),\ L^{(BB)} = 0.585M,$$

Definition 5.3.18 and Theorem 1.2.3. □

Example 5.3.20 *In the experiments in [2],* $\mu = 0.12$. *Suppose that Alice and Bob get* $n = 2000$ *qubits. Let* $\epsilon = 2.886 \cdot 10^{-7}$. *If Eve splits every light pulse, with probability larger than* $1 - 2.886 \cdot 10^{-7}$, *she gets no more than*

$$l_\epsilon(2000, 0.12) = 183$$

bits with the canonical measurement, and no more than

$$l'_\epsilon(2000, 0.12) = 189$$

bits with the Breidbart measurement.

5.3.3 Analysis for the Combination Strategy

The analysis for the Intercept/Resend strategy and Beamsplitting strategy in the two previous sections are based on the following facts:

$$\begin{aligned} E \sim & Binomial(k, 1/4); \\ C \sim & Binomial(k, 3/4); \\ L^{(IC)} \sim & Binomial(c, 2/3); \\ L^{(IB)} = & 0.585K; \\ M \sim & Binomial(n, \mu); \\ L^{(BC)} \sim & Binomial(n, \mu/2); \\ L^{(BB)} = & 0.585M. \end{aligned} \tag{5.30}$$

For the Combination strategy, as pointed out in Subsection 5.2.3, assumptions are made to avoid counting twice the qubits that Eve gets through intercept/resending a successfully split pulse. For Eve's canonical measurement, we assume that Eve tries to split all pulses that she does not intercept/resend and that she also tries to split those pulses that she intercepts and resends but with the wrong bases (hence gets random information). Then the number of successfully split pulses satisfies $M^{(BC)} \sim Binomial(n - L^{(IC)}, \mu)$. For the Breidbard measurement, we assume that Eve only splits those pulses that she does not intercept/resend, so the number of successfully split pulses satisfies $M^{(BB)} \sim Binomial(n - k, \mu)$. That means equation (5.30) is replaced by the following one in case of Eve's Combination

strategy.

$$\begin{aligned} E \sim & Binomial(k, 1/4); \\ C \sim & Binomial(k, 3/4); \\ L^{(IC)} \sim & Binomial(c, 2/3); \\ L^{(IB)} = & 0.585K; \\ M^{(BC)} \sim & Binomial(n - L^{(IC)}, \mu); \\ M^{(BB)} \sim & Binomial(n - k, \mu); \\ L^{(BC)} \sim & Binomial(n - L^{(IC)}, \mu/2); \\ L^{(BB)} = & 0.585M^{(BB)}. \end{aligned} \tag{5.31}$$

Let $L^{(CC)}$ and $L^{(CB)}$ denote the amount of information Eve gets by a canonical resp. the Breidbart measurement with the Combination strategy. Then $L^{(CC)} = L^{(IC)} + L^{(BC)}$ and $L^{(CB)} = L^{(IB)} + L^{(BB)}$.

From Lemma 1.2.2, it follows that

$$\Pr\left[L^{(CC)} \geq l | C = c\right] \leq E[e^{(L^{(CC)} - l)u}]. \tag{5.32}$$

Moreover,

$$\begin{aligned} & E[e^{(L^{(CC)} - l)u}] \\ & = \sum_{j=0}^{n} \sum_{i=0}^{\min\{j,c\}} \binom{c}{i} \left(\frac{2}{3}\right)^i \left(\frac{1}{3}\right)^{c-i} \binom{n-i}{j-i} \left(\frac{\mu}{2}\right)^{l-i} \left(1 - \frac{\mu}{2}\right)^{n-l} e^{(j-l)u} \\ & = \frac{\left(\frac{2e^u/3}{\mu e^u/2 + 1 - \mu/2} + \frac{1}{3}\right)^c \left(\frac{\mu e^u}{2} + 1 - \frac{\mu}{2}\right)^n}{e^{ul}} \end{aligned}$$

Let $e^u = x$ and $f(x) = \left(\frac{2x/3}{\mu x/2 + 1 - \mu/2} + \frac{1}{3}\right)^c \left(\frac{\mu}{2}x + 1 - \frac{\mu}{2}\right)^n / x^l$. Then we get the derivative

$$\begin{aligned} f'(x) = & \left[\frac{\mu}{2}\left(\frac{\mu}{2} + 2\right)(n-l)x^2 + \left(\frac{\mu}{2} - 1\right)\left(2(l-c) - \frac{\mu}{2}(n-2l)\right)x - \left(1 - \frac{\mu}{2}\right)^2 l\right] \\ & \cdot \frac{1}{3}\left[\left(\frac{\mu}{2} + 2\right)x/3 + \left(1 - \frac{\mu}{2}\right)/3\right]^{c-1} \left(\frac{\mu}{2}x + 1 - \frac{\mu}{2}\right)^{n-c-1} x^{-l-1}. \end{aligned}$$

Let a, b be the coefficients of x^2 and x, and o be the constant in the first brackets of the above expression. It is easy to see that when $x = \left(-b + \sqrt{b^2 - 4ao}\right)/(2a)$, $f(x)$ achieves its minimum. Since $-b/2a$ is increasing if l increases, it is possible to find an l satisfying $x \geq 1$ such that

$$\Pr\left[L^{(CC)} \geq l | C = c\right] \leq \frac{\left(\frac{2x/3}{\mu x/2 + 1 - \mu/2} + \frac{1}{3}\right)^c \left(\frac{\mu}{2}x + 1 - \frac{\mu}{2}\right)^n}{x^l}. \tag{5.33}$$

Definition 5.3.21 *For $\epsilon > 0$ we define:*

$$l_\epsilon(c,n,\mu) = \min\left\{ l \in \mathbb{N} \mid \frac{\left(\frac{2x/3}{\mu x/2+1-\mu/2} + \frac{1}{3}\right)^c \left(\frac{\mu}{2}x + 1 - \frac{\mu}{2}\right)^n}{x^l} \leq \epsilon,\ x > 1 \right\},$$

where $x = \left(-b + \sqrt{b^2 - 4ao}\right)/2a$, $a = \mu(\mu/2+2)(n-l)/2$, $b = (\mu/2-1)(2(l-c) - \mu(n-2l)/2)$, *and* $o = -(1-\mu/2)^2 l$.

When Eve uses a Breidbart measurement,

$$L^{(CB)} = L^{(IB)} + L^{(BB)} = 0.585\left(K + M^{(BB)}\right).$$

From $M^{(BB)} \sim Binomial(n-k,\mu)$ and Theorem 1.2.3, the inequality

$$\Pr\left[M^{(BB)} \geq m \mid K = k\right] \leq \frac{(\mu x + 1 - \mu)^{n-k}}{x^m}. \tag{5.34}$$

follows for $m \geq (n-k)\mu$ and $x = \frac{m}{n-k-m} \cdot \frac{1-\mu}{\mu}$.

Definition 5.3.22 *For $\epsilon > 0$, define $l'_\epsilon(k,n,\mu)$ as*

$$0.585 \cdot \left\{ k + \min\left\{ m \in \mathbb{N} \mid (n-k)\mu \leq m < (n-k),\ \frac{(\mu x + 1 - \mu)^{n-k}}{x^m} \leq \epsilon \right\} \right\},$$

where $x = \frac{m}{n-k-m} \cdot \frac{1-\mu}{\mu}$.

Theorem 5.3.23 *Let $\gamma(e)$ be defined as in (5.22). For the Combination strategy, one has the following estimates.*

(1) If $\epsilon \geq \gamma(E)$, then

$$Pr\left[L^{(CC)} < \min_{\alpha, 0 \leq \alpha \leq \epsilon} l_{\epsilon-\alpha}\left(c_\alpha(E), n, \mu\right)\right] \geq 1 - \epsilon, \tag{5.35}$$

for the canonical measurement and

$$Pr\left[L^{(CB)} < \min_{\alpha, 0 \leq \alpha \leq \epsilon} l'_{\epsilon-\alpha}\left(k_\alpha(E), n, \mu\right)\right] \geq 1 - \epsilon, \tag{5.36}$$

for the Breidbart measurement;

(2) If $0 < \epsilon < \gamma(E)$, then

$$Pr\left[L^{(CC)} < l_\epsilon\left(n - E, n, \mu\right)\right] \geq 1 - \epsilon, \tag{5.37}$$

for the canonical measurement and

$$Pr\left[L^{(CB)} < l'_\epsilon\left(n, n, \mu\right)\right] \geq 1 - \epsilon, \tag{5.38}$$

for the Breidbart measurement.

We omit the proof of the above theorem since it follows the same idea of the proof for Theorem 5.3.10.

Example 5.3.24 *Let* $\epsilon = 2.886 \cdot 10^{-7}$. *When* $n = 2000$, $b = 621$, $\mu = 0.12$, *and* $e = 79$, *we have* $\gamma(79) = 8.34 \cdot 10^{-144} < \epsilon$, *and the canonical measurement in the Combination strategy gives*

$$\min_{\alpha,\ 0\leq\alpha\leq\epsilon} l_{\epsilon-\alpha}(c_\alpha(79), 2000, 0.12) = 478,$$

where the minimum is attained for $\alpha = 1.92 \cdot 10^{-7}$. *Hence 880 bits can be distilled. When* $n = 2000$, $b = 993$, $\mu = 0.12$, *and* $e = 160$, *it follows that* $\gamma(160) = 8.301 \cdot 10^{-85} < \epsilon$, *and*

$$\min_{\alpha,\ 0\leq\alpha\leq\epsilon} l_{\epsilon-\alpha}(c_\alpha(160), 2000, 0.12) = 684$$

where the minimum is attained for $\alpha = 1.92 \cdot 10^{-7}$. *Hence 302 bits can be distilled as the final secret key. With probability* **guaranteed** *to be at most* $5.987 \cdot 10^{-7}$ *Eve obtains one or more bits of deterministic information about the keys.*

Example 5.3.25 *Let* $\epsilon = 2.886 \cdot 10^{-7}$. *The Breidbart measurement in the Combination strategy gives*

$$\min_{\alpha,\ 0\leq\alpha\leq\epsilon} l'_{\epsilon-\alpha}(k_\alpha(79), 2000, 0.12) = 454$$

where the minimum is attained for $\alpha = 1.92 \cdot 10^{-7}$, *hence 904 secret bits can be distilled. It also gives*

$$\min_{\alpha,\ 0\leq\alpha\leq\epsilon} l'_{\epsilon-\alpha}(k_\alpha(160), 2000, 0.12) = 652$$

where the minimum is attained for $\alpha = 1.75 \cdot 10^{-7}$, *hence 334 secret bits can be generated from the privacy amplification phase. With probability at most* $5.987 \cdot 10^{-7}$ *Eve obtains one or more bits of deterministic information.*

For $\epsilon = 2.886 \cdot 10^{-7}$, the upper bounds on $L^{(CC)}$ and $L^{(CB)}$ as a function of e are illustrated in Figure 5.2.

Up to now, we based our analysis on the assumption that Eve has no power to store the light pulses until the announcement of the correct bases. That means that whenever Eve detects any light pulse, by interception or splitting, she has to measure it immediately to get the polarization information.

Another case was also considered in [2]. In this setting, Eve is able to store the light pulses until the correct bases are publicly announced (it should be noted that Alice and Bob may thwart this attack by delaying their announcement until Eve's stored pulses fade and lose the polarization information). For Eve's Combination strategy with storing power, the amount of information leaked to Eve during a raw quantum transmission session is estimated in [2] to be at most $l = \rho + 5\sigma$ bits, where $\rho = (\mu + (4/\sqrt{2})p)n$ is the expected value, $\sigma = \sqrt{n\mu(1-\mu)(4+2\sqrt{2})p}$ is the

standard deviation, of the number of bits of deterministic information obtained by Eve, and p is bit error rate between Alice's and Bob's qubits (this is determined by e). Let us consider the two examples from [2].

Example 5.3.26 *$n = 2000$, $e = 79$, $b = 621$. Then $p = 0.039$, and $l = 604$. The length of the final secret key determined by privacy amplification is $2000-621-604-21 = 754$ bits ($s = 21$). With probability at most $2.866 \cdot 10^{-7} + 3.121 \cdot 10^{-7} = 5.987 \cdot 10^{-7}$* **approximately** *Eve obtains one or more bits of deterministic information about the final key.*

Example 5.3.27 *$n = 2000$, $e = 160$, $b = 993$. Then $p = 0.080$, and $l = 881$. So the final length of the secret key is $2000 - 993 - 881 - 21 = 105$. With probability at most $2.866 \cdot 10^{-7} + 3.121 \cdot 10^{-7} = 5.987 \cdot 10^{-7}$* **approximately** *Eve obtains one or more bits of deterministic information about the final key.*

To analyze the probabilistic upper bound on the amount of leaked information for the case that Eve has the power to store photons until the correct bases are announced, we notice that

$$L^{(BC)} = M^{(BC)} \sim Binomial\left(n - L^{(IC)}, \mu\right)$$

in the Combination strategy. Therefore, (5.35) should be replaced by

$$\Pr\left[L^{(CC)} < \min_{\alpha,\ 0\le\alpha\le\epsilon} l''_{\epsilon-\alpha}\left(c_\alpha(E), n, \mu\right)\right] \ge 1 - \epsilon, \tag{5.39}$$

where

$$l''_\epsilon(c, n, \mu) = \min\left\{ l \in \mathbb{N} \mid \frac{\left(\frac{2x/3}{\mu x+1-\mu} + \frac{1}{3}\right)^c (\mu x + 1 - \mu)^n}{x^l} \le \epsilon,\ x > 1 \right\},$$

$a = \mu(\mu+2)(n-l)$, $b = (\mu-1)(2(l-c) - \mu(n-2l))$, $o = -(1-\mu)^2 l$, and $x = -b + \sqrt{b^2 - 4ao}/(2a)$. The results of the Breidbart measurement are not affected by Eve having power to store photons or not.

By applying our probabilistic upper bound to Example 5.3.26 and Example 5.3.27, we get the following results.

Example 5.3.28 *When Eve has power to store photons, with $\epsilon = 2.886 \cdot 10^{-7}$, her Combination strategy gives*

$$\min_{\alpha,\ 0\le\alpha\le\epsilon} l''_{\epsilon-\alpha}(c_\alpha(79), 2000, 0.12) = 595$$

where the minimum is attained for $\alpha = 1.92 \cdot 10^{-7}$, so 763 secret bits can be distilled. Further

$$\min_{\alpha,\ 0\le\alpha\le\epsilon} l''_{\epsilon-\alpha}(c_\alpha(160), 2000, 0.12) = 784$$

where the minimum is attained for $\alpha = 1.75 \cdot 10^{-7}$, so 202 secret bits can be distilled, about which Eve's information is less than 10^{-6}.

For this case, the upper bounds on $L^{(CC)}$ as a function of e are also depicted in Figure 5.2.

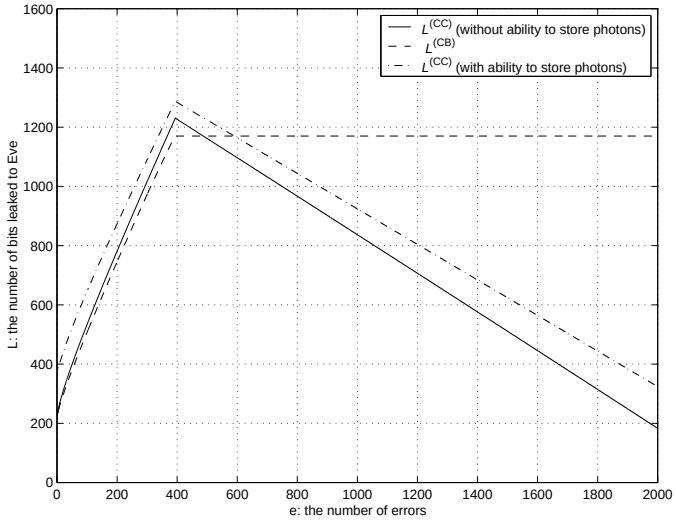

Figure 5.2: Probabilistic upper bounds for $L^{(CC)}$ and $L^{(CB)}$ as a function of e with $\mu = 0.12$ for $\epsilon = 3.121 \cdot 10^{-7}$ in case of Eve's Combination strategy.

5.4 Concluding Remarks

While generating a secret key it is better for Alice and Bob to take a probabilistic worst case scenario on the amount of eavesdropping of Eve into account. Given the number of qubits that Alice and Bob obtain after a quantum transmission session, the number of errors in Bob's qubits, and the average number of photons per light pulse, we considered upper bounds on the amount of information Eve gets from a quantum transmission session assuming different strategies for Eve, namely the Intercept/Resend strategy, the Beamplitting strategy, the combination of the two, measuring light pulses in the canonical bases or the Breidbart bases. In the situation of significant eavesdropping by Eve we improved the existing probabilistic upper bound on the number of bits revealed to Eve during the raw quantum transmission with a non-neglectable percentage (see Examples 5.3.14, 5.3.26, 5.3.27 and 5.3.28). Contrary to the existing probabilistic upper bound which gives an estimate of the probability with which the upper bound holds, we give an explicit bound on the probability with which our upper bound holds. We guarantee a specific amount of security. This is needed in practice. With an estimate one can only give an approximation of the level of security of the final key.

Bibliography

[1] R. Ahlswede and I. Csiszár, *Common randomness in information theory and cryptography - Part I: secret sharing*, IEEE transactions on Information Theory, Vol. 39, No. 4, pp. 1121-1132, 1993.

[2] C.H. Bennett, F. Bessette, G. Brassard, L. Salvail, and J. Smolin, *Experimental quantum cryptography*, Journal of Cryptology, Vol. 5, No. 1, pp. 3-28, 1992.

[3] C.H. Bennett, G. Brassard, C. Crépeau, and U.M. Maurer, *Generalized privacy amplification*, IEEE Trans. Inform. Theory, Vol. 41, No. 6, pp. 1915–1923, 1995.

[4] C.H. Bennett, G. Brassard, and J.-M.Robert, *How to reduce your enemy's information*, In: Adv. in Cryptology – Proceedings of CRYPTO '85, Lecture Notes in Comput. Sci., Vol. 218, Berlin. Germany: springer-Verlag, pp. 468-476, 1986.

[5] C.H. Bennett, G. Brassard, and J.-M.Robert, *Privacy amplification by public discussion*, SIAM J. Comput., Vol. 17, No. 2, pp. 210-229, April 1988.

[6] B. den Boer, *A simple and key-economical unconditional authentication scheme*, Journal of Computer Security, Vol. 2, No. 1, pp. 65-71, 1993

[7] G. Brassard and L. Salvail, *Secret-key reconciliation by public discusssion*, In: Adv. in Cryptology – Proceedings of EUROCRYPT '93, Lecture Notes in Comput. Sci., Vol. 765, pp. 410-423, 1994.

[8] R. Brunner, C. Cachin, U. Maurer, and C. Vonäsch, *Demonstration System for Secret Key Agreement by Public Discussion*, http://www.inf.ethz.ch/department/TI/um/.

[9] C. Cachin, *Entropy measures and unconditional security in cryptography*, Ph.D Thesis, ETH Zrich, Hartung-Gorre Verlag, Konstanz, 1997.

[10] C. Cachin and U. Maurer, *Linking information reconciliation and privacy amplification*, Journal of Cryptology, Vol. 10, No. 2, pp. 97-110, 1997.

[11] C. Cachin and U. Maurer, *Smoothing Probability Distributions and Smooth Entropy*, Proc. 1997 IEEE Symposium on Information Theory (Abstracts), 1997.

[12] C. Cachin and U. Maurer, *Unconditional Security Against Memory-Bounded Adversaries*, Advances in Cryptology - CRYPTO '97, Lecture Notes in Computer Science, Springer-Verlag, Vol. 1294, pp. 292-306, 1997.

[13] A.B. Carleial and M.E. Hellman, *A note on Wyner's wiretap channel*, IEEE Transations on Information Theory, Vol. 23, pp. 387-390, 1977.

[14] J.L. Carter and M.N. Wegman, *Universal classes of hash functions*, J. Computer and System Sci. Vol. 18, pp. 143-153, 1971.

[15] I. Csiszár and J. Körner, *Broadcast channels with confidential messages*, IEEE Transations on Information Theory, Vol. 24, pp. 339-348, 1978.

[16] M. Dietzfelbinger, J. Gil, Y. Matias, and N. Pippenger, *Polynomial hash functions are reliable*, Proc. 19th International Colloquium on Automata, Languages and Programming, LNCS 623, W. Kuich, Ed., Springer-Verlag, pp. 235-246, 1992.

[17] M. van Dijk, *Coding gain strategies for the binary symmetric broadcast channel with confidential messages*, In: Proceedings of the 16th Symposium on Information Theory in the Benelux, pp. 53–60, May 18–19, 1995.

[18] M. van Dijk, *Secret key sharing and secret key generation*, Ph.D thesis, Eindhoven university of technology, 1997.

[19] M. van Dijk and A. Koppelaar, *High rate reconciliation*, In: Proceedings of ISIT'97, p. 92, 1997.

[20] M. van Dijk and A. Koppelaar *Quantum key agreement*, Proc. of the 18th Symposium on Information Theory in the Benelux, pp. 97-104, 1997.

[21] M. van Dijk, A. Koppelaar, *Quantum key agreement*, in proceedings of ISIT'98, Cambridge, USA, p. 350, 1998.

[22] M. van Dijk, A. Koppelaar, S. Liu, and P. Tuyls, *On information leakage during quantum key distribution*, submitted to *Information Processing Letters*.

[23] M. van Dijk and H. van Tilborg, *The art of distilling*, In: Proceedings of ITW'98, pp. 158-159, 1998.

[24] R.G. Gallager, *Low density parity check codes*, IRE Trans. Info. Theory, IT-8, pp. 21-28, Jan 1962.

[25] R.G. Gallager, *Low density parity check codes*, Number 21 in research monograph series, MIT Press, Cambridge, Mass., 1963.

[26] M.J. Gander and U.M. Maurer, *On the secret-key rate of binary random variables*, In: Proceedings of ISIT'94, p. 351, 1994.

[27] P. Gemmell and M. Naor, *Codes for interactive authentication*, Advances in Cryptology-CRYPTO'93, Lecture notes in Computer Science, Vol. 773, pp. 355-367, Springer-Verlag, 1993.

[28] E. Gilbert, F.J. MacWilliams, and N. Sloane, *Codes which detect deception*, The Bell System Technical Journal, Vol. 53, No. 3, March 1974.

[29] G.A. Kabatianskii, B. Smeets, and T. Johansson, *On the cardinality of systematic authentication codes via error-correcting codes*, IEEE Trans. Inform. Theory, Vol. 42, No. 2, pp. 566–578, March 1996.

[30] D.E. Knuth, *The art of computer programming*, Addison Wesley Longman, Inc. Vol. 1, 3rd edition, p. 104.

[31] A.N. Kolmogorov, *Grundbegriffe der Wahrscheinlichkeitsrechnung*, Springer 1933,

[32] T. Johansson, G. Kabatianskii, and B. Smeets, *On the relation between A-codes and codes correcting independent errors*, Advances in Cryptology-EUROCRYPT'93, Lecture Notes in Computer Science. Vol. 765, pp. 1-11, Springer-Verlag, 1994.

[33] N.L. Johnson and S. Kotz, Discrete Distributions, John Wiley & Sons, 1969.

[34] S. Liu and Y. Wang, *Privacy amplification against active attacks with strong robustness*, Electronic letters, Vol. 35, No. 9, 1999.

[35] S. Liu and Y. Wang, *An authentication scheme over non-authentic public channel in information-theoreic secret-key agreement*, AAECC-13 Symposium, Honolulu, Hawaii, USA, Nov.14-19, Springer-Verlag, pp. 294-301, 1999.

[36] S. Liu and H. van Tilborg, *Optimizing secret key reconciliation protocol Cascade*, submitted to IEEE Journal on Selected Areas in Communications, Special issue on Design and Analysis Techniques for Security Assurance.

[37] S. Liu and H. van Tilborg, *Privacy amplification over a non-authentic public channel*, submitted to ISIT'02.

[38] S. Liu, H. van Tilborg, and M. van Dijk, *A Practical Protocol for Advantage Distillation and Information Reconciliation*, submitted to Designs Code and Cryptography.

[39] H.K. Lo, S. Popescu, and T. Spiller, *Introduction to quantum computation and information*, World Scientific, 1998.

[40] D. J.C. MacKay, *Good error-correcting codes based on very sparse matrices*, IEEE Trans. Inform. Theory, Vol. 45, No. 2, pp. 399-431, 1999.

[41] U.M. Maurer, *Perfect cryptographic security from partially independent channels*, In Proc. 23st Annual ACM Symposium on Theory of Computing, pp. 561-571, New Orleans, Louisiana, May 1991.

[42] U.M. Maurer, *Protocols for secret key agreement by public discussion based on common information*, In: Adv. in Cryptology – Proceedings of CRYPTO '92, Lecture Notes in Comput. Sci., Vol. 740, springer-Verlag, pp. 461-470, 1993.

[43] U.M. Maurer, *Secret key agreement by public discussion from common information*, IEEE Trans. Inform. Theory, Vol. 39, pp. 733–742, May 1993.

[44] U.M. Maurer, *The Role of Information Theory in Cryptography*, Proc. of 4th IMA Conference on Cryptography and Coding, The Institute of Mathematics and its Applications, Southend-on-Sea, England, pp. 49-71, 1993.

[45] U.M. Maurer, *The strong secret-key rate of discrete random triples*, Communication and Cryptography - Two Sides of One Tapestry, Kluwer Academic Publishers, pp. 271-285, 1994.

[46] U.M. Maurer, *New Information-Theoretic Bounds in Authentication Theory*, Proc. 1995 IEEE International Symposium on Information Theory (Abstracts), p. 12, 1995.

[47] U.M. Maurer, *A Unified and Generalized Treatment of Authentication Theory*, Proc. 13th Symp. on Theoretical Aspects of Computer Science (STACS'96), Lecture Notes in Computer Science, Springer-Verlag, Vol. 1046, pp. 387-398, 1996.

[48] U.M. Maurer, *Information-theoretically secure secret-key agreement by NOT authenticated public discussion*, Advances in Cryptology-EUROCRYPT'97, Lecture Notes in Computer Science. Vol. 1233, pp. 209-225, Springer-Verlag, 1997.

[49] U.M. Maurer, *Information-Theoretic Cryptography*, Advances in Cryptology - CRYPTO '99, Lecture Notes in Computer Science, Springer-Verlag, Vol. 1666, pp. 47-64, 1999.

[50] U.M. Maurer, *Authentication Theory and Hypothesis Testing*, IEEE Transaction on Information Theory, Vol. 46, No. 4, pp. 1350-1356, 2000.

[51] U.M. Maurer and S. Wolf, *Towards characterizing when information-theoretic secret key agreement is possible*, Advances in Cryptology-ASIACRYPT'96, Lecture Notes in Computer Science. Vol. 1163, pp. 196–209, Springer-Verlag, 1996.

[52] U.M. Maurer and S. Wolf, *Privacy amplification secure against active adversaries*, Advances in Cryptology-CRYPTO'97, Lecture Notes in Computer Science. Vol. 1294, pp. 307-321, Springer-Verlag, 1997.

[53] U.M. Maurer and S. Wolf, *Unconditionally secure key agreement and the intrinsic conditional information* IEEE Transaction on Information Theory, Vol. 45, No. 2, pp. 499-514, 1999.

[54] U.M. Maurer and S. Wolf, *The intrinsic conditional mutual information and perfect secrecy*, Proc. of the 1997 IEEE Symp. on Information Theory, Ulm, Germany, p. 99, 1997.

[55] U.M. Maurer and S. Wolf, *Unconditionally Secure Key Agreement and the Intrinsic Conditional Information*, IEEE Transaction on Information Theory, Vol. 45, No. 2, pp. 499-514, 1999.

[56] U.M. Maurer and S. Wolf, *From Weak to Strong Secrecy in Information-Theoretic Key Agreement*, Proceedings of ISIT 2000, 2000.

[57] U.M. Maurer and S. Wolf, *Information-Theoretic Key Agreement: From Weak to Strong Secrecy for Free*, Advances in Cryptology - EUROCRYPT '00, Lecture Notes in Computer Science, Springer-Verlag, Vol. 1807, pp. 351-368, 2000.

[58] James L. Massey, Lecture notes for *Applied digital information theorey I*, Abteilung für Electrotechnik, ETH Zürich, 1993.

[59] James L. Massey, *A simplified treatment of Wyner's wire-tap channel*, In Proc. 21st Annual Allerton Conf. on Communications, Control and Computing, pp. 268-276, Monticello, IL., Oct. 1983.

[60] K. Mehlhorn and U. Vishkin, *Randomized and deterministic simulagtions of PRAMs by parallel machines with restricted granularity of parallel memories*, Acta Informatica, Vol. 21, Fasc. 4, pp. 235-246, 1984.

[61] J. Pearl, *Probabilistic reasoning in intelligent systems: Networks of plausible inference*, Morgan Kaufmann, San Mateo, 1988.

[62] S. Wolf, *Strong security against active attacks in information-theoretic secret-key agreement*, Advances in Cryptology-ACIACRYPT'98, Lecture Notes in Computer Science. Vol. 1514, pp. 405–419, Springer-Verlag, 1998.

[63] S. Wolf, *Unconditional Security in Cryptography Lectures on data security: modern cryptology in theory and practice*, Lecture Notes in Computer Science, Springer-Verlag, Vol. 1561, pp. 217-250, 1998.

[64] S. Wolf, *Information-Theoretically and Computationally Secure Key Agreement in Cryptography*, ETH dissertation No. 13138, ETH Zürich, 1999.

[65] Alfréd Rényi, *On measures of entropy and information*, Proc. 4th Berkeley Symposium on Mathematical Statistics and Probability (Berlekey), Vol. 1, Univ. of Calif. Press, 1961, pp. 547-561.

[66] Alfréd Rényi, *On the foundations of information theory*, Rev. Inst. Internat. State. Vol. 33, 1965.

[67] Alfréd Rényi, *Probability theory*, North-Holland, Amsterdam, 1970.

[68] D.V. Sarwate, *A note on universal classes of hash functions*, Information Processing Letters, Vol. 10, No. 1, pp. 41-45, 1980.

[69] C.E. Shannon, *A mathematical theory of communication*, Bell System Technical Joural, Vol. 27, pp. 379-423, 623-656, 1948.

[70] C.E. Shannon, *Communication theory of secrecy systems*, Bell System Technical Joural, Vol. 28, pp. 656-715, 1949.

[71] Peter W. Shor, *Algorithms for quantum computation: discret log and factoring*, Proc. 35th IEEE Symposium on Foundations of Computer Science (FOCS), 1994, pp. 124-134.

[72] D.R. Stinson, *Universal hashing and auhtentication codes*, Advances in Cryptology-CRYPTO'91, Lecture notes in Computer Science, Vol. 576, pp. 74-85, Springer-Verlag, 1992.

[73] T. Sugimoto and K. Yamazaki, *A study on secret key reconciliation protocol "Cascade"*, Trans. of the IEICE, Vol. E83-A, No. 10, pp. 1987-1991, 2000.

[74] M.N. Wegman and J.L. Carter, *New hash functions and their use in authentication and set equality*, J. Computer and System Sci. Vol. 22, pp. 265-279, 1981.

[75] A.D. Wyner, *The wire-tap channel*, Bell System Technical Journal, Vol. 54, No. 8, pp. 1355-1387.

[76] K. Yamazaki, M. Osaki, and O. Hirota, *On reconciliation of discrepant sequences shared through quantum mechanical channels*, Lecture Notes in Computer Science, Vol.1396, (Eds. E.Okamoto, G.Davida and M.Mambo), pp. 345-356, Springer-Verlag, 1998.

[77] K. Yamazaki and T. Sugimoto, *On secret reconciliation protocol– modification of "Cascade" protocol* International Symposium on Information Theory and Its applications, Honolulu, Hawaii, Nov 5-8, pp. 223-226, 2000.

[78] S.P. Vadhan, *Extracting all the randomness from a weakly random source*, manuscript, MIT, 1998.

Index

Summary

Secret key agreement solves one of the basic problems in Cryptography, i.e., how two legitimate users can share a secret key. With the secret key, the two users may carry out secret communications with each other later on. There are two types of security models on which secret key agreement is based. One is computational security. Computational security assumes that the adversary has limited computing resources. If the amount of work to break a cryptosystem significantly exceeds the computational resources available to an adversary, the system is called computationally secure. With the model of computational security, a user can use a public key cryptosystem to transfer a secret key to the other user. However, the security level of current public key systems is decreasing since new attack methods are showing up and computers are getting faster and faster.

The other security model is information-theoretic security. It is based on information theory and is a stronger security model, since there is no limit on the adversary's computing resources. Usually, the approaches to information-theoretic security make use of the fact that no one has exact knowledge about the physical world because of its probabilistic behavior. Examples are the noise during communication, or the uncertainty principle of quantum mechanics. Therefore, information-theoretic secret key agreement protocols can be developed by taking advantage of the noise during communication, while quantum key agreement protocols can be developed by making use of the uncertainty principle of quantum mechanics.

The model of information-theoretic secret key agreement assumes that the two legitimate users and the adversary receive noisy versions of a random binary output. There is also a public channel connecting the two users. The process of generating a secret key can be divided into three phases, namely, the *advantage distillation* phase, the *information reconciliation* phase, and the *privacy amplification* phase. The idea of *advantage distillation* is that the two users exploit the authenticity of the public channel to gain an advantage with respect to the information about each other's random variable over the adversary. *Information reconciliation* is the process in which the two users exchange information over the public channel, with which they reconcile their strings to arrive at a common but partially secret string. Finally *privacy amplification* enables the two users to distill from the common and partially secret string a shorter but highly secret string of which the adversary knows only a negligible amount of information.

Quantum key agreement protocols assume that a quantum channel and a public channel connect the two users. With the quantum channel, they implement a *quantum transmission* protocol and some quantum bits are transmitted from one user to the other. The possible eavesdropping of the adversary and the noise of the quantum channel may introduce discrepancies between the two users' qubits. Hence, *information reconciliation* and *privacy amplification* should be implemented over the public channel, so that the two users can reconcile the discrepancies to get a common string and distill some secret bits from the common string.

In Chapter 1 we introduce some basic concepts of probability theory and information theory. We review the history of the research on information-theoretic security and introduce information-theoretic key agreement protocols.

In Chapter 2, we propose a practical protocol to implement both advantage distillation and information reconciliation with as goal that the two users leak as little information as possible. The performance of the protocol is compared with the known protocols.

Privacy amplification is studied in Chapter 3. We focus on the problem of how to use authentication codes to detect active attacks by the adversary. There are three possible settings. First of all, the two users have a partially secret string and a (shorter) secret string. Secondly, they have only one partially secret string. The last setting is that they have two independent, partially secret strings. We show how to use authentication codes in privacy amplification to detect the adversary's active attacks in the three settings.

In Chapter 4, authentication schemes are proposed and analyzed for implementation of information reconciliation over a non-authentic public channel under the assumption that the bit error rate between the random strings of the two users is less than that between theirs and the adversary's.

In Chapter 5, probabilistic upper bounds on the adversary's information are given for different strategies of the adversary in traditional quantum key transmission protocols. The amount of information is related to the final length of the secret key distilled in privacy amplification.

Samenvatting

Hoe twee legitieme gebruikers onderling een geheime sleutel kunnen afspreken, is een van de fundamentele problemen in de cryptografie. Met behulp van die geheime sleutel kunnen de gebruikers hun verdere communicatie afschermen van een eventuele tegenpartij. Er zijn twee soorten modellen van veiligheid waarbinnen sleutelafspraakmethoden worden beoordeeld. De eerste is computationele veiligheid. Hierbij gaat men ervan uit dat de tegenpartij slechts over beperkte rekenkracht beschikt. Als de hoeveelheid rekenwerk die nodig is om een cryptosysteem te kraken de rekenkracht van de tegenpartij ver te boven gaat, heet het systeem computationeel veilig. In dit model kan de ene gebruiker een asymmetrisch cryptosysteem benutten om een geheime sleutel naar de andere gebruiker te versturen. Echter, de veiligheid van de huidige asymmetrische cryptosystemen neemt steeds verder af door de opkomst van nieuwe kraakmethoden en snellere computers.

Het tweede model van veiligheid is informatietheoretische veiligheid. Dit model is gebaseerd op informatietheorie en is sterker, omdat er geen aanname betreffende de rekenkracht van de tegenpartij aan ten grondslag ligt. Meestal maken methoden die informatietheoretische veiligheid bieden gebruik van het feit dat niemand de exacte toestand van de fysieke wereld kan weten, als gevolg van het probabilistische gedrag van die wereld. De ruis op een communicatiekanaal en het onzekerheidsbeginsel van de quantummechanica zijn voorbeelden van zulke probabilistische verschijnselen. Daarom kunnen informatietheoretische sleutelafspraak-protocollen worden ontwikkeld door de ruis op een communicatiekanaal te benutten, en quantumsleutelafspraak-protocollen door het onzekerheidsbeginsel te gebruiken.

Bij informatietheoretische sleutelafspraak-protocollen wordt aangenomen dat de twee legitieme gebruikers en de tegenpartij een door ruis aangetaste willekeurige rij binaire bits ontvangen. Ook worden de twee gebruikers verbonden door een publiek leesbaar kanaal. Het afspreken van een geheime sleutel kan nu worden verdeeld in drie fasen: *voordeeldistillatie*, *informatie-overeenstemming* en *privacyvergroting*. Bij *voordeeldistillatie* benutten de gebruikers de authenticiteit van het publieke kanaal om een voorsprong te krijgen op de tegenpartij in informatie betreffende elkaars bitrijen. Bij *informatie-overeenstemming* wisselen de gebruikers informatie uit over het publieke kanaal, teneinde hun beider bitrijen om te vormen tot één en dezelfde, gedeeltelijk geheime rij. Tenslotte stelt *privacyvergroting* de gebruikers in staat om uit hun gemeenschappelijke en gedeeltelijk geheime bitrij een weliswaar kortere,

maar in veel hogere mate geheime rij te distilleren, waarover de tegenpartij slechts een verwaarloosbare hoeveelheid informatie bezit.

Bij quantum-sleutelafspraak wordt aangenomen dat de twee gebruikers worden verbonden door zowel een quantumkanaal als een publiek kanaal. Met behulp van het quantumkanaal implementeren zij het *quantumtransmissie*-protocol, en de ene gebruiker verstuurt een aantal quantumbits naar de andere gebruiker. Ruis en eventueel afluisteren door de tegenpartij kunnen verschillen introduceren tussen de quantumbits van de beide gebruikers. Daarom moeten de *informatie-oveereenstemming* en de *privacyvergroting* worden geïmplementeerd over het publieke kanaal. Zo kunnen de twee gebruikers de discrepanties teniet doen teneinde een gemeenschappelijke bitrij te verkrijgen, en daaruit een hogelijk geheime sleutel te distilleren.

In Hoofdstuk 1 introduceren we enkele beginselen van de waarschijnlijkheidsrekening en de informatietheorie. We bespreken de geschiedenis van het onderzoek naar informatietheoretische veiligheid, en introduceren informatietheoretische sleutelafspraak-protocollen.

In Hoofdstuk 2 stellen we een praktisch protocol voor om zowel voordeeldistillatie als informatie-overeenstemming te bewerkstelligen, met als doel zo weinig mogelijk informatie te laten uitlekken naar de tegenpartij. De prestaties van ons protocol worden vergeleken met die van bestaande protocollen.

In Hoofdstuk 3 bestuderen we privacyvergroting. We leggen daarbij de nadruk op het gebruik van authenticatiecodes om actieve aanvallen van een tegenpartij te detecteren. We bestuderen dit probleem onder drie verschillende aannamen. In het eerste geval beschikken de twee gebruikers bij aanvang al over zowel een gedeeltelijk geheime bitrij als een kortere, strikt geheime bitrij. In het tweede geval beginnen ze met alleen een gedeeltelijk geheime bitrij, en in het derde geval hebben ze twee onafhankelijke, gedeeltelijk geheime, bitrijen. We laten zien hoe in elk van deze situaties authenticatiecodes kunnen worden gebruikt om privacyvergroting te bewerkstelligen.

In Hoofdstuk 4 introduceren en analyseren we authenticatieschema's ter implementatie van informatie-overeenstemming over een publiek kanaal waarvan authenticiteit niet gegarandeerd kan worden. We gaan daarbij uit van de veronderstelling dat het percentage verschillen tussen de bitrijen van de twee gebruikers kleiner is dan het percentage verschillen met de door de tegenpartij afgeluisterde bitrij.

In Hoofdstuk 5 leiden we probabilistische bovengrenzen af op de hoeveelheid informatie van de tegenpartij bij verschillende strategieën van die tegenpartij in traditionele quantum-transmissieprotocollen. De hoeveelheid informatie wordt gerelateerd aan de lengte van de geheime sleutel die uiteindelijk, na privacyvergroting, overblijft.

Acknowledgements

First of all, I would like to thank my supervisor Henk van Tilborg, who offered me an opportunity to continue my Ph.D. study and broaden my horizons. During my two-year stay in Technische Universiteit Eindhoven, he showed great patience to teach me how to write articles and give presentations in English. Without his thorough checking word by word, hardly can I improve the presentation of my thesis.

I am grateful to Marten van Dijk, who was with Philips Research Laboratories and is now a visiting scientist of the MIT Laboratory for Computer Science, Cambridge. I cooperated much with him to do the research during the past two years. He provided me lots of materials, suggestions and inspiring ideas.

Special thanks go to Martijn Stam, Dr. Peter Beleen, Jeroen Doumen and Jan Draisma. Martijn checked for me Chapter 4 and most of Chapter 2, Peter did Chapter 1, Jeroen did Chapter 5, and Jan helped me with the Dutch Samenvatting. I benefit much from my roommate Martijn Stam's effort to try to drag my Chinglish back to English. He also showed lots of patience to try to teach me Dutch (he gave up due to my laziness). Though we hardly agree on the room temperature of HG 9.91 and affairs of Tibet, it never prevents me from appreciating his humor and wits, which keep me laughing every day :)

I enjoyed lunches with the cozy AIO group, where I felt different cultures. Among the group is Dr. Sander van Rijnswou, Ralf Gramlich, Silvia Boumova, Alexander Kholosha, Stefania Cavallar, Ernesto Reinaldo, Heike Gramberg, Dr. Scott murray, Man Nguyen van Minh, etc.

I want to express my gratitude to my former supervisor Prof. Yumin Wang. Prof. Wang guided me to the field of Cryptography and part of the research was done during the three-year Ph.D. study at Xidian University. Thanks also go to Prof. Xinmei Wang and Prof. Guozhen Xiao for their help during my Ph.D. life at Xidian University. I will never forget the life when I stayed with the brothers and sisters in the Lab. of Information Security and Secrecy at Xidian University: social drink in summer evenings, dancing at the weekends, going to the cinema for celebrations, playing games, renting and borrowing VCDs, etc. I thank Dr. Fangguo Zhang, Jian Mao and Dr. Yajuan Luo for their help after I left Xidian University. I also express my appreciation to Yingfeng Liu, a former classmate of mine in 7-911 at Xidian University, for everything.

I thank Stefan Wolf and Pim Tuyls for interesting discussion about the subjects

of my research.

Thanks go to the members of ACSSNL (Association of Chinese Students and Scholars in the Netherlands) for giving me a feeling at home. Among them are Yuanqing Guo, Jieheng Guo, Ping Wang, Guoying Zhao, Qin Zhao, Bin Yin, Peiyu He, Qing Deng, Yong Liu, Xinwei Jin, Xiaogang Mao, Shengbo Xu, Xiwen Lu, Chen Wang, etc. Since the summer 2001, I have had a very good time playing badminton with some Chinese friends in the badminton club in the sport center of TU/e. The Chinese friends include Yong Zhang, Xiao Dong, Haibin Yang, Jining Xu, Yichun Huang, Gang Qin, Yingbo Zhu, Chengming Gao, etc.

I enjoy my living in the student house since Feburary 2001. I have very nice and lively housemates: Geert Verhaag, Peter Poplavko, Jil Steijvers, Hendrik van Stralen, Colin Gubbels, and Pim Coenen. Whenever beautiful melody is overflowing, it must be Jil and Hendrik singing; whenever I feel the house quaking, without doubt Pim is jumping; Colin's TV is the one that I am always watching; with Geert I like chatting; to beat Peter in pingpong, I still need practising.

I am indebted to my father Minghai Liu and my mother Yuefen Niu. They are always ready to support me and provide me a peaceful harbor. My gratitude to my parents can never be expressed by words. I also thank my brother Xiangyang Liu and my sister-in-law Yaru Jia, who take care of my parents, and my three-year-old little niece Yiyi Liu, who brings so much fun to the whole family.

Curriculum Vitae

Shengli Liu was born on January 5, 1975 (or November 23, 1974 according to the Chinese traditional calender). From September 1991 till July 1995, she studied in School of Economic Management, Xidian University, where she got her Bachelor degree. From September 1995 till February 1998, she worked towards a Master degree in School of Computer Science with a dissertation titled "Interface Design and Study of Verification of a PC Card", supervised by Prof. Bocheng Li. She started Ph.D. study in March 1997 in the Lab. of Information Security and Secrecy, School of Communication Engineering, Xidian University with Prof. Yumin Wang as her supervisor. From March 2000 till February 2002, she continued her Ph.D. Study at the Discrete Mathematics group, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven.

STELLINGEN

behorende bij het proefschrift

Information-Theoretic Secret Key Agreement

van

Shengli Liu

1. Suppose that Alice and Bob have access to a non-authentic public channel. If they share information that is unknown to the adversary Eve, they can change this advantage into creating an authentic public channel. On the other hand, if the public channel provides authenticity, this authenticity may also be changed into an advantage between Alice and Bob.

2. If Eve intercepts/resends every light pulse that Alice sends to Bob, no advantage can be distilled by them between their qubits.

3. The advantage of Hyperelliptic Curve Cryptosystems (HCC) [1] over Elliptic Curve Cryptosystems (ECC) is that a smaller ground field can be used to achieve the same order of magnitude of the Abelian group. That means that HCC can be implemented with a smaller word length in computers than ECC. But this does not mean that HCC are more efficient than ECC. The reason is the slower addition in Jacobians.

4. ASN.1 syntax shows the standard bit representation of the domain parameters of ECC. But ECC parameters can be represented in a more compressed way [2]. For example, choosing fields with trinomial basis or of prime order of special form. Compact representation of the domain parameters of HCC can be found similarly. HCC parameters contain the field over which the curve is defined, the curve itself, the order of the Jacobian, and a generating point.

5. The Weil pairing that can be used to attack ECC can also find positive applications in cryptography: short signatures [4], an identity-based encryption scheme [3], and a three party one round Diffie-Hellman key exchange scheme [5]. The short signature scheme from the Weil pairing can be changed into a zero-knowledge proof of possession of a signature.

6. If today is February 26 2002, Tuesday, everyone knows which day "next Monday" refers to. The problem is: which day does "next Wednesday" mean? February 27 or March 6? The problem remained open in the last two years and will be left to English people. The suggestion is to use "coming Wednesday" instead when Chinese and Dutch want to make an appointment on February 27, and use "Wednesday next week" to refer to March 6.

7. Chinese has a higher information rate than Dutch. Therefore it is harder to perform error corrections. For example, there is only one word to refer to a third party in the singular, but Dutch has at least three, namely "zij, hij, het".

8. All things that are, are with more spirit chased than enjoy'd.

9. Better a witty fool than a foolish wit.

References

[1] N. Koblitz, *Hyperelliptic cryptography*, Journal of Cryptology, No.1, pp. 139-150, 1989.

[2] N. Smart, *Compressed ECC Parameters*. Available at http://www.secg.org/collateral/compressed_ecc.pdf

[3] D. Boneh and M. Franklin, *Identity-Based Encryption from the Weil Pairing*, Advances in Cryptology-CRYPTO'01, Lecture Notes in Computer Science, Springer-Verlag, Vol.2139, pp. 213-229, 2001.

[4] D. Boneh, B. Lynn, and H. Shacham, *Short Signatures from the Weil Pairing*, Advances in Cryptology-ASIACRYPT'01, Lecture Notes in Computer Science, Springer-Verlag, Vol. 2248, pp. 514-532, 2001.

[5] A. Joux, *A one round protocol for tripartite Diffie-Hellman*, Proc of ANTS 4, LNCS 1838, pp. 385-394, 2000.