Perfectly Secure Message Transmission Protocols with Low Communication Overhead and Their Generalization

Griggio, J.

Leiden University

Faculty of Science

Mathematical Institute

Master Thesis

Perfectly Secure Message Transmission Protocols with Low Communication Overhead and Their Generalization

Thesis Advisors
Dr. Robbert de Haan
Prof. dr. Ronald Cramer

Candidate
Jacopo Griggio

Academic Year 2011–2012

Those who leave for a journey leave forever.

Chi parte per un viaggio non fa più ritorno.

Acknowledgements (Ringraziamenti)

Il mio primo ringraziamento, doveroso, va alla mia famiglia. In primis, per la loro totale fiducia in me e nelle mie capacità; ma anche perchè l'avere un solido nucleo familiare alle spalle dà molta certezza nella vita.

Ringrazio anche Dino Festi per colui che è. I contrasti tra le nostre personalità e le nostre filosofie di vita mi hanno spesso dato grandi spunti di riflessione.

Un grazie particolare anche a Marta Lucchini per gli interminabili ma efficaci pomeriggi di studio, per i suoi manicaretti e per la sua costante presenza da coinquilino aggiunto.

E grazie anche a tutti gli altri miei amici, vicini e lontani. Non posso fare un elenco ma un ringraziamento è loro dovuto per tutti i ricordi dei bei momenti passati e per i momenti che verranno.

I want to say thanks to my supervisor Robbert de Haan. I had a tough time with him but it was necessary. I am stubborn, rough and inexpert in writing and his patience and his being careful to details were fundamental for making this thesis.

And thank you, Lusine. For everything.

Abstract

The problem of perfectly secure message transmission concerns two synchronized non-faulty processors sender ($\mathcal{S}$) and receiver ($\mathcal{R}$) that are connected by a synchronous network of $n \geq 2t + 1$ noiseless 2-way communication channels. Their goal is to communicate privately and reliably, despite the presence of an adversary that may actively corrupt at most t of the channels. These properties should hold information theoretically and without error.

Previous work has studied the round complexity and communication overhead of perfectly secure message transmission. The round complexity is the number of consecutive (possibly bi-directional) data transmissions that are required to complete the protocol execution. The communication overhead gives a worst-case value for the total number of bits that needs to be transmitted for each bit of the communicated message, given that the message consists of ℓ bits. In other words, if the worst-case total number of communicated bits is $\mathcal{C}_\ell$ for a message of size ℓ, then the communication overhead for the transmission of this message is $\mathcal{C}_\ell/\ell$.

When $n \leq 3t$, at least two rounds of communication are required and there exists a general lower bound for the communication overhead of $n/(n-2t)$. Furthermore, protocols with linear (respectively constant) communication overhead are known when $n - 2t$ is constant (respectively linear in n).

In order to study the optimal possible communication overhead it is natural to consider messages that are relatively large with respect to the number of channels n. We therefore mainly consider messages consisting of $\Omega(n^\gamma)$ field elements of a given finite field $\mathbb{F}_q$ with $\gamma \in \mathbb{Z}_{\geq 3}$. In this work we study the *precise* optimal communication overhead of such protocols. In particular, we demonstrate that for 3-round protocols the known general lower bound is essentially tight, i.e., there exist 3-round protocols for which the communication overhead is $(n+1)/(n-2t)+O(n^{3-\gamma})$, which converges to $(n+1)/(n-2t)$ as the message size increases.

Furthermore, we narrow the theoretical gap for 2-round protocols by demonstrating that there exist 2-round protocols for which the communication overhead is $6n/(n-2t) + O(n^{2-\gamma})$, which converges to $6n/(n-2t)$ as the message size increases. This should be compared with the best previously known communication overhead for such protocols, which was roughly $25n/(n-2t)$.

We moreover study the basic coding-theoretic techniques involved with these protocol constructions and generalize our new approach. This has potential (theoretical) applications to settings where the message is very small with respect to the number of available communication channels.

Contents

1 Introduction

The problem of transmitting messages securely finds its roots in ancient history. More than 2000 years ago, much before the telecommunications era began, people already had secrets to share. Regardless of whether these secrets were of a personal or a military nature, two aspects were important. The first was that the secret had to reach its destination. The second was that the secret remained secret.

The oldest documented cryptosystem is due to the Roman general Caius Iulius Caesar. Caesar used to protect his secrets using an enciphering method that today is known as the *Caesar cipher*, where every letter of the message was shifted forward three positions in the alphabet. In this way, an A became a D, a B became an E and so on. The principle was simple: even if the courier that was carrying the message was intercepted by an enemy, the enemy would still be unable to determine the secret. This method had a disadvantage though, as only very few people should know about the method. Indeed, the more people would know about it, the more likely it was that this method would be revealed to the enemy. Moreover, once the enemy knew about the method, no secret would have been secure any longer.

Cryptography has advanced dramatically since then. In 1883, in *La Cryptologie Militaire*, Kerckhoffs formulated the famous *Kerckhoffs' principle*: a cryptosystem should be secure even if the enemy knows everything about the system except the key. It is clear that Caesar's method did not respect Kerckhoffs' principle.

Furthermore, the rise of general-purpose computers made it more attractive to design new cryptosystems that respect Kerckhoffs's principle. Perhaps the most famous among such systems is the one due to Rivest, Shamir and Adleman, the *RSA cryptosystem*. The security of this method however relies heavily on the computational capabilities of currently existing calculators; keys that are safe today may not be safe in the near future, especially if the technology of quantum computers can be sufficiently developed.

Hence, better security may be needed. A cryptosystem should be safe independently of an enemy's computational power. Indeed, techniques that are currently out of reach with our technology might become feasible in a future. Hence, we should make the assumption that the enemy is a *worst-case* enemy: he knows every detail of the system and therefore its weaknesses and can take advantage of all of them. Furthermore it may be that for all practical purposes the enemy has unbounded computational power.

Some cryptosystems that are able to deal with unbounded enemies are already known, such as for example the *one-time pad* encryption method. This method consists of having as a key a sequence of pairwise independent numbers, and to then encipher the message by rotating the i-th component of the message by as many positions in the alphabet as the i-th number in the key. In other words, it's like applying a different Caesar's cipher to each position of the message. As long as the key remains secret, this cryptosystem is proven to be unbreakable. On the other hand, the key can be used only once.

All these cryptosystems rely on the same assumption: the key must remain private. Indeed, even the most safe and unbreakable cryptosystem is easily broken once the enemy obtains the key. The solution to this problem is clearly that keys should be exchanged privately, but this might look like a paradox: it seems that we are required to have a private communication to exchange the key we want to use for a private communication. Apart from the logical implication (also the first private communication would require a private key, generating an infinite backward loop), there is a practical implication: if we have already performed a private communication there is no point in doing another a key exchange, because the secret could already be sent. These issues moreover already appear when the enemy is merely eavesdropping. If the enemy is able to additionally corrupt or disturb the communication, no key exchange can safely be executed at all.

A twist to the approach to this problem is due to Dolev, Dwork, Waarts and Yung. In their 1993 paper [DDWY93] they introduce a substantially different assumption, namely that instead of one communication channel multiple, i.e., $n > 1$ communication channels are available to transmit information. They assume moreover that the enemy is able to take control of less than half of these channels; in fact, they proved that this is a necessary and sufficient condition for this model to be useful.

This branch of cryptography is called *perfectly secure message transmission.* We say that perfectly secure message transmission is achieved when two requirements are met. First, the receiver is able to output the message with perfect correctness. Second, no adversary can learn partial information about the message. It was proven that these requirements can be met only if the communication takes place over n channels with $n > 2t$, where t is the number of channels controlled by an adversary. Furthermore, it was proven that in the setting where $n \leq 3t$ at least two communication rounds are necessary.

Since the introduction of perfectly secure message transmission, protocols have been developed that achieve these security requirements. It was proven by Narayanan et al. in [NPRS04] that in the $n \geq 2t+1$ setting a general communication overhead lower bound holds for perfectly secure message transmission protocols of $\frac{n}{n-2t}$, regardless on the number of communication rounds. Most previous work in this field has focused on achieving optimal communication overhead complexity.

Contributions

Our primary aim in this work is to improve the *precise* communication overhead of currently known perfectly secure message transmission protocols.

For protocols that run in 3 communication rounds a theoretical lower bound of $\frac{n}{n-2t}$ for the communication overhead has been proven. Nevertheless, a protocol that meets this lower bound tightly has never been presented. In this work we provide a 3-round protocol which (asymptotically) meets an overhead of $\frac{n+1}{n-2t}$ as the message size increases and is therefore essentially optimal.

The theoretical lower bound of $\frac{n}{n-2t}$ for the communication overhead also holds for 2-round protocols. In previous works an overhead of $\Omega\left(\frac{n}{n-2t}\right)$ was achieved, but the hidden multiplicative constants are not optimal; in the best currently known protocol this constant can in fact be proven to be 25. Therefore, we further optimize the currently known techniques for 2-round perfectly secure message transmission protocols. The communication overhead of the resulting protocol is again $\Omega\left(\frac{n}{n-2t}\right)$, but this time with hidden multiplicative constant 6. Therefore, this protocol is more efficient in terms of communication overhead than all currently known 2-round protocols.

Finally, we study the problem of achieving perfectly secure message transmission in a general setting. Indeed, once we remove restrictions on the linear codes that are used and on the used finite field, the hypotheses required by currently known techniques need not be met anymore. We introduce new techniques that require weaker hypotheses and thereby generalize the constructions of these protocols.

2 Preliminary Theory

In this section we provide some basic theory and background information on the topic that is studied in this thesis. After a brief introduction on Coding Theory in Section 2.1, we introduce and treat the Perfectly Secure Message Transmission problem in Section 2.2. Finally, we discuss Secret Sharing in Section 2.3.

Coding theory is a mathematical discipline which saw its origins in the second half of the last century. As electronic communication began to impact everyday life, the need of developing a mathematical theory that could improve the efficiency of this kind of communication increased. It was important that messages could be sent and received correctly even if the communication channel was subject to errors. Shannon's "A Mathematical Theory of Communication" in 1948 [Sha48] was the starting point for Information Theory. From there on the theory of error-correcting codes was developed in order to expand the theoretical background. Even though Coding Theory is an applied branch of mathematics, several pure disciplines are related to it: Linear Algebra, Algebra, Combinatorics, Number Theory and Algebraic Geometry. Section 2.1 presents some of its basic aspects, such as the Hamming weight and the definition of linear codes, and also provides some theory involving linear codes.

Perfectly Secure Message Transmission (briefly, PSMT) is a concept that was first introduced by Dolev, Dwork, Waarts and Yung in their homonymous paper [DDWY93]. In cryptography there is a crucial problem: if one wants to have encrypted private communication, an encryption/decryption key first must be available. Hence, keys must be exchanged privately before the transmission of the message. However, in a single-channel model this requires a private communication itself, thus generating a paradox. The idea that was introduced by Dolev et al. was to assume that the communication did not take place through a single channel, but through $n > 1$ channels of which less than half were controlled by the adversary. Following their results more advanced protocols were developed in order to reach more and more efficiently the two basic aspects of PSMT: on the one hand eavesdroppers should not be able to obtain even partial information about the message; on the other hand, disturbances generated by intruders should not compromise the correct transmission of the message. Section 2.2 will introduce the problem and provide a mathematical model for it, together with some definitions.

Secret Sharing is a sub-area in theoretical cryptology that has its origins in Shamir's paper *How to Share a Secret* in 1979 [Sha79]. In this paper Shamir introduces the problem of key storage. Since even an unbreakable cryptosystem is easily broken when an adversary knows the secret key, the key of a cryptosystem should be stored in some location which is unaccessible to adversaries. The problem turns out to be that this kind of storage introduces a single-point-of-failure: if the storage does not work properly, there is the possibility that the key is partially or completely lost. The storage in multiple locations can be considered a solution to the problem, but in this case it is more likely that some location leaks some information about the key to adversaries.

The solution that was proposed by Shamir was to keep the key *secret* and then break it into n pieces called *shares*, with the property that for any set of t shares all possible secrets are equally likely, while any set of $t+1$ shares uniquely determines the secret. With this method a key can be stored in several locations without being more vulnerable to adversaries.

2.1 Coding Theory

In this section we present some definitions and results in Coding Theory, mainly focusing on the basic results that are relevant to this work.

From here on, let $\mathbb{F}_q$ denote the field with q elements, where $q = p^r$ for some prime number p and some $r \in \mathbb{Z}_{\geq 1}$. The " $\cdot$ " denotes the usual scalar product between vectors or between matrices, or between vectors and matrices. Let $M_{n\times m}(\mathbb{F}_q)$ denote the set of all matrices with n rows and m columns with coefficients in $\mathbb{F}_q$. For $A \in M_{n\times m}(\mathbb{F}_q)$ we denote its transpose by A^T.

For a vector $\vec{v} \in \mathbb{F}_q^n$ and $1 \leq j \leq n$ we use the notation v_j to denote the element in the j-th coordinate of $\vec{v}$, i.e., we have that $\vec{v} = (v_1, \dots, v_n)$.

Definition 2.1. *Let $n \in \mathbb{Z}_{\geq 1}$ and consider the vector space $\mathbb{F}_q^n$. A* linear code *C is a subspace of $\mathbb{F}_q^n$. We call n the* length *of C.*

Notation. An $[n,k]$-*linear code* is a code of length n and (subspace) dimension k. A *codeword* of a (linear) code C is a vector $\vec{c} \in C$.

It is worthwhile to note that while in general not all error-correcting codes are linear, we only consider linear codes in this work. Linear codes are commonly used for their error-correction capabilities.

Suppose we have a linear code C of length n over $\mathbb{F}_q$ and that a codeword $\vec{c} \in C$ is sent through a channel, where afterwards the vector $\vec{d} \in \mathbb{F}_q^n$ is received. We say that a *transmission error* occurred if there exists an index j such that $d_j \neq c_j$. In this case we define the value $d_j - c_j \in \mathbb{F}_q$ to be the transmission error in the j-th coordinate and the vector $\vec{e} = \vec{d} - \vec{c}$ to be the *error-vector*.

Whether a code C allows to recover the originally transmitted vector $\vec{c}$ from $\vec{d}$ depends on certain properties of the code and the number of indices for which transmission errors have occurred. To make this concept more precise we now introduce the notion of *distance* for vectors of $\mathbb{F}_q^n$. The idea is that a received vector can be corrected if and only if it is "close enough" to the original codeword that was sent.

Definition 2.2. *The* Hamming weight *of a vector* $\vec{v} \in \mathbb{F}_q^n$ *is:*

$$\mathrm{wt}(\vec{v}) := |\{i \in \{1, \dots, n\} \mid v_i \neq 0\}|.$$

It is easy to see that the Hamming weight naturally defines a "distance function" between vectors of $\mathbb{F}_q^n$.

Definition 2.3. *The* Hamming distance *between two vectors* $\vec{v}, \vec{w} \in \mathbb{F}_q^n$ *is:*

$$d(\vec{v}, \vec{w}) := \mathrm{wt}(\vec{v} - \vec{w}).$$

We are now able to define the *minimum distance* of a code.

Definition 2.4. *The* minimum distance *d(C) of a code C is the value:*

$$d(C) := \min_{\vec{c_1}, \vec{c_2} \in C : \vec{c_1} \neq \vec{c_2}} \{d(\vec{c_1}, \vec{c_2})\}.$$

Remark. If C is a linear code the vector $\vec{c_1} - \vec{c_2}$ belongs to C for every $\vec{c_1}, \vec{c_2} \in C$. Therefore, for a linear code C the minimum distance turns out to be equal to the minimal weight among the non-zero codewords of C:

$$d(C) = \min\{\mathrm{wt}(\vec{c}) \mid \vec{c} \in C \setminus \{\vec{0}\}\}.$$

It is well-known from Coding Theory that a linear code is able to correct an error-vector of weight w if and only if $w \leq \lfloor \frac{d-1}{2} \rfloor$, since in this case there is always a unique nearest codeword. In particular circumstances, as we will see in Lemma 2.38, a linear code is able to correct even error-vectors of weight up to $d - 1$, provided that the coordinates of the transmission errors are known.

There are different equivalent ways to represent a linear code. The representation we use in this work is via a *generator matrix* of the code.

Definition 2.5. *A* generator matrix *for an* $[n, k]$*-linear code* C *is a matrix* $G \in M_{k \times n}(\mathbb{F}_q)$ *of rank* k *such that the row vectors span* C*.*

Remark. It follows that the generator matrix of a linear code is uniquely defined up to a choice of basis.

From here on, when we make use of a linear code $C \subset \mathbb{F}_q^n$ we sometimes implicitly assume that a generator matrix $G \in M_{k\times n}(\mathbb{F}_q)$ for this code has a priori been fixed. To *encode* k information symbols means to pick a vector $\vec{v} \in \mathbb{F}_q^k$ and map it into a codeword $\vec{c} = \vec{v} \cdot G$.

Definition 2.6. *A* check matrix *for an $[n, k]$-linear code C is a matrix $H \in M_{n\times(n-k)}(\mathbb{F}_q)$ of rank $n - k$ with the property that $G \cdot H = 0_{k\times(n-k)}$, where $0_{k\times(n-k)}$ is the all-zero matrix in $M_{k\times(n-k)}(\mathbb{F}_q)$.*

From this definition it follows that if C is a linear code with generator matrix G and check matrix H, then for every $\vec{c} \in C$ we have that $\vec{c} \cdot H = 0 \in \mathbb{F}_q^{(n-k)}$.

A fact from Coding Theory regarding the check matrix of a linear code and its minimum distance is given by the following lemma. The proof of this lemma is omitted here since it is a basic linear-algebra argument.

Lemma 2.7. *Let C be a code with check matrix H. Then the minimum distance of C is equal to the cardinality of the smallest subset of rows of H that is linearly dependent.*

Whenever we define a code we implicitly define another code, namely its *dual code.*

Definition 2.8. *Let C be a code of length n over $\mathbb{F}_q$. Then the* dual code *of C is:*

$$C^* = \{\vec{v} \in \mathbb{F}_q^n \mid \forall \vec{c} \in C \ : \ \vec{c} \cdot \vec{v}^T = 0\}.$$

In particular, if C is an $[n, k]$-linear code then this defines a subspace of $\mathbb{F}_q^n$ given by k independent linear equations. This implies that C^* is an $[n, n-k]$-linear code. Now, the following lemma holds for linear codes and allows us to justify this last definition.

Lemma 2.9. *Let $C \subset \mathbb{F}_q^n$ be a linear code with generator matrix $G \in M_{k\times n}(\mathbb{F}_q)$ and check matrix $H \in M_{n\times(n-k)}(\mathbb{F}_q)$. Let C^* be the dual code of C. Then H^T and G^T are respectively a generator matrix and a check matrix for C^*.*

Proof. Let $\vec{w} \in \mathbb{F}_q^{n-k}$ and consider the vector $\vec{v} = \vec{w} \cdot H^T$. Then for every $\vec{c} \in C$ we have:

$$\vec{c} \cdot \vec{v}^T = \vec{c} \cdot (\vec{w} \cdot H^T)^T = \vec{c} \cdot H \cdot \vec{w}^T = \vec{0} \cdot \vec{w}^T = \vec{0},$$

hence C^* contains the linear code generated by H^T. On the other hand $\dim(C^*) = n - k$, while H^T has rank $n - k$, so its rows span an $(n - k)$-dimensional subspace of $\mathbb{F}_q^n$. It follows that C^* and the code generated by H^T have the same dimension, and in fact they are the same code.

Now by definition of G and H we have:

$$H^T \cdot G^T = (G \cdot H)^T = 0^T_{k\times(n-k)} = 0_{(n-k)\times k},$$

so G^T is a check matrix of C^*. □

This lemma implies that, given a linear code C, automatically its dual C^* is uniquely defined up to isomorphisms. Moreover, it also implies that $(C^*)^* = C$, which justifies the fact that this code is called *the dual* code.

A very important result is a bound on the parameters that a linear code can have. Indeed, for a fixed length and a fixed dimension, it is not true that there exist codes with arbitrary minimum distances; the minimum distance has an upper bound.

Proposition 2.10 (Singleton-bound). *Let C be an $[n, k]$-linear code with minimum distance d. Then $n - k \geq d - 1$.*

Codes for which the parameters meet the Singleton-bound (i.e., for which $n - k = d - 1$) are called *Maximum Distance Separable* (*MDS*) codes. Examples of MDS-codes are the *Reed-Solomon* codes.

Definition 2.11. *Let $|\mathbb{F}_q| \geq n$ and $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}_q$ be distinct elements. A* Reed-Solomon code *of length n and dimension $t+1$ over a field $\mathbb{F}_q$ is the subspace of $\mathbb{F}_q^n$ consisting of the vectors of the form*

$$(f(\alpha_1), \ldots, f(\alpha_n)),$$

where $f \in \mathbb{F}_q[x]$ with $\deg(f) \leq t$.

This definition also shows how encoding is done when using Reed-Solomon codes. Suppose we want to encode r elements $a_1, \ldots, a_r \in \mathbb{F}_q$, with $r \leq t$, into a codeword of an $[n, t+1]$-Reed-Solomon code C. We choose an arbitrary index set $\{i_1, \ldots, i_{t+1}\} \subset \{1, \ldots, n\}$. Then using Lagrange interpolation we find a polynomial $f \in \mathbb{F}_q[x]$ with $\deg(f) = t$ such that $f(\alpha_{i_1}) = a_1, \ldots, f(\alpha_{i_r}) = a_r$ and $f(\alpha_{i_{r+1}}) = a_{r+1}, \ldots, f(\alpha_{i_{t+1}}) = a_{t+1}$ for some random values $a_{r+1}, \ldots, a_{t+1} \in \mathbb{F}_q$. Finally, we define the codeword $\vec{c} \in C$ by setting $\vec{c} = (f(\alpha_1), \ldots, f(\alpha_n))$. f is said to be the *defining polynomial* of $\vec{c}$.

2.2 Perfectly Secure Message Transmission

In this section we sketch the background environment in which we are going to work. Suppose a sender $\mathcal{S}$ wants to send a secret message M to a receiver $\mathcal{R}$. The most natural way in which $\mathcal{S}$ can do this is to send M through a two-way channel, say s, that connects $\mathcal{S}$ with $\mathcal{R}$. However, suppose there is also a third party present during this process: an adversary $\mathcal{A}$ who tries to eavesdrop on and/or to disturb the communication. If $\mathcal{A}$ is able to read any information that is sent through s, then $\mathcal{A}$ learns the content of the secret message. Moreover, if $\mathcal{A}$ is also able to modify information sent through s, $\mathcal{R}$ cannot even be sure that the message that he received is indeed the secret message sent by $\mathcal{S}$. Summing up, two reasonable requirements for secure message transmission are violated: secrecy and reliability. Our goal is to perfectly (i.e., with probability 1) meet these two requirements.

For this, we require the following model. The communication is not going to take place through a single channel, but through $n > 1$ distinct channels. Indeed, a necessary assumption that we require to reach our goal is that $\mathcal{A}$ cannot read or modify data on all communication channels. More specifically, we will assume that strictly less than half of the channels are under the influence of $\mathcal{A}$. Next, we equip $\mathcal{S}$ and $\mathcal{R}$ with a protocol $\mathcal{P}$. This protocol guarantees that at the end of its execution these two parties will share enough common secret information to communicate using the perfectly secure method of one-time pad encryption. In this way, $\mathcal{A}$'s influence on the channels becomes irrelevant both for gathering information about M and for disturbing the communication.

2.2.1 The model

The precise model for the environment explained above is the following. Let $\mathcal{S}$, $\mathcal{R}$ and $\mathcal{A}$ be three synchronized non-faulty processors. Let $N := \{1, \ldots, n\}$ be a public index set for n two-way channels connecting $\mathcal{S}$ to $\mathcal{R}$. A pair of algorithms implemented by $\mathcal{S}$ and $\mathcal{R}$ (i.e., a *protocol*) $\mathcal{P}$ is known by all three parties and is organized according to the following framework. $\mathcal{A}$ selects a set $I_{\mathcal{A}} \subset N$ with these properties: if $i \in I_{\mathcal{A}}$ then $\mathcal{A}$ is able to read the information sent through the i-th channel and to replace it with a private random string. If $i \notin I_{\mathcal{A}}$ instead, $\mathcal{A}$ is not able to read the information sent through the channel, and is not able to change it either. We say that the i-th channel is *controlled* by $\mathcal{A}$ if and only if $i \in I_{\mathcal{A}}$.

A *communication round* proceeds as follows. First, for every $i \in N$ one party (that could be either $\mathcal{S}$ or $\mathcal{R}$) sends a random private vector $\vec{c_i}$ (usually an element of a fixed finite vector space) through channel i. Next, for each $i \in N$ the other party receives a vector $\vec{c_i}'$, with $\vec{c_i}' = \vec{c_i}$ if $i \notin I_{\mathcal{A}}$ and $\vec{c_i}' = \vec{c_i} + \vec{a_i}$ if $i \in I_{\mathcal{A}}$, where $\vec{a_i}$ is a private vector chosen by $\mathcal{A}$.

We assume that information is transmitted through every channel without delay and without transmission errors. We also assume that $\mathcal{A}$ has unbounded computational power and that his actions on the channels he controls give rise to negligible delays in transmission. In particular, we are interested in worst-case adversaries, i.e., adversaries that are able to take advantage of every possible weakness of the protocol.

There are two types of adversary. We say that an adversary is *passive* if he only reads the data transmitted over the channels he controls without modifying it, while we say that an adversary is *active* if he modifies data as well. Note that worst-case adversaries are not necessarily active adversaries, as we demonstrate further below. Also note that an adversary may be active on some channels and passive on others.

A channel is said to be *corrupted* if the behaviour of the adversary is active.

Definition 2.12. *An adversary $\mathcal{A}$ is called a t-*adversary *if at the beginning of every round $|I_{\mathcal{A}}| \leq t$.*

As we stated above, we everywhere assume that the adversary always controls strictly less than half of the channels. In other words, we require $n > 2t$. It has indeed been proven by Dolev et al. in [DDWY93] that otherwise PSMT is not possible.

Definition 2.13. *The* view *of the adversary $\mathcal{A}$ at a given point during the protocol execution, denoted by $V_{\mathcal{A}}$, consists of his randomly generated strings, together with all the data transmitted so far over the controlled channels.*

Remark. Note that this definition indirectly also includes all modifications that $\mathcal{A}$ made to the data he read, and all other information that can be deduced from his view. Indeed, all the modifications are obtained deterministically from the initial random values and the observed data.

Suppose that $\mathcal{S}$ initially selects a private random message M from a finite set $\mathfrak{M}$, and that he initiates a communication with $\mathcal{R}$ to transfer M securely. Our goal in this communication is to achieve perfectly secure message transmission (*PSMT*), as defined below.

Definition 2.14. *A protocol $\mathcal{P}$ achieves* perfectly secure message transmission *if at the end of every execution of $\mathcal{P}$ the following two properties hold:*

- *(Correctness) $\mathcal{R}$ outputs a string M' with:*

$$P(M' = M) = 1.$$

- *(Privacy) For every $M_1, M_2 \in \mathfrak{M}$ chosen uniformly at random and for any possible adversarial view V, we have:*

$$P(M = M_1 | V_{\mathcal{A}} = V) = P(M = M_2 | V_{\mathcal{A}} = V).$$

These probabilities are defined over the local random coins of $\mathcal{S}$, $\mathcal{R}$ and $\mathcal{A}$.

Definition 2.15. *A protocol $\mathcal{P}$ is a* perfectly secure message transmission protocol *if it achieves perfectly secure message transmission.*

The inputs of $\mathcal{P}$ consist of the random data that is locally generated by $\mathcal{S}$, $\mathcal{R}$ and $\mathcal{A}$ at the beginning of its execution, and in addition the message $M \in \mathfrak{M}$ for $\mathcal{S}$. Note that after these initial random choices are made, the rest of the protocol runs deterministically.

Commonly used perfectly secure message transmission protocols can in general be divided into three steps.

Step 1: **Transmission Subprotocol.** During this step $\mathcal{S}$ and $\mathcal{R}$ have communication rounds as described above, based on their initial random strings and all the additional data they collected from the previous rounds.

Step 2: **Information Reconciliation.** During this step $\mathcal{S}$ and $\mathcal{R}$ use the correlation established in the first step to interactively agree on a mutual (bit) string. Note that $\mathcal{A}$ may partially know this string, but not completely.

Step 3: **Privacy Amplification.** During this step $\mathcal{S}$ and $\mathcal{R}$ use a previously agreed upon algorithm to extract from the string computed during the information reconciliation a key which is totally unknown to $\mathcal{A}$, and that can be used for a one-time pad encryption.

Although it is known that a protocol made of these three steps can achieve PSMT, a converse of this fact has not yet been proven.

Suppose that for a message transmission we require correctness but not privacy. This may occur in several cases, for example when the conversation is already enciphered with some enciphering method unknown to $\mathcal{A}$, or if the data sent does not change the view $V_{\mathcal{A}}$. In this case, the easiest (and perhaps most natural) method to transmit is the following.

Definition 2.16. *We say that a message is* broadcast *if it is sent simultaneously over all the n channels.*

Suppose we are in a situation where $n > 2t$. If a message is broadcast and an active adversary corrupts this message on some of his channels, the receiver can easily determine which among all the messages he received is the correct one by simply choosing the message that appears most. $\mathcal{A}$ learns the message, but this is not important because privacy is not required.

Remark. This is a case in which the worst-case adversary is a passive adversary. Indeed, suppose that an active adversary corrupts some data on his channels. Then, once the receiver knows which message is the original message, he immediately spots the corrupted channels. So in this case for $\mathcal{A}$ it is more convenient not to corrupt any data.

Since we always assume worst-case adversaries, whenever a message is broadcast we assume without loss of generality that the adversary is passive during broadcasts.

2.2.2 Complexity

Whenever we deal with an algorithm or a protocol which requires data transmission and computation, a very important aspect to be considered is complexity. It is obvious that all other aspects being equal, the algorithm or the protocol which needs less computation performed or less information sent is better. Here we give two definitions for communication complexity which are used to measure the efficiency of a protocol; these two definitions take into account that in modern electronic communication messages are sent in the form of binary strings.

Definition 2.17. *The* communication complexity $\mathcal{C}_\ell$ *of a protocol is the minimum number of total bits that must be transmitted by* $\mathcal{S}$ *and* $\mathcal{R}$ *during the whole execution of the protocol in the worst-case scenario, given a message consisting of* ℓ *bits.*

Definition 2.18. *The* communication overhead $\mathcal{L}$ *of a protocol is the ratio between the communication complexity needed to send a message of* ℓ *bits and the length of the message itself, i.e.* $\mathcal{L} := \mathcal{C}_\ell/\ell$.

This definition refers to the number of computations that parties must perform to run the protocol instead.

Definition 2.19. *The* computational cost *of a protocol is the minimum number of bit operations that must be performed by* $\mathcal{S}$ *and* $\mathcal{R}$ *for the protocol to complete in the worst-case, given a message consisting of* ℓ *bits.*

This third definition is important because a protocol which has a very low communication complexity could be totally inefficient in terms of computation, for example when it needs to solve a computationally hard problem between one transmission and another. Note that in general, since the transmission of a bit always requires computation, a protocol with a high communication complexity will always have high computational cost.

We are of course mainly interested in those protocols which have communication overhead and computational cost as low as possible. In general, a protocol is considered efficient when these two parameters are *polynomial* in n and ℓ, i.e. they are $O(n^{k_1}\ell^{k_2})$ for some $k_1, k_2 \in \mathbb{Z}_{\geq 1}$.

2.2.3 Known Results

After having stated the PSMT problem in 1993, Dolev et al. in [DDWY93] gave some initial fundamental results. The first was that perfect correctness and perfect privacy can be achieved in a single-round transmission if and only if $n \geq 3t+1$. If we allow more communication rounds, which implies an interaction between the sender and the receiver, then we can reach our goal even with $n \geq 2t+1$. Finally, it was proven that if $n \leq 2t$ no solution to the problem exists.

In 2007 in their paper [FFGHV07] Fitzi, Franklin, Garay and Harsha Vardan proved that an optimal communication overhead of $\mathcal{L} = \frac{n}{n-3t}$ can be achieved for a one-round protocol under the condition $n \geq 3t+1$; in 2004 a lower bound of $\mathcal{L} = \frac{n}{n-2t}$ for the communication overhead of protocol with two (or more) rounds in the case $n \geq 2t+1$ was given by Narayanan et al. in the paper [NPRS04].

In 2006 Agarwal, Cramer and de Haan [ACH06] presented a protocol for $n \geq 2t+1$ with the (optimal) communication overhead of $\Omega(n)$; this protocol, however, was computationally inefficient. In 2008 in their paper [KS08] Kurosawa and Suzuki presented a PSMT protocol for $n \geq 2t+1$ which not only reaches the communication overhead of $\Omega(n)$ ($25n$, precisely), but also has efficient computational cost. These are the best results currently known.

Even though a communication overhead lower bound of $\frac{n}{n-2t}$ was proven, no known protocol at the moment reaches this overhead, i.e., none meets tightly the lower bound given by Narayanan et al. . Section 3 presents the first efficient protocol for $n \geq 2t+1$ that (asymptotically) minimizes the communication overhead. Moreover, in Section 4 we present a 2-round protocol which significantly improves the communication overhead reached by Kurosawa and Suzuki.

Since these results rely on the possibility of choosing arbitrarily large finite fields, one may ask if PSMT protocols can be defined in an environment where the size of the field does not depend on n (and can in particular be much smaller). Appendix A approaches this problem in a more general setting.

2.3 Secret Sharing

The application of secret sharing that we consider in this work is the application to perfectly secure message transmission. Indeed, suppose that a sender $\mathcal{S}$ wants to send a secret to a receiver $\mathcal{R}$, and suppose that n channels connect the two parties. Then $\mathcal{S}$ divides the secret into n shares, and sends each share through a different channel. If a passive adversary $\mathcal{A}$ has size t, then the t shares he learns during the transmission should not be enough for him to learn the secret. In general secret sharing techniques allow to create a distribution beween chunks of data such that certain combinations can determine the secret, while others cannot.

It is worthwhile to note that this does not solve the PSMT problem itself, because an adversary may be active; a secret sharing scheme is not protected against corruptions by itself. For this reason we not only use secret sharing techniques.

2.3.1 Definitions

Let $S_0, \ldots, S_n$ be random variables which are defined respectively on the finite alphabets $X_0, \ldots, X_n$. Without loss of generality we may assume that $\forall i \in \{1, \ldots, n\}$, $\forall s_i \in X_i$:

$$P(S_i = s_i) > 0.$$

Definition 2.20. *A* vector of random variables *is a vector $(S_i)_{i \in I}$ where $I = \{1, \ldots, n\}$ is a non-empty index set, and for every $i \in I$ S_i is a random variable.*

Definition 2.21. *A* secret sharing scheme *is a pair $(\vec{S}, j)$ with the following properties:*

- *$\vec{S} := (S_i)_{i \in I}$ is a vector of random variables;*
- *$n > 1$;*
- *$|X_j| > 1$;*
- *For every $s_j \in X_j$ we have:*

$$P(S_j = s_j) = \frac{1}{|X_j|},$$

i.e., S_j has the uniform distribution on X_j.

The index j is called the *designated index.* From here on, we denote as I^* the set $I \setminus \{j\}$.

Definition 2.22. *Let $(s_i)_{i\in I} \in \prod_{i\in I} X_i$, with:*

$$P((S_i)_{i\in I} = (s_i)_{i\in I}) \neq 0.$$

Then s_j is said to be the secret *of $(s_i)_{i\in I}$, while the elements $\{s_i\}_{i\in I^*}$ are said to be the* shares.

Let $\mathfrak{P}(I^*)$ be the set of all subset of I^*. For a secret sharing scheme we define two sets $\Gamma, \mathrm{A} \subset \mathfrak{P}(I^*)$ as follows.

Definition 2.23. *A subset $B \subset I^*$ is said to be* accepted *if for every possible choice $(s_i)_{i\in B} \in \prod_{i\in B} X_i$ we have that:*

$$\exists s_j \in X_j \text{ s.t. } P(S_j = s_j \mid \{S_i = s_i\}_{i\in B}) = 1.$$

Definition 2.24. *A subset $B \subset I^*$ is said to be* rejected *if for every possible choice $(s_i)_{i\in B} \in \prod_{i\in B} X_i$ and for every $s_j \in X_j$ we have:*

$$P(S_j = s_j \mid \{S_i = s_i\}_{i\in B}) = \frac{1}{|X_j|}.$$

We then define Γ to be the set of all accepted subsets, and A to be the set of all rejected subsets.

Remark. Γ and A clearly have the property that $\Gamma \cap \mathrm{A} = \emptyset$, while it is not true in general that $\Gamma \cup \mathrm{A} = \mathfrak{P}(I^*)$.

Definition 2.25. *A secret sharing scheme for which $\Gamma \cup \mathrm{A} = \mathfrak{P}(I^*)$ is said to be* perfect.

An interesting property that Γ and A have is that they are closed with respect to union and intersection, respectively. In other words, Γ is *monotone*, while A is *anti-monotone*.

Definition 2.26. *A secret sharing scheme has t-*privacy *if for every $B \subseteq I^*$ with $|B| \leq t$ we have that $B \in \mathrm{A}$.*

Definition 2.27. *A secret sharing scheme has r-*reconstruction *if for every $B \subseteq I^*$ with $|B| \geq r$ we have that $B \in \Gamma$.*

These two definitions are just a formal way to say that if a secret sharing scheme has t-privacy and r-reconstruction, then every subset of I^* of cardinality at least r is accepted, while every subset of cardinality at most t is rejected. Since $\Gamma \cap \mathrm{A} = \emptyset$ as we pointed above, it is straightforward to see that $0 \leq t < r \leq n$ holds.

Definition 2.28. *An index $i \in I^*$ is said to be* dummy *if the two following conditions hold:*

- $\forall B \in \Gamma$*,* $B \setminus \{i\} \in \Gamma$*;*
- $\forall B \in \mathrm{A}$*,* $B \cup \{i\} \in \mathrm{A}$*.*

Equivalently to this definition, we can say that an index is dummy if both its presence and its absence are irrelevant to learn the secret.

Definition 2.29. *A secret sharing scheme is said to be* ideal *if it has no dummy indices.*

For practical reasons we are mainly interested in ideal secret sharing schemes. A very important aspect of ideal perfect secret sharing schemes comes from the following theorem.

Theorem 2.30. *Let $(\vec{S}, j)$ be an ideal perfect secret sharing scheme, with $\vec{S} = (S_i)_{i \in I}$. Then, for every $i \in I^*$ we have $|X_i| \geq |X_j|$.*

Proof. See for instance [Haa09]. □

It follows from this theorem that, since for each index $i \in I^*$ the number of possible shares in that position is larger than the number of possible secrets, the minimum number of bits required to represent each share is greater than or equal to the minimum number of bits required to represent the secret. In other words, each share is at least as large as the secret.

Among ideal schemes, a very interesting family of schemes is represented by *threshold schemes.*

Definition 2.31. *A secret sharing scheme is said to be t-*threshold *if it has t-privacy and $(t+1)$-reconstruction.*

Note that this definition implies that a t-threshold scheme is ideal, as the following easy lemma states.

Lemma 2.32. *A threshold secret sharing scheme is ideal.*

Proof. Consider a t-threshold secret sharing scheme and suppose by contradiction that it has a dummy index i. Then every set in A of cardinality t contains i. Indeed, if there exists $B \in \mathrm{A}$ with $|B| = t$ and $i \notin B$, then $B \cup \{i\} \in \mathrm{A}$ because i is a dummy index; but $|B \cup \{i\}| = t+1$, so $B \cup \{i\} \in \Gamma$, contradicting the fact that $\Gamma \cap \mathrm{A} = \emptyset$. Hence, every set in A of cardinality t contains i; but by hypothesis A contains every set of cardinality at most t, and not all of them include i. This leads to a contradiction. □

We remark also that threshold schemes are perfect. Indeed, for every $B \in \mathfrak{P}(I^*)$ either $|B| \leq t$, or $|B| \geq t+1$, hence it immediately follows that $\Gamma \cup \mathrm{A} = \mathfrak{P}(I^*)$.

We now report a famous secret sharing scheme introduced by Shamir in [Sha79] in 1979.

Example 2.33 (Shamir's Secret Sharing Scheme)**.** Shamir's (t,n)-secret sharing scheme is described as follows. Let $\mathbb{F}_q$ be a finite field with $q > n$. Privately select uniformly at random an element $s \in \mathbb{F}_q$ and a polynomial $f \in \mathbb{F}_q[x]$ with $\deg(f) \leq t$ and $f(0) = s$. We define $f(0) = s$ to be the secret, and $\{f(i)\}_{i=1}^{n}$ to be the shares.

Shamir's (t,n)-secret sharing scheme is a t-threshold scheme. Indeed, by Lagrange's Interpolation Theorem we can recover a polynomial f of degree t from r evaluation points if and only if $r > t$; for this reason all sets of size $\geq t+1$ are accepted. On the other hand, given any set B of size t the set $B \cup \{0\}$ has size $t+1$; for any pair of possible secrets $s_1, s_2 \in \mathbb{F}_q$ Lagrange Interpolation gives two polynomials $f_1(x), f_2(x) \in \mathbb{F}_q[x]$ with $f_1(0) = s_1$, $f_2(0) = s_2$ and clearly $f_1 \neq f_2$. Since the secrets have a uniform distribution over $\mathbb{F}_q$ it follows that the polynomials that can be deduced from B using Lagrange Interpolation are as many as the secrets and uniformly distributed; hence, the set B is rejected.

2.3.2 Error-Correcting Codes

As a consequence of Theorem 2.30, for any ideal secret sharing scheme each share is at least as large as the secret. From here on, we consider a particular type of ideal schemes, where every random variable takes values in the same space: a finite field $\mathbb{F}_q$ for some fixed prime power q. Also, for simplicity the index set I will be $\{1, \ldots, n+1\}$ and the designated index j will be $n+1$.

There is a link between secret sharing schemes and error-correcting codes. Suppose we have a code C of length $n+1$, and let $\vec{c} = (c_i)_{i=1}^{n+1} \in C$ be a codeword. Then we obtain a secret sharing scheme simply by setting c_{n+1} to be the secret, and c_i to be the i-th share for $i \in \{1, \ldots, n\}$.

One can study the properties of these kinds of secret sharing schemes. In particular, we are interested in how the parameters of the code C influence peculiarities of the scheme, such as the privacy and the reconstruction thresholds. For instance, it is known that a code C with minimum distance d is able to correct up to $e = \left\lfloor \frac{d-1}{2} \right\rfloor$ transmission errors. This immediately implies that the secret sharing scheme associated with C has (at least)

$(n-e)$-reconstruction. On the other hand, if we take

$$C = \{\vec{c} \in \mathbb{F}_q^{n+1} \mid c_{n+1} = \sum_{i=1}^{n} c_i\},$$

then there is no chance to recover the secret c_{n+1} without knowing all the n shares. In this case, we have clearly $(n-1)$-privacy.

In this section we describe in more detail how these thresholds behave. First, we introduce a new family of secret sharing schemes.

Definition 2.34. *A secret sharing scheme is said to be* linear *if for any two secrets s_{n+1}, s'_{n+1} with respective share vectors $(s_1, \dots, s_n)$ and $(s'_1, \dots, s'_n)$ and for any $\lambda \in \mathbb{F}_q$, the vectors $(s_1 + s'_1, \dots, s_n + s'_n)$ and $(\lambda s_1, \dots, \lambda s_n)$ are valid share vectors for the secrets $s_{n+1} + s'_{n+1}$ and λs_{n+1} respectively.*

It follows immediately from this definition that a secret sharing scheme associated with a linear code C is in fact a linear secret sharing scheme.

Definition 2.35. *A linear secret sharing scheme is said to be a* ramp scheme *if it has t-privacy and r-reconstruction with $r > t+1$.*

Ramp schemes are also called *quasi-threshold* secret sharing schemes. Whether a linear secret sharing scheme associated with a linear code C is a threshold scheme or a ramp scheme is explained by the following theorem.

Theorem 2.36. *A secret sharing scheme associated with a linear code C is a threshold scheme if and only if C is a MDS-code.*

Proof. See [Haa09]. □

Example 2.37. Shamir's (t,n)-secret sharing scheme is a t-threshold scheme. Indeed, it can be seen as a secret sharing scheme associated with a Reed-Solomon code of length $n+1$ and dimension $t+1$.

In this work we do not always deal with MDS-codes, so the schemes we use are in general ramp schemes.

The next lemma points out a useful property of linear codes. Recall that an *erasure* is a transmission error for which the position (but not the value) is known.

Lemma 2.38. *Let C be a code of length n over $\mathbb{F}_q$ with minimum distance d. Then C is able to correct up to $d-1$ erasures.*

Proof. Suppose a codeword $\vec{c} \in C$ is received with $d-1$ erasures in the positions $i_1, \dots, i_{d-1}$; let $I = \{i_1, \dots, i_{d-1}\}$. Consider the subset $V \subset \mathbb{F}_q^n$ defined this way:

$$V = \{\vec{v} \in \mathbb{F}_q^n \mid v_i = c_i \ \forall i \notin I\}.$$

Then, we prove that $V \cap C = \{\vec{c}\}$. Indeed, for sure $\vec{c} \in V \cap C$. Suppose there exists $\vec{c}' \in V \cap C$, $\vec{c}' \neq \vec{c}$; Then $d(\vec{c}, \vec{c}') \leq d-1$, but this is a contradiction since the minimum distance of C is d. □

We are now able to state how the privacy and the reconstruction thresholds behave in the case of linear secret sharing schemes associated with linear codes.

Theorem 2.39. *Let C be an $[n+1, k]$-linear code with minimum distance d; let also C^* be its dual code, and d^* the minimum distance of C^*. Then the linear secret sharing scheme associated with C is a $(d^*-2, n-d+2)$-ramp scheme.*

Proof. First, we prove the reconstruction threshold. Suppose we have $n-d+2$ elements of a codeword $\vec{c} \in C$ at the positions $i_1, \dots, i_{n-d+2}$ with $n+1$ not appearing in this list. Let $I = \{i_1, \dots, i_{n-d+2}\}$; we flag the elements in the positions $\{1, \dots, n\} \backslash I$ as erasures. Then we have exactly $(n+1)-(n-d+2) = d-1$ erasures, and by Lemma 2.38 we can uniquely recover $\vec{c}$. This implies that I is accepted.

Now, we prove the privacy threshold. Assume that $I = \{i_1, \dots, i_{d^*-2}\}$ with $n+1 \notin I$ is a set of shares which jointly determine the secret. Then there is a row in a check matrix of C with 0 in all entries but in the positions determined by $I \cup \{n+1\}$. This is equivalent to saying that the code C^* contains a codeword whose non-zero entries lie only on the positions given by $I \cup \{n+1\}$. This codeword has Hamming weight d^*-1 by definition, so we get a contradiction. Hence, the shares in I do not determine the secret. □

2.3.3 High Information Rate Ramp Schemes

The main advantage that ramp schemes have when compared to threshold secret sharing schemes is that, at the price of some uncertainty about the privacy and reconstruction thresholds, they allow a higher information rate.

The secret sharing schemes we studied so far all have in common the fact that the secret is just a single element of the field $\mathbb{F}_q$. Theorem 2.30 states that for every ideal secret sharing scheme each share is at least as large as the secret. If, however, we go beyond the standard definition of secret sharing schemes, we can also go beyond this result. Hence, it is reasonable to ask whether it is possible to have secrets of bigger length.

Here is a first generalization of the schemes associated with linear codes that we have seen so far. Let C be a code of length $n+\ell$ with $\ell > 1$, and define for each codeword the elements at the positions $\{1, \ldots, n\}$ to be the shares, and those at the positions $\{n+1, \ldots, n+\ell\}$ to be the secrets.

Lemma 2.40. *Let C be an $[n+\ell, k]$-linear code with minimum distance $d \geq \ell+1$; let also C^* be its dual code, and $d^* \geq \ell+1$ the minimum distance of C^*. Then the ramp scheme associated with the code C is a $(d^*-\ell-1, n+\ell-d+1)$-ramp scheme.*

Proof. Acceptance follows by applying the same proof as in Theorem 2.39, substituting n with $n+\ell-1$.

As for rejection, we use again an argument which is similar to the one used for Theorem 2.39. Assume that $I = \{i_1, \ldots, i_{d^*-\ell-1}\}$ with $\{n+1, \ldots, n+\ell\} \cap I = \emptyset$ is a set of shares which have a linear relation with the secret (or with part of it). Then the code C^* contains a codeword whose non-zero entries lie only on the positions contained in $\{n+1, \ldots, n+\ell\} \cup I$. This codeword has Hamming weight at most $d^* - \ell - 1 + \ell = d^* - 1$ by definition, so we get a contradiction. Hence, the shares in I are uncorrelated with the secret. □

The limitation $d \geq \ell + 1$ is due to the fact that, since we have n shares in this scheme, we need that $n + \ell - d + 1 \leq n$, while the limitation $d^* \geq \ell + 1$ is needed because the privacy threshold should be at least 0.

Here we give an example that will be very useful later.

Example 2.41. Let C be an $[n+\ell, t+\ell]$-Reed-Solomon code. Since C is MDS it meets the Singleton bound, hence its minimum distance is $(n+\ell) - (t+\ell)+1 = n-t+1$. On the other side, we have that the minimum distance of the dual code is $(t+\ell)+1 = t+\ell+1$. With these parameters, and assuming that $n-t+1 \geq \ell+1$ as in the hypothesis of Lemma 2.40, we have that the ramp scheme associated with C has $(t+\ell+1)-\ell-1 = t$ privacy and $n+\ell-(n-t+1)+1 = t+\ell$ reconstruction.

In [Haa09] we find a construction that allows to have secrets of size $\Omega(n)$, without using codes of length bigger than n. Let $\hat{C}$ be an $[n, \hat{k}]$-linear code, and let $C \subset \hat{C}$ be a subcode. In particular, C is an $[n, k]$-linear code with $k < \hat{k}$. Now, let $\ell = \hat{k} - k$ and consider a linear map:

$$\varphi \colon \mathbb{F}_q^\ell \to \mathbb{F}_q^n,$$

with the properties:

- φ is injective;

- $\mathrm{Im}(\varphi) \oplus C = \hat{C}$.

Let $S = \mathrm{Im}(\varphi)$. Such an S can always be found, for example by completing a basis of C to a basis of $\hat{C}$. Any $\vec{\hat{c}} \in \hat{C}$ can be uniquely written as:

$$\vec{\hat{c}} = \vec{c} + \vec{s},$$

with $\vec{c} \in C$, $\vec{s} \in S$. Let also $\vec{v} \in \mathbb{F}_q^\ell$ s.t. $\varphi(\vec{v}) = \vec{s}$.

In this environment, we define the vector $\vec{\hat{c}}$ to be the set of shares, and the vector $\vec{v}$ to be the secret.

Theorem 2.42. *Let $\hat{C}$ be an $[n, \hat{k}]$-linear code with minimum distance $\hat{d}$ and let C be a subcode of $\hat{C}$ of dimension $k < \hat{k}$. Let d^* be the minimum distance of the dual code of C. Then the secret sharing scheme described above is a $(d^* - 1, n - \hat{d} + 1)$-ramp scheme.*

Proof. As pointed out above, every codeword $\vec{\hat{c}}$ of $\hat{C}$ can be written as $\vec{\hat{c}} = \vec{c} + \vec{s}$, with $\vec{c} \in C$, $\vec{s} \in S$. It follows that $\vec{s} = \vec{\hat{c}} - \vec{c}$. Therefore, a set I is accepted if it is accepted by both the scheme associated with $\hat{C}$ and the scheme associated with C. On the other hand, a set I is rejected if it is rejected by both those schemes. Let d be the minimum distance of C and $\hat{d}^*$ be the minimum distance of the dual code of $\hat{C}$.

First, we prove acceptance. By Theorem 2.39 the acceptance thresholds for the schemes associated with $\hat{C}$ and C are, respectively, $n - \hat{d} + 1$ and $n - d + 1$. I must be accepted by both, so it must have cardinality at least the maximum of the two. $d \geq \hat{d}$ because $C \subset \hat{C}$, so it follows that $n - \hat{d} + 1 \geq n - d + 1$ and acceptance is proven.

Now, we prove rejection. By Theorem 2.39 the rejection thresholds for the schemes associated with $\hat{C}$ and C are, respectively, $\hat{d}^* - 1$ and $d^* - 1$. I must be rejected by both, so it must have cardinality at most the minimum of the two. We have $\hat{d}^* \geq d^*$ because $\hat{C}^* \subset C^*$, so it follows that $d^* - 1 \leq \hat{d}^* - 1$ and rejection is proved. □

This concludes the discussion of secret sharing schemes. In the next sections we apply secret sharing to perfectly secure message transmission and present two communication-optimal PSMT protocols.

3 An Asymptotically Optimal 3-Round Protocol

In this section we treat the first of the main topics of this work, which is a 3-round protocol that asymptotically meets the known lower bound for communication overhead. Recall that Narayanan et al. [NPRS04] proved that in the case of $n \geq 2t+1$ (where n is the number of channels and t is the size of the adversary) PSMT protocols have a lower bound in the communication overhead equal to $\frac{n}{n-2t}$. Nevertheless, a protocol which actually meets this bound has never been presented.

Throughout this section we always stick to the case $q = \Omega(n)$. Indeed, the problem is much easier when we let q be arbitrarily larger than n, while it is non-trivial for $q = \Omega(n)$.

In Section 3.1 we introduce a technique that was first used by Kurosawa and Suzuki in [KS08]: the use of a *maximal set of linearly independent vectors modulo a code.* We also give a more general approach so it can be applied to generic codes. Then, in Section 3.2 we present a basic 3-round protocol on which the new protocol will be based. In Section 3.3 we present the new notion of *block-maximal* set of linearly independent vectors modulo a code. Finally, we introduce our protocol step by step, giving formal proofs for every detail including privacy, reliability, communication cost and communication overhead. We also generalize from the setting with $n = 2t+1$ to the setting with $n = 2t+b$ with $b \in \mathbb{Z}$. t will always represent the size of the adversary $\mathcal{A}$.

3.1 Linear Dependence Modulo a Code

In this section we introduce the concepts of *linear dependence modulo a code*, based on the work of Kurosawa and Suzuki [KS08].

Suppose that we have a linear code C of length n, and we transmit a set of codewords $\mathcal{X} = \{\vec{c}_1, \dots, \vec{c}_r\}$. Assume that a t-adversary $\mathcal{A}$ is able to corrupt all the entries at the positions contained in $I_{\mathcal{A}} = \{i_1, \dots, i_t\} \subset \{1, \dots, n\}$. Let $\mathcal{D} = \{\vec{d}_1, \dots, \vec{d}_r\}$ be the set of received codewords.

In general, some of the $\vec{d}_i$'s will not belong to C. In particular, if the minimum distance of C is $d > t$ (as it will be in our applications) then any modification made by $\mathcal{A}$ in a vector $\vec{c}_i$ will result in a vector $\vec{d}_i \notin C$. Indeed, $\mathcal{A}$ can modify up to t entries, hence $d(\vec{d}_i, \vec{c}_i) \leq t < d$.

The error-vectors $\vec{e}_i = \vec{d}_i - \vec{c}_i$ for $i \in \{1, \dots, r\}$ span a subspace $E \subset \mathbb{F}_q^n$ called the *error-vector space.* The following lemma is straightforward to prove.

Lemma 3.1. $\dim(E) \leq t$.

Proof. Since $\mathcal{A}$ can modify only the entries in the positions in $I_{\mathcal{A}}$, it follows that all the non-zero entries of every $\vec{e}_i$ will be in positions contained in $I_{\mathcal{A}}$. For every $j \in \{1, \dots, n\}$ we denote $\vec{\epsilon}_j$ as the vector with 1 on the j-th position and 0 on every other position; then we have that E is contained in the vector space spanned by $\{\vec{\epsilon}_{i_1}, \dots, \vec{\epsilon}_{i_t}\}$, which has dimension t. □

Moreover, it is clear that we have $E \cap C = \{\vec{0}\}$. We are interested in finding a minimal subset $\{\vec{d}_{i_1}, \dots, \vec{d}_{i_a}\} \subseteq \mathcal{D}$ with $a \leq t$ so that:

$$\langle \vec{e}_{i_1}, \dots, \vec{e}_{i_a} \rangle = E.$$

Definition 3.2. *Let C be a linear code of length n over $\mathbb{F}_q$, and let $\vec{v}_1, \dots, \vec{v}_s \in \mathbb{F}_q^n \setminus \{\vec{0}\}$. Then we say that $\vec{v}_1, \dots, \vec{v}_s$ are* linearly independent *modulo C if any relation of the type:*

$$\sum_{i=1}^{s} \alpha_i \vec{v}_i = \vec{c},$$

with $\alpha_1, \dots, \alpha_s \in \mathbb{F}_q$ and $\vec{c} \in C$ holds if and only if $\alpha_1 = \cdots = \alpha_s = 0$ and $\vec{c} = \vec{0}$.

We can now prove the following simple fact, which will be very useful later on.

Theorem 3.3. *Let B be a maximal set of vectors of $\mathcal{D}$ that are linearly independent modulo C. Then $|B| = \dim(E)$.*

Proof. Let $B = \{\vec{d}_{i_1}, \dots, \vec{d}_{i_a}\}$ and consider the vectors $\vec{e}_{i_j} = \vec{d}_{i_j} - \vec{c}_{i_j}$ for $j \in \{1, \dots, a\}$. Then trivially also the $\vec{e}_{i_j}$ are linearly independent modulo C. Since the subspace spanned by the $\vec{e}_{i_j}$ is contained in E, it follows that $|B| \leq \dim(E)$.

Conversely, let $\{\vec{e}_{i_1}, \dots, \vec{e}_{i_s}\}$ be a basis of E contained in $\{\vec{e}_1, \dots, \vec{e}_r\}$; then, for $j \in \{1, \dots, s\}$ we have $\vec{e}_{i_j} = \vec{d}_{i_j} - \vec{c}_{i_j}$, with $\vec{d}_{i_j} \in \mathcal{D}$ and $\vec{c}_{i_j} \in C$. Let $\alpha_1, \dots, \alpha_s \in \mathbb{F}_q$ not all 0, $\vec{c} \in C$ s.t.:

$$\sum_{j=1}^{s} \alpha_j \vec{d}_{i_j} = \vec{c},$$

then:

$$\sum_{j=1}^{s} \alpha_j \vec{d}_{i_j} = \sum_{j=1}^{s} \alpha_j (\vec{e}_{i_j} + \vec{c}_{i_j}) = \sum_{j=1}^{s} \alpha_j \vec{e}_{i_j} - \sum_{j=1}^{s} \alpha_j \vec{c}_{i_j},$$

so

$$\sum_{j=1}^{s} \alpha_j \vec{e}_{i_j} = \vec{c}'$$

for some $\vec{c}' \in C$. Now, the Hamming weight of the term on the left side is less than or equal to $s \leq t$, while $\vec{c}'$ is a codeword of C, so this equality is possible only if $\vec{c}' = \vec{0}$. Since the $\vec{e}_{i_j}$ are linearly independent, it follows that $\alpha_1 = \cdots = \alpha_s = 0$. In other words, the set $\{\vec{d}_{i_1}, \ldots, \vec{d}_{i_s}\}$ is a set of vectors of $\mathcal{D}$ that are linearly independent modulo C. Hence, $\{\vec{d}_{i_1}, \ldots, \vec{d}_{i_s}\} \subseteq B$ and $|B| \geq \dim(E)$. □

From this theorem it is straightforward to see that such B is a minimal set of vectors in $\mathcal{D}$ whose error-vectors span the whole E. Indeed, as a consequence of this theorem we have that the error-vectors of any proper subset of B generate a vector space over $\mathbb{F}_q$ of dimension strictly less than $\dim(E)$.

Furthermore, by Lemma 3.1 the errors that are generated by a t-adversary's activity span a subspace of dimension at most t. In our applications we use an $[n, k]$-linear code C with minimum distance $d \geq t + 1$. It follows by the Singleton bound that $k + t \leq n$, hence the error-vector space can always be fully determined. Indeed, if we find such a set B with $|B| = t$, then we must have an error-vector space of dimension t. It follows that the condition $k + t \leq n$ is necessary and sufficient.

It only remains to state how to find the set B in practice. In [KS08] Kurosawa and Suzuki's method is based on Lagrange's Interpolation Theorem. However, in this thesis the hypothesis of the theorem on the field size is not always satisfied. Nevertheless, we can determine B with a basic linear-algebra argument as well. This remains computationally efficient even despite the lack of structure that characterizes generic codes.

Before we state the argument, we require the following definition.

Definition 3.4. *Let C be a linear code of length n over the field $\mathbb{F}_q$, H a check matrix for C and $\vec{v} \in \mathbb{F}_q^n$. Then we call the* syndrome *of $\vec{v}$ the vector $\vec{\sigma} = \vec{v} \cdot H$.*

It follows from the definition that a vector of $\mathbb{F}_q^n$ belongs to C if and only if its syndrome is $\vec{0}$. The following proposition can be used to compute B.

Proposition 3.5. *Let C be a linear code of length n over $\mathbb{F}_q$, and $\vec{v}_1, \ldots, \vec{v}_s \in \mathbb{F}_q^n$. Let also H be a check matrix for C. Then $\vec{v}_1, \ldots, \vec{v}_s$ are linearly independent modulo C if and only if their syndromes $\vec{\sigma}_1, \ldots, \vec{\sigma}_s$ are linearly independent.*

Proof. We prove equivalently that $\vec{v}_1, \ldots, \vec{v}_s$ are linearly dependent modulo C if and only if $\vec{\sigma}_1, \ldots, \vec{\sigma}_s$ are linearly dependent. Let $\alpha_1, \ldots, \alpha_s \in \mathbb{F}_q$ not all 0 such that

$$\sum_{i=1}^{s} \alpha_i \vec{v}_i - \vec{c} = \vec{0}$$

for some arbitrary $\vec{c} \in C$. Then we claim that this equality is true if and only if

$$\left(\sum_{i=1}^{s} \alpha_i \vec{v}_i - \vec{c} \right) \cdot H = \vec{0}.$$

Indeed, the first equality certainly implies the second. On the other hand, if the second equality holds then the left hand side of the first equation is equal to some $\vec{c'} \in \ker(H) = C$. Hence, since also $\vec{c'} \cdot H = \vec{0}$, both the following equalities hold at the same time:

$$\sum_{i=1}^{s} \alpha_i \vec{v}_i - \vec{c} = \vec{c'} \quad \text{and} \quad \left(\sum_{i=1}^{s} \alpha_i \vec{v}_i - \vec{c} \right) \cdot H = \vec{c'} \cdot H.$$

This clearly implies that if we replace $\vec{c}$ by $\vec{c} + \vec{c'}$ in our inequalities (and we can by the arbitrariness of $\vec{c}$) the second equality implies the first. Therefore,

$$\vec{0} = \vec{c} \cdot H = \left(\sum_{i=1}^{s} \alpha_i \vec{v}_i \right) \cdot H = \sum_{i=1}^{s} \alpha_i \vec{v}_i \cdot H = \sum_{i=1}^{s} \alpha_i \vec{\sigma}_i,$$

which means that the first equality (i.e., saying that $\vec{v}_1, \ldots, \vec{v}_s$ are linearly dependent modulo C) is equivalent to saying that $\vec{\sigma}_1, \ldots, \vec{\sigma}_s$ are linearly dependent. □

It follows from Proposition 3.5 that B can be computed in a very simple way. We can follow this simple algorithm. We initialize a set $\mathcal{B} = \emptyset$. For $i = 1, \ldots, r$ we check if the vector $\vec{\sigma}_i = \vec{d}_i \cdot H$ is a linear combination of the syndromes of the vectors in $\mathcal{B}$; if it is not, we add $\vec{d}_i$ to $\mathcal{B}$ and proceed with $\vec{d}_{i+1}$ (or we stop, if $i = r$). In the end, we output $\mathcal{B} = B$.

This algorithm only requires r linear dependency checks, each of which can be done in time polynomial in n.

3.2 A basic 3-Round Protocol

Now we present a basic 3-round protocol for the case $n = 2t + 1$, on which the new 3-round protocol we are presenting is based. We also explain the main issues in reaching the optimal communication overhead of $\frac{n}{n-2t}$.

3.2.1 Overview

This basic protocol mainly relies on the technique of finding a maximal set of linearly independent vectors modulo a code explained above in Section 3.1. The main idea is to take advantage of having a third communication round by giving complete feedback about the channels that were corrupted during the first round.

Let $q \geq n + t + 1$. As stated in the introduction we consider the case $q = \Omega(n)$ since with larger field size the problem becomes much easier to solve.

In this protocol a code C over $\mathbb{F}_q$ is used, with the following properties. The secret sharing scheme associated with C is a t-threshold scheme. We need t-privacy since each share in this scheme will be sent over a different channel, and t of these channels are controlled by $\mathcal{A}$. On the other hand, we need $(t + 1)$-reconstruction for the information reconciliation part of the protocol.

This protocol consists of four steps. In the first step, $\mathcal{S}$ sends some secret random values to $\mathcal{R}$. Then in the second step, $\mathcal{R}$ computes a maximal set of linearly independent vectors modulo the code that is used for the error-vector space generated by $\mathcal{A}$ as described in Section 3.1 and sends it to $\mathcal{S}$. In the third step $\mathcal{S}$ enciphers the message using the random values selected at the beginning of the first step. Next, he sends the enciphered message to $\mathcal{R}$ together with a description of the channels that were corrupted during the first step. Finally, in the fourth and final step $\mathcal{R}$ recovers all random data that was sent during the first step and deciphers the message.

This protocol is not efficient enough in terms of communication overhead because sending the message during the third communication round is relatively expensive. In fact, even though sending the message immediately during the first step would be less expensive, the transmission of the maximal set of linearly independent vectors modulo the code in the second step would then leak information about the message. Hence, only random data is sent during the first step to prevent leakage, at the cost of increasing the overhead to roughly $2n$.

3.2.2 The Protocol

Let C be an $[n + 1, t + 1]$-Reed-Solomon code over $\mathbb{F}_q$. By Example 2.37 the secret sharing scheme associated with C is a t-threshold scheme. Note that the set of vectors consisting of the first n positions of the codewords of C forms a code C'. C' is again a linear code, but in particular by Definition 2.11 it is also a Reed-Solomon code with length n and dimension $t + 1$. As a consequence, the secret sharing scheme associated with C' is a t-threshold scheme as well.

Suppose that $\mathcal{S}$ wants to send to $\mathcal{R}$ a message $M \in \mathbb{F}_q^{n^3}$. The details of the first communication round can be found in Figure 3.1.

$\mathcal{S}$ does the following.

- He selects $y_1, \ldots, y_{n^3+t}$ elements of $\mathbb{F}_q$ uniformly at random.
- For each $k \in \{1, \ldots, n^3+t\}$ he selects a codeword $\vec{c}^{(k)} \in C$ uniformly at random such that $c_{n+1}^{(k)} = y_k$. The vector $\vec{c}^{(k)'} = (c_1^{(k)}, \ldots, c_n^{(k)}) \in \mathbb{F}_q^n$ is obtained by restricting to the first n positions of $\vec{c}^{(k)}$
- For each $i \in \{1, \ldots, n\}$ and each $k \in \{1, \ldots, n^3+t\}$ he sends through channel i the element $c_i^{(k)}$, thus keeping secret the element $c_{n+1}^{(k)} = y_k$ to any collusion of t channels.

Figure 3.1: First communication round.

Basically, in the first round $\mathcal{S}$ samples n^3+t codewords of C uniformly at random, which each encode a uniformly random value in $\mathbb{F}_q$. The reason for the "$+t$" is that later in the protocol at most t of those codewords are discarded. Next, he sends the codewords component-wise through the communication channels.

The table below intuitively shows how encoding is done.

$c_1^{(1)}$	$c_2^{(1)}$	$\ldots$	$c_{n-1}^{(1)}$	$c_n^{(1)}$	$c_{n+1}^{(1)} = y_1$
$c_1^{(2)}$	$c_2^{(2)}$	$\ldots$	$c_{n-1}^{(2)}$	$c_n^{(2)}$	$c_{n+1}^{(2)} = y_2$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
$c_1^{(n^3+t-1)}$	$c_2^{(n^3+t-1)}$	$\ldots$	$c_{n-1}^{(n^3+t-1)}$	$c_n^{(n^3+t-1)}$	$c_{n+1}^{(n^3+t-1)} = y_{n^3+t-1}$
$c_1^{(n^3+t)}$	$c_2^{(n^3+t)}$	$\ldots$	$c_{n-1}^{(n^3+t)}$	$c_n^{(n^3+t)}$	$c_{n+1}^{(n^3+t)} = y_{n^3+t}$
n shares					1 secret

Assume now that $\mathcal{R}$ received the vectors $\vec{d}^{(k)} \in \mathbb{F}_q^n$ at the end of the first round. Figure 3.2 shows the details of the second round.

As we saw in Section 3.1 the maximal set B of linearly independent vectors modulo C' fully determines the error-vector space spanned by the errors carried by the vectors $\vec{d}^{(k)}$. $\mathcal{R}$ computes B and then broadcasts it.

Figure 3.3 shows how $\mathcal{S}$ acts in the third (and last) communication round. Here we assume that $B = \{\vec{d}^{(k_1)}, \ldots, \vec{d}^{(k_r)}\}$, with $r \leq t$.

$\mathcal{R}$ does the following.

- As in Section 3.1 he computes a maximal set B of linearly independent vectors modulo C' for the whole error-vector space.
- He broadcasts B.

Figure 3.2: Second communication round.

$\mathcal{S}$ does the following.

- He computes a set of indices I consisting of all the non-zero entries of the vectors $\vec{d}^{(k_\ell)} - \vec{c}^{(k_\ell)'}$ for $\ell \in \{1, \dots, r\}$.
- He defines the set $J \subseteq \{1, \dots, n^3+t\}$ as the first n^3 indices k, according to the usual order on the natural numbers, of the set $\{1, \dots, n^3 + t\} \setminus \{k_1, \dots, k_r\}$.
- He joins all the values y_k with $k \in J$, obtaining a one-time-pad key K of length n^3 over $\mathbb{F}_q$.
- He broadcasts I and the message M enciphered with K.

Figure 3.3: Third communication round.

$\mathcal{S}$ first determines all channels that have been corrupted by $\mathcal{A}$ during the first round. Then he discards those secret values corresponding to the vectors in the set B and uses the others to form a secret key K. He then enciphers the message using K and sends it across.

Finally, $\mathcal{R}$ acts as in Figure 3.4 in order to recover the message M.

$\mathcal{R}$ does the following.

- He computes the set J.
- He considers the vectors $(\vec{d}^{(k)}, 0) \in \mathbb{F}_q^{n+1}$ for $k \in J$, and erases the entries corresponding to the indices in $I \cup \{n+1\}$.
- He corrects those erasures, obtaining the vectors $\vec{c}^{(k)}$ and in particular the values y_k with $k \in J$.
- He computes K, deciphers the enciphered message and obtains M.

Figure 3.4: Message decoding by $\mathcal{R}$

In this final step $\mathcal{R}$ erases all data corresponding to the channels that were corrupted by $\mathcal{A}$ during the first communication round on the vectors $\vec{c}^{(k)\prime}$ with $k \in J$. This allows him to recover the random data that was initially sent by $\mathcal{S}$. He then computes the secret key K, which allows him to decipher the encrypted message.

This concludes the protocol. In the next section we prove that this protocol reaches perfect privacy and perfect reliability, with an overhead of $2n$.

3.2.3 Proofs

First, we claim that with this simple protocol $\mathcal{R}$ is able to compute M reliably.

Proposition 3.6. *At the end of every protocol execution $\mathcal{R}$ outputs a message M' with $P(M' = M) = 1$.*

Proof. First of all, we recall that the vector $\vec{c}^{(k)\prime}$ is a codeword of C'. $\mathcal{A}$ has size at most t, hence in the first communication round he can corrupt up to t entries for each codeword $\vec{c}^{(k)\prime}$. Therefore, for every k we have $d(\vec{d}^{(k)}, \vec{c}^{(k)\prime}) \leq t$. Since $d(C') = n - (t+1) + 1 = t + 1$, this implies that if $\mathcal{A}$ corrupted any data during the transmission of $\vec{c}^{(k)\prime}$ we have that $\vec{d}^{(k)} \notin C'$. So, the errors carried by the vectors $\vec{d}^{(k)}$ span an error-vector space E of dimension at most t. The maximal set of linearly independent vectors modulo C' that is computed by $\mathcal{R}$ is hence well-defined and gives complete information about all the positions of the corruptions that were made by $\mathcal{A}$ during the first communication round.

As a consequence, the set I that $\mathcal{S}$ computes in the third round contains all the channels that have been corrupted by $\mathcal{A}$ and clearly $|I| \leq t$. When $\mathcal{R}$ receives this set I, he can erase all the entries contained in $I \cup \{n+1\}$ from the vectors $(\vec{d}^{(k)}, 0) \in \mathbb{F}_q^{n+1}$. He still has at least $t+1$ correct entries of each codeword $\vec{c}^{(k)}$, and the reconstruction threshold of C is $t+1$. Therefore, he can recover each codeword $\vec{c}^{(k)}$ and extract the secret $c_{n+1}^{(k)} = y_k$.

$\mathcal{R}$ now has all the values y_k with $k \in J$ and from those he can compute the secret key K, and hence he can reliably output the message M. □

Now, we prove the privacy of the protocol.

Proposition 3.7. *After every protocol execution the view of $\mathcal{A}$ is independent of the message M.*

Proof. After the first communication round $\mathcal{A}$ knows t entries of each codeword $\vec{c}^{(k)}$. However, the privacy threshold of C is t, so he cannot compute the secret element $y_k = c_{n+1}^{(k)}$. After the second communication round, instead, he can compute all the codewords $\vec{c}^{(k_\ell)}$ for $\ell \in \{1, \ldots, r\}$ corresponding to the vectors in the maximal set of linearly independent vectors modulo C'. Hence, he knows the values $y_{k_1}, \ldots, y_{k_r}$. Since these values are discarded, the secret key K is perfectly secure. □

Finally, it is easy to see that at most $2n^4 + tn^2 + tn$ field elements are sent in this protocol in order to transmit a message of size n^3. Hence $2n$ is an approximate upper bound to the communication overhead of the protocol.

This completes the study of this basic 3-round protocol. The basic 3-round protocol presented here is close to optimal, even though it is very simple. However, if we want to actually reach optimality we have to overcome a major problem: after the sending of the maximal set of linearly independent vectors modulo C', $\mathcal{A}$ can have information about up to t values y_k. Since we do not know a priori which values y_k are compromised, we must discard them a posteriori. This implies that only randomness can be sent during the first communication round, doubling the communication overhead. So, if we want to send the message during the first communication round we need a technique which allows $\mathcal{S}$ and $\mathcal{R}$ not to discard any value but still keep the message private. The technique we use involves *block-maximal* sets of linearly independent vectors modulo a code and is introduced in the next section.

3.3 Block-Maximal Sets of Linearly Independent Vectors Modulo a Code

Suppose we divide our codewords $\vec{c}^{(1)}, \ldots, \vec{c}^{(n^3)}$ into n^2 blocks of n vectors. From here on, vectors will be denoted as $\vec{v}^{(j,k)}$, where:

- j is the index of the block that contains the vector;
- k is the index of the vector inside the the j-th vector block.

The ℓ-th component of a vector $\vec{v}^{(j,k)}$ will be denoted as $\vec{v}^{(j,k)}_\ell$.

First we explain what we mean when we say that a block j is *involved* by a maximal set of linearly independent vectors modulo C'.

Definition 3.8. *Let $\mathcal{D} = \{\vec{d}^{(j,k)} \mid j \in \{1, \ldots, n^2\},\ k \in \{1, \ldots, n\}\}$ and let $B = \{\vec{d}^{(j_1,k_1)}, \ldots, \vec{d}^{(j_r,k_r)}\}$ with $r \leq t$, $\{j_1, \ldots, j_r\} \subset \{1, \ldots, n^2\}$, $\{k_1, \ldots, k_r\} \subset \{1, \ldots, n\}$ be a maximal set of linearly independent vectors modulo C'. Then the block j is said to be* involved *by B if $j \in \{j_1, \ldots, j_r\}$.*

Next, we introduce an ordering for these maximal sets.

Definition 3.9. *Let B_1, B_2 be two maximal sets of linearly independent vectors modulo C'. Then we say that $B_1 \preccurlyeq B_2$ if:*

$$\{j \mid j \textit{ is involved by } B_1\} \subseteq \{j \mid j \textit{ is involved by } B_2\}$$

It is easy to see that $\preccurlyeq$ is a partial order.

Definition 3.10. *A maximal set B of linearly independent vectors modulo C' is said to be* block-maximal *if it is maximal with respect to the partial order $\preccurlyeq$.*

A block-maximal set of linearly independent vectors modulo C' always exists because every set of involved blocks is contained in the set $\{1, \dots, n^2\}$. In our applications the upper bound will be even smaller.

For every maximal set B of linearly independent vectors modulo C' we introduce three definitions:

- We denote by $R(B)$ the set $\{j \mid j \text{ is involved by } B\}$;
- We denote by $S(B) \subseteq R(B)$ the set

  $$\{j \in R(B) \text{ s.t. } |\{k \text{ s.t. } \vec{d}^{(j,k)} \in B\}| = 1\}.$$

  In other words, $S(B)$ is the set of blocks that are involved only once by B;
- We denote by $B' \subseteq B$ the set of vectors $\{\vec{d}^{(j,k)} \in B \text{ s.t. } j \in S(B)\}$. B' will be called the *reduced* block-maximal set of linearly independent vectors modulo C'.

We now prove a very important property about block-maximal sets of linearly independent vectors modulo C'.

Proposition 3.11. *Let B be a block-maximal set of linearly independent vectors modulo C' for the whole error-vector space. Suppose that $j \notin R(B)$ and let E_j be the error-vector space spanned over C' by the vectors $\vec{d}^{(j,k)}$ for $k \in \{1, \dots, n\}$. Then a maximal set of linearly independent vectors modulo C' for E_j is contained in B'.*

Proof. Assume by contradiction that no maximal set of linearly independent vectors over C' of E_j is contained in B'. Let $\langle B' \rangle$ be the error-vector space spanned by the errors carried by the vectors in B'. Then:

$$\dim(E_j) - \dim(E_j \cap \langle B' \rangle) = a \geq 1.$$

Hence, there are a vectors $\vec{d}^{(j,k_1)}, \ldots, \vec{d}^{(j,k_a)}$ that, when added to a maximal set of linearly independent vectors modulo C' for $E_j \cap \langle B' \rangle$, form a maximal set of linearly independent vectors modulo C' for E. These vectors will be linearly independent modulo C' with all the vectors in B' as well. Now, since B is a maximal set of linearly independent vectors modulo C' for the whole error-vector space, we have that a maximal set of linearly independent vectors modulo C' for E_j must be contained in B. Hence, we can replace a vector in $B \setminus B'$ with one of the vectors $\vec{d}^{(j,k_1)}, \ldots, \vec{d}^{(j,k_a)}$, obtaining an equivalent maximal set $\hat{B}$ of linearly independent vectors modulo C' for the whole error-vector space. As a consequence, $R(\hat{B}) = R(B) \cup \{j\}$, which contradicts the block-maximality of B. □

This proposition not only states an important property of block-maximal set of linearly independent vectors modulo C' that will be fundamental later on in describing the protocol, but it also gives an idea of how to compute such a block-maximal set of linearly independent vectors modulo C'.

Suppose that $\mathcal{R}$ wants to compute a block-maximal set of linearly independent vectors modulo C' for the whole error-vector space. Then he can use the following algorithm:

- He computes a (generic) maximal set B of linearly independent vectors modulo C' with the method given in Section 3.1;
- He computes $R(B)$, $S(B)$ and B';
- For every $j \in \{1, \ldots, n^2\} \setminus R(B)$, he does the following:
    - He computes the error-vector space E_j spanned by the errors carried by the vectors $\vec{d}^{(j,k)}$;
    - If $E_j \subseteq \langle B' \rangle$ and j is the maximum in the set $\{1, \ldots, n^2\} \setminus R(B)$, he outputs B, $R(B)$, $S(B)$ and B';
    - If $E_j \subseteq \langle B' \rangle$ and j is not the maximum in the set $\{1, \ldots, n^2\} \setminus R(B)$, he passes to the next j;
    - If E_j is not contained in $\langle B' \rangle$, he chooses a vector $\hat{\vec{d}}^{(j,k)}$ s.t. $\langle \hat{\vec{d}}^{(j,k)} \rangle \subseteq E_j \setminus (E_j \cap \langle B' \rangle)$. Then, he replaces a vector in $B \setminus B'$ with $\hat{\vec{d}}^{(j,k)}$ in such a way that the obtained set is still a maximal set of linearly independent vectors modulo C' for the whole error-vector space. After this, he replaces B with this new set, he computes the new $R(B)$, $S(B)$ and B' and moves on to the next j.

All the steps in this algorithm work in polynomial time and since the number of j's is polynomial in n we have that the overall computational cost of the algorithm is polynomial. The key reason is that for our protocol we require the block-maximal set of linearly independent vectors modulo C' to be *maximal* and not *maximum* with respect to the partial order $\preccurlyeq$.

3.4 The Optimal 3-Round Protocol

With the new technique introduced in Section 3.3 we are able to present our optimal 3-round protocol. Recall that this protocol is communication-optimal, in the sense that as the message size increases its communication overhead is (asymptotically) $\frac{n+1}{n-2t}$. In Section 3.4.3 we prove that this protocol is perfectly private and in Section 3.4.4 we prove that it is perfectly reliable. The communication overhead is analyzed in Section 3.4.5. Finally, a generalization in the case of $n = 2t+b$ with $b \in \mathbb{Z}$ is presented in Section 3.4.6.

3.4.1 Overview of the Protocol

Let $n = 2t + 1$ and assume that $q \geq n + 1$. As in the basic 3-round protocol, here just two codes are necessary for our purposes: the codes C and C', defined precisely as in Section 3.2. Indeed, we just need their privacy and reconstruction properties.

The main idea in this protocol is to have $\mathcal{S}$ break the message into several message blocks of length n. In order to prevent the leakage caused by the transmission of the block-maximal set of linearly independent vectors modulo the code in the second communication round, he first encodes each message block with 1-privacy. Then, he sends the enciphered message blocks across during the first communication round encoded as secret vectors. Since $\mathcal{R}$ sends a reduced block-maximal set of linearly independent vectors modulo the code in the second communication round, 1-privacy is enough to prevent leakage on the message in the involved blocks. The blocks which are not involved instead give a perfectly secure one-time pad which is used to encipher and send again the involved message blocks. The cost of this last operation is relatively low as long as the involved blocks are only a small fraction of the total number. Hence, the only communication round during which a significant number of field elements is sent is the first round. This allows the communication rate to be (asymptotically) $n+1$, which is equal to the lower bound of $\frac{n+1}{n-2t}$, as the message size increases.

3.4.2 The Protocol

We define two Reed-Solomon codes C, C' exactly as the codes C, C' in Section 3.2 and with the same relation between them: the first n positions of each codeword of C form a codeword of C' and every codeword of C' consists of the first n positions of a codeword of C. The secret sharing schemes associated with these codes are both t-threshold schemes.

Let $\gamma \in \mathbb{Z}$, $\gamma \geq 3$ and suppose that $\mathcal{S}$ wants to send to $\mathcal{R}$ a message M of size $L = n^\gamma$ over $\mathbb{F}_q$. In the first communication round $\mathcal{S}$ acts as in Figure 3.5.

$\mathcal{S}$ does the following.

First, he breaks the message M into $n^{\gamma-1}$ vectors $\vec{m}^{(1)}, \ldots, \vec{m}^{(n^{\gamma-1})} \in \mathbb{F}_q^n$, where each $\vec{m}^{(j)}$ will be called the *j-th message block*. Then, for every $j \in \{1, \ldots, n^{\gamma-1}\}$ he does the following:

- He selects $y_j \in \mathbb{F}_q$ uniformly at random;
- He initializes the vector $\vec{y}^{(j)} = (y_j, \ldots, y_j) \in \mathbb{F}_q^n$;
- He initializes the vector $\vec{x}^{(j)} = (\vec{m}^{(j)} + \vec{y}^{(j)}, y_j) \in \mathbb{F}_q^{n+1}$. This vector is called the *j-th encoded block*;
- He finds for every $k \in \{1, \ldots, n+1\}$ a codeword $\vec{c}^{(j,k)} \in C$ s.t. $c_{n+1}^{(j,k)} = x_k^{(j)}$. From here on, vectors will be denoted as in Section 3.3.
- For every k he defines $\vec{c}^{(j,k)\prime} = (c_1^{(j,k)}, \ldots, c_n^{(j,k)}) \in C'$.
- For every $i \in \{1, \ldots, n\}$ and for every $k \in \{1, \ldots, n+1\}$ he sends through channel i the element $c_i^{(j,k)}$, keeping secret all the values $c_{n+1}^{(j,k)} = x_k^{(j)}$.

Figure 3.5: First communication round.

What $\mathcal{S}$ actually does is to encode each message block $\vec{m}^{(j)}$ into a vector $\vec{x}^{(j)}$ with 1-privacy. The method can be seen as a Caesar cipher, with the key of the cipher as the last element of the vector. He then sends each element of each encoded block as a secret in the secret sharing scheme associated with C.

The table below shows intuitively how the encoding of a message block $\vec{m}^{(j)}$ is done.

$c_1^{(j,1)}$	$c_2^{(j,1)}$	$\dots$	$c_{n-1}^{(j,1)}$	$c_n^{(j,1)}$	$c_{n+1}^{(j,1)} = x_1^{(j)} = m_1^{(j)} + y_j$
$c_1^{(j,2)}$	$c_2^{(j,2)}$	$\dots$	$c_{n-1}^{(j,2)}$	$c_n^{(j,2)}$	$c_{n+1}^{(j,2)} = x_2^{(j)} = m_2^{(j)} + y_j$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
$c_1^{(j,n)}$	$c_2^{(j,n)}$	$\dots$	$c_{n-1}^{(j,n)}$	$c_n^{(j,n)}$	$c_{n+1}^{(j,n)} = x_n^{(j)} = m_n^{(j)} + y_j$
$c_1^{(j,n+1)}$	$c_2^{(j,n+1)}$	$\dots$	$c_{n-1}^{(j,n+1)}$	$c_n^{(j,n+1)}$	$c_{n+1}^{(j,n+1)} = x_{n+1}^{(j)} = y_j$
n shares					1 secret

Suppose now that $\mathcal{R}$ received the vectors $\vec{d}^{(j,k)}$ for $j \in \{1, \dots, n^{\gamma-1}\}$ and $k \in \{1, \dots, n+1\}$. Then in the second communication round $\mathcal{R}$ acts as in Figure 3.6.

$\mathcal{R}$ does the following.

- He computes a *block-maximal* set B of linearly independent vectors modulo C' for the whole error-vector space as explained in Section 3.3. In general, $B = \{\vec{d}^{(j_1,k_1)}, \dots, \vec{d}^{(j_r,k_r)}\}$ with $r \leq t$, $\{j_1, \dots, j_r\} \subset \{1, \dots, n^{\gamma-1}\}$, $\{k_1, \dots, k_r\} \subset \{1, \dots, n+1\}$. He also computes $R = R(B)$, $S = S(B)$ and B'.

- He broadcasts the sets B', R and S.

Figure 3.6: Second communication round.

In other words, $\mathcal{R}$ computes a block-maximal set of linearly independent vectors modulo C' as in Section 3.3. As we said above, since the blocks involved by the reduced block-maximal set of linearly independent vectors modulo C' are involved by only one vector, the 1-privacy provided by the vectors $\vec{x}^{(j)}$ is enough to prevent leakage on the message.

In the third communication round $\mathcal{S}$ acts as in Figure 3.7.

In this step $\mathcal{S}$ computes all the channels that were corrupted by $\mathcal{A}$ during the first communication round on the blocks j with $j \notin R$. He then broadcasts this set of channels. Afterwards, if this set of channels is not enough to determine all the channels corrupted by $\mathcal{A}$ also for the blocks j with $j \in R$, he also sends those message blocks again using broadcast, where the y_j with $j \notin R$ are used as a key for a one-time pad encryption.

Finally in order to output the message M, $\mathcal{R}$ acts as in Figure 3.8.

In this last step, first $\mathcal{R}$ recovers all the codewords $\vec{c}^{(j,k)}$ with $j \notin R$ by erasing the channels contained in the set I. From them, he is able to compute all the message blocks $\vec{m}^{(j)}$ with $j \notin R$. If the channels contained in I are not enough to do the same operations for the remaining blocks, then during the third communication round he received those message blocks

$\mathcal{S}$ does the following.

- He determines the channels corrupted by $\mathcal{A}$ during the first phase on the j-th vector blocks with $j \notin R$ by looking at the non-zero entries of the vectors $\vec{c}^{(j_s,k_s)'} - \vec{d}^{(j_s,k_s)}$, obtaining the set of channels $I = \{i_1, \dots, i_a\}$ with $a \leq t$;
- He broadcasts the set I;
- If $a \neq t$ and $|R| \neq |S|$ then he also does the following:
  - He chooses the first $|R|n$ indices in $\{1, \dots, n^{\gamma-1}\} \setminus R$, say $\{j_1, \dots, j_{|R|n}\}$;
  - He divides the vector $(y_{j_1}, \dots, y_{j_{|R|n}})$ into $|R|$ vectors $\vec{z}^{(1)}, \dots, \vec{z}^{(|R|)} \in \mathbb{F}_q^n$;
  - For every $j_h \in R$ he broadcasts the message $\vec{m}^{(j_h)}$ using the vector $\vec{z}^{(h)}$ as a key for a one-time-pad encryption.

Figure 3.7: Third communication round.

$\mathcal{R}$ outputs M. To do so, he does the following for every $j \notin R$.

- He considers the vectors $(\vec{d}^{(j,k)}, 0) \in \mathbb{F}_q^{n+1}$, and erases the entries corresponding to the indices contained in $I \cup \{n+1\}$.
- He corrects those erasures, obtaining the vectors $\vec{c}^{(j,k)}$ and in particular the values $x_k^{(j)}$.
- He recovers $\vec{m}^{(j)}$ and y_j from $\vec{x}^{(j)}$.

After that, he recovers the remaining part of the message:

- He computes all the vectors $\vec{z}^{(h)}$;
- He recovers all the $\vec{m}^{(j_h)}$ with $j_h \in R$ by subtracting $\vec{z}^{(h)}$ from the received encrypted messages.

Figure 3.8: Message reconstruction by $\mathcal{R}$

again, enciphered with the y_j with $j \notin R$. After computing these values, he can decipher those message blocks as well.

This concludes the protocol. In the following sections we provide all the proofs.

3.4.3 Privacy of the Protocol

In this section we show in detail that our protocol is perfectly private, i.e., that in every step the adversary does not even get partial information about the message M.

After the first communication round $\mathcal{A}$ knows at most t positions of each vector $\vec{c}^{(j,k)'}$ that is sent by $\mathcal{S}$. Hence, since the secret sharing scheme associated with C' has t-privacy, he cannot recover any vector that was sent. In particular, he cannot recover any of the elements $c_{n+1}^{(j,k)} = x_k^{(j)}$ (and hence the encoded blocks) either.

We now show that the technique of using a block-maximal set of linearly independent vectors modulo C' allows $\mathcal{R}$ to send back to $\mathcal{S}$ some vectors $\vec{d}^{(j,k)}$ without leaking information about the message. Indeed, during the second round $\mathcal{A}$ gets to know B', R and S. The latter two are two sets of indices which are independent of M. We should prove, therefore, that he gains no partial knowledge about the message M during the transmission of B'.

Let $\vec{d}^{(j,k)} \in B'$, i.e., $j \in S$. Then, by the definition of S, the vector block j is involved by B' only by the vector $\vec{d}^{(j,k)}$.

From the vector $\vec{d}^{(j,k)}$ $\mathcal{A}$ learns the codeword $\vec{c}^{(j,k)}$ and hence the value $c_{n+1}^{(j,k)} = x_k^{(j)}$. However:

- if $k = n+1$, then $x_{n+1}^{(j)} = y_j$, which is a random element of $\mathbb{F}_q$ completely unrelated with the j-th message block;
- if $k \neq n+1$, then $x_k^{(j)} = m_k^{(j)} + y_j$. Hence, for the adversary it is impossible to learn the value $m_k^{(j)}$ without first guessing y_j.

In other words, B' is uncorrelated with the message.

Finally, during the third round $\mathcal{A}$ learns the set I (which is already known by $\mathcal{A}$) and the vectors $\vec{m}^{(j_h)} + \vec{z}^{(h)}$. To prove perfect privacy, we need to prove that $\mathcal{A}$ has no knowledge about the vectors $\vec{z}^{(h)}$. Now, we saw above that after the first communication round $\mathcal{A}$ has no knowledge about any of the y_j. During the second communication round he might however learn up to t values y_j: precisely those with $j \in S$. Since by definition the j's used for the vectors $\vec{z}^{(1)}, \ldots, \vec{z}^{(|R|)}$ are not in S, we can conclude that $\mathcal{A}$ has no knowledge about them.

This proves that our 3-round protocol is perfectly secure.

3.4.4 Reliability of the Protocol

In this section we prove that our protocol is perfectly reliable, i.e., that at the end of the protocol execution $\mathcal{R}$ outputs a message M' with $P(M' = M) = 1$. We do this by proving that every step is well-defined.

Assume that $\mathcal{R}$ received the vectors $\vec{d}^{(j,k)}$ after the first communication round.

Remark. For all j, k the vector $\vec{c}^{(j,k)\prime} = (c_1^{(j,k)}, \ldots, c_n^{(j,k)})$ is a codeword of C'. Suppose that $\mathcal{A}$ corrupted some of these entries. It follows that since the size of $\mathcal{A}$ is t, we have that $d(\vec{c}^{(j,k)\prime}, \vec{d}^{(j,k)}) \leq t < d(C')$; hence, $\vec{d}^{(j,k)} \notin C'$. Hence, the vectors $\vec{d}^{(j,k)}$ define an error-vector space E of dimension less than or equal to t and a (block-) maximal set of linearly independent vectors modulo C' for it can be found.

During the third round $\mathcal{S}$ again sends some encrypted parts of M only if $a \neq t$ or $|R| \neq |S|$. Indeed, if $a = t$ or $|R| = |S|$ then there is no need for this last part. If $a = t$ then I contains all the channels controlled by the adversary, hence $\mathcal{R}$ will be able to correct the errors that occurred during the first phase on all the vector blocks, including those with $j \in R$. On the other hand, if $|R| = |S|$ then since $|B| \geq |R| \geq |S| = |B'|$ we have that $B = B'$ and B' is a maximal set of linearly independent vectors modulo C' for the whole error-vector space. Hence, also in this case the set I is enough for $\mathcal{R}$ to determine and correct all the errors that occurred during the first phase. Observe now that since $|R| \leq t$ by definition, the cardinality of the set $\{1, \ldots, n^{\gamma-1}\} \setminus R$ is at least $n^{\gamma-1} - t > |R|n$. Therefore the vectors $\vec{z}^{(1)}, \ldots, \vec{z}^{(|R|)}$ are well defined.

We now conclude this analysis by proving that with all the information he received, $\mathcal{R}$ is able to output M. With the data received during the third communication round, $\mathcal{R}$ for every $j \in \{1, \ldots, n^{\gamma-1}\} \setminus R$ can do the following:

- He considers the vectors $(\vec{d}^{(j,k)}, 0) \in \mathbb{F}_q^{n+1}$ and erases the entries corresponding to the indices in $I \cup \{n+1\}$. Since the minimum distance of C is $(n+1) - (t+1) + 1 = t + 2$ he can correct those erasures and recover the codewords $\vec{c}^{(j,k)}$. In particular, he obtains all the elements $c_{n+1}^{(j,k)}$, and hence the encoded blocks $\vec{x}^{(j)}$.
- He decodes the message block $\vec{m}^{(j)}$ from the equality $\vec{x}^{(j)} = (\vec{m}^{(j)} + \vec{y}^{(j)}, y_j)$.

Hence, now $\mathcal{R}$ has all the message blocks $\vec{m}^{(j)}$ for $j \notin R$.

Now, if $|I| = t$ or $|S| = |R|$ then as we saw previously the information he has about the adversary is enough to perform the same operations also on the j-th vector blocks for $j \in R$ and he can therefore recover the entire message.

If $a \neq t$ and $|S| \neq |R|$ instead, the messages $\vec{m}^{(j)}$ for $j \in R$ are easily recovered by subtracting the corresponding $\vec{z}^{(j)}$'s from the vectors received during broadcast. We remark that indeed, $\mathcal{R}$ has all the y_j's needed to compute the vectors $\vec{z}^{(j)}$. Therefore, the whole message can be received with perfect correctness.

3.4.5 Communication Overhead of the Protocol

We now analyze the communication overhead of the protocol.

- During the first communication round $n^\gamma + n^{\gamma-1}$ codewords of length n are sent for an amount of $n^{\gamma+1} + n^\gamma$ field elements.
- During the second communication round up to t codewords of length n and up to $2t$ indices are broadcast, for an amount of $tn^2 + 2tn$ field elements.
- During the third communication round we have two possibilities.
    - If $a = t$ or $|S| = |R|$, then only a set of indices of size at most t is broadcast, for an amount of up to tn field elements;
    - If $a \neq t$ and $|S| \neq |R|$ then up to t vectors of length n and up to t indices are broadcast, for an amount of $tn^2 + tn$ field elements.

Therefore it is easy to see that we reach the worst case when $a \neq t$ and $|S| \neq |R|$. In this case we have that in total $n^{\gamma+1} + n^\gamma + 2tn^2 + 3tn = n^{\gamma+1} + n^\gamma + n^3 + (t-1)n$ field elements were sent in order to send a message of size $L = n^\gamma$. Hence, the information rate is

$$n + 1 + \frac{1}{n^{\gamma-3}} - \frac{1}{2n^{\gamma-2}} = n + 1 + O\left(\frac{1}{n^{\gamma-3}}\right).$$

It is clear that as γ grows, this quantity asymptotically tends to $n + 1 = \frac{n+1}{1} = \frac{n+1}{n-2t}$.

3.4.6 The Case $n = 2t + b$

The construction of our optimal 3-round protocol can be easily generalized to the case $n = 2t + b$, with $b \in \mathbb{Z}$. Note that b can be linear in n, i.e., $b = \varepsilon t$ for some $0 < \varepsilon < 1$. Suppose that $q \geq n + b + 1$.

Fix $n + b$ elements $\alpha_1, \ldots, \alpha_{n+b} \in \mathbb{F}_q$. Define C to be an $[n + b, t + b]$-Reed-Solomon code over $\mathbb{F}_q$. As we saw in Example 2.41, a secret sharing scheme associated with C with secrets of size b is a $(t, t + b)$-ramp scheme.

Now, the set of vectors consisting of the first n positions of the codewords of C forms a code C' which can be proven to be an $[n, t + b]$-Reed-Solomon code. The secret sharing scheme associated with C' also has t-privacy and $(t+b)$-reconstruction.

Let $\gamma \in \mathbb{Z}$, $\gamma \geq 3$ as before and suppose that $\mathcal{S}$ wants to send to $\mathcal{R}$ a message M of size $L = bn^{\gamma}$ over $\mathbb{F}_q$. The protocol proceeds exactly as in the case $n = 2t+1$, with the following differences.

- At first, M is divided into $bn^{\gamma-1}$ message blocks of length n. The definition of the encoded blocks does not change.

- Every j-th vector block contains b encoded blocks, in the following way. Consider the k-th position of b encoded blocks, say $x_k^{(j_1)}, \dots, x_k^{(j_b)}$. Then $\mathcal{S}$ selects a polynomial $f \in \mathbb{F}_q[x]$ of degree at most $t+b-1$ such that $f(\alpha_{n+1}) = x_k^{(j_1)}, \dots, f(\alpha_{n+b}) = x_k^{(j_b)}$. This polynomial will define the codeword $\vec{c}^{(j,k)}$. The last b positions of the codewords are kept secret, and the first communication round proceeds as before.

  The table below shows how encoding is performed in this setting for each block j. Here in the notation we omit the index "j" for simplicity.

$c_1^{(1)}$	$c_2^{(1)}$	$\dots$	$c_{n-1}^{(1)}$	$c_n^{(1)}$	$c_{n+1}^{(1)} = x_1^{(j_1)}$	$\dots$	$c_{n+b}^{(1)} = x_1^{(j_b)}$
$c_1^{(2)}$	$c_2^{(2)}$	$\dots$	$c_{n-1}^{(2)}$	$c_n^{(2)}$	$c_{n+1}^{(2)} = x_2^{(j_1)}$	$\dots$	$c_{n+b}^{(2)} = x_2^{(j_b)}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$c_1^{(n)}$	$c_2^{(n)}$	$\dots$	$c_{n-1}^{(n)}$	$c_n^{(n)}$	$c_{n+1}^{(n)} = x_n^{(j_1)}$	$\dots$	$c_{n+b}^{(n)} = x_n^{(j_b)}$
$c_1^{(n+1)}$	$c_2^{(n+1)}$	$\dots$	$c_{n-1}^{(n+1)}$	$c_n^{(n+1)}$	$c_{n+1}^{(n+1)} = x_{n+1}^{(j_1)}$	$\dots$	$c_{n+b}^{(n+1)} = x_{n+1}^{(j_b)}$
n shares					b secrets		

- It is worth noting that the block-maximal set B of linearly independent vectors modulo C' that $\mathcal{R}$ computes is maximal in the vector blocks involved, but not in the encoded blocks involved. Even though this is equivalent because the number of the encoded blocks involved is exactly b times the number of vector block involved, the definition of $R = R(B)$ differs.

- During the second communication round $\mathcal{A}$ learns b secrets for each vector in the reduced block-maximal set of linearly independent vectors modulo C' that is broadcast. However, each of these single secrets belongs to a single encoded block, and privacy is preserved for the same reasons as stated in Section 3.4.3.

- with the information contained in the index I computed by $\mathcal{S}$ after the second communication round, $\mathcal{R}$ after the third communication round can fix the codewords in all the j-th vector blocks with $j \in R$, recovering b message blocks for each vector block. In particular, he has at least $bn^{\gamma-1} - bt$ values y_j.

- during the third communication round, if $a \neq t$ and $|S| \neq |R|$, $\mathcal{S}$ must broadcast $b|R|$ messages instead of only $|R|$; however, since there are at least $bn^{\gamma-1} - bt > b|R|n$ values y_j on which $\mathcal{A}$ has no knowledge about, he can still define the vectors $\vec{z}^{(h)}$ (they will be $b|R|$ this time) and use them as keys for a one-time pad encryption.

We now analyze the information rate of this variant of the protocol.

- During the first communication round still $n^\gamma + n^{\gamma-1}$ codewords of length n are sent, for an amount of $n^{\gamma+1} + n^\gamma$ field elements.

- During the second communication round up to t codewords of length n and up to $2t$ indices are broadcast, for an amount of $tn^2 + 2tn$ field elements; again here there is no difference.

- During the third communication round we again have two possibilities.
  - if $a = t$ or $|S| = |R|$, then only a set of indices of size at most t is broadcast, for an amount of up to tn field elements;
  - if $a \neq t$ and $|S| \neq |R|$ then up to bt vectors of length n and up to t indices are broadcast, for an amount of $btn^2 + tn$ field elements.

Therefore in total $n^{\gamma+1} + n^\gamma + t(b+1)n^2 + 3tn$ field elements were sent in order to send a message of size $L = bn^\gamma$. Hence, the information rate becomes

$$\frac{n}{b} + \frac{1}{b} + \frac{1}{2n^{\gamma-3}} + \frac{3}{2bn^{\gamma-2}} = \frac{n+1}{b} + O\left(\frac{1}{n^{\gamma-3}}\right).$$

It is clear that as γ grows, this quantity asymptotically tends to $\frac{n+1}{b} = \frac{n+1}{n-2t}$, nearly reaching optimality in this case as well.

4 A Nearly Optimal 2-Round Protocol

In Section 3 we presented a 3-round protocol which asymptotically meets the lower bound for the communication overhead of $\frac{n}{n-2t}$. This lower bound for the case $n \geq 2t+1$ holds for any number of rounds which is larger than 2, hence it is worth asking whether we can meet the lower bound tightly in a 2-round setting as well.

Just as for the 3-round case, a 2-round protocol with a communication overhead of $\frac{n}{n-2t}$ (or asymptotically close) has never been presented. The best result currently known is due to Kurosawa and Suzuki [KS08]. In their paper they reach $\frac{cn}{n-2t}$ overhead, up to a multiplicative constant c. This constant c can be shown to be 25. In this section we introduce some new ideas to improve this constant, and present a 2-round protocol that as the message size increases reaches a (asymptotic) communication overhead $\frac{6n}{n-2t}$. That is, we improve the multiplicative constant from 25 to 6.

First we present a version of the protocol that holds for $n = 2t+1$. We open with an overview and then prove privacy, reliability and communication overhead. Next, in Section 4.4, we generalize this protocol for the $n = 2t+b$ case.

4.1 Graph Matching

In this section we explain in detail a technique that was presented by Narayanan et al. in [NPRS04] and then used in [ACH06] and [KS08].

Let C be a linear code of length n over $\mathbb{F}_q$ with minimum distance at least $t+1$. Assume that during the initial communication round a party (here we assume, without loss of generality, $\mathcal{R}$) sends n vectors $\vec{c}^{(k)} \in C$ for $k \in \{1, \ldots, n\}$ by sending the element $c_i^{(k)}$ through each channel i. Assume moreover that he also transmits through each channel i the vector $\vec{c}^{(i)}$.

Suppose that $\mathcal{A}$ corrupts the transmission. Hence, the other party (here we assume $\mathcal{S}$) received the vectors $\vec{d}^{(k)}$, and moreover a vector $\vec{v}^{(i)}$ from each channel i.

It is easy to see that for each $i \in \{1, \ldots, n\}$ the received vector $\vec{v}^{(i)}$ can differ from the vector $\vec{c}^{(i)}$ originally sent by $\mathcal{R}$ only if i is controlled by the adversary.

Definition 4.1. *A* conflict *is a vector $((i, \ell), u_{i,\ell})$ with the following properties:*

- $i \in \{1, \ldots, n\}$,
- $\ell \in \{1, \ldots, n\} \setminus \{i\}$,

- $u_{i,\ell} \in \mathbb{F}_q$,
- $u_{i,\ell} = v_\ell^{(i)}$ *and*
- $d_\ell^{(i)} \neq v_\ell^{(i)}$.

In other words, a conflict is a vector which contains information about a position in which $\vec{v}^{(i)}$ and $\vec{d}^{(i)}$ differ. With the data that he received, $\mathcal{S}$ is able to determine all such conflicts between the vectors $\vec{v}^{(i)}$ and the $\vec{d}^{(i)}$.

First, assume without loss of generality that for every $i \in \{1, \dots, n\}$ we have $\vec{v}^{(i)} \in C$. Indeed, the case $\vec{v}^{(i)} \notin C$ can happen only in the case of a corruption of $\mathcal{A}$. Since we are interested in worst-case adversaries, we therefore assume that whenever $\mathcal{A}$ controls a channel i he transmits a vector $\vec{v}^{(i)} \in C$.

Also assume, again without loss of generality, that $d(\vec{v}^{(i)}, \vec{d}^{(i)}) \leq t$. Indeed, if $d(\vec{v}^{(i)}, \vec{d}^{(i)}) > t$, then since $\mathcal{A}$ can corrupt at most t entries of $\vec{d}^{(i)}$ it is clear that $\vec{v}^{(i)} \neq \vec{c}^{(i)}$ and that channel i is controlled by $\mathcal{A}$. Since again we are interested in worst-case adversaries, we therefore assume that $\mathcal{A}$, for any channel i he controls, causes at most t conflicts with respect to $\vec{v}^{(i)}$.

Two other observations are important. The first is that there cannot be a conflict of type $((i,i), u_{i,i})$. Indeed, the existence of such a conflict would trivially imply that channel i is corrupted and we consider only worst-case adversaries. The second is that if $\vec{d}^{(i)} \neq \vec{c}^{(i)}$, then $\vec{d}^{(i)} \notin C$ and in particular $\vec{d}^{(i)} \neq \vec{v}^{(i)}$. This implies that at least one conflict will be generated in this case.

Suppose that $\mathcal{S}$ wants to send to $\mathcal{R}$ all the conflicts he computed from the received data. We need a method to limit the amount of data he has to transmit and this method is given by the *maximal matching*. Let $G = (V, E)$ be the undirected graph defined as follows:

- the vertex set is $V = \{1, \dots, n\}$;
- a pair (i, ℓ) belongs to the edge set E if and only if there is a conflict $((i, \ell), u_{i,\ell})$.

It is clear that if $(i, \ell) \in E$ then at least one of the channels i and ℓ is corrupted by the adversary.

Definition 4.2. *A* matching M *of a graph G is a set $M \subseteq E$ such that if $(i_1, \ell_1), (i_2, \ell_2) \in M$ then i_1, i_2, ℓ_1, ℓ_2 are pairwise distinct.*

We are interested in a *maximal* matching, i.e., a matching M for which there does not exist a matching M' with $M \subset M' \subseteq E$. Such a maximal matching can easily be computed in polynomial time (see [KS08]).

Each edge is given by a conflict. The following proposition states how a maximal matching gives an upper bound on the total number of conflicts that $\mathcal{S}$ computes. A similar result can be found in [KS08], but here directly we apply it to our purposes.

Proposition 4.3. *$|E| \leq 2|M|t$.*

Proof. Define:

$$V(M) = \{i \in V \mid \exists \ell \in \{1, \ldots, n\} \text{ s.t. } (i, \ell) \in M\}.$$

Then, since M is a maximal matching, for each $(i, \ell) \in E$ we have that either i or ℓ lies in $V(M)$. Moreover, $|V(M)| = 2|M|$, and for each $i \in V$ we assume without loss of generality that $|\{\ell \mid (i, \ell) \in E \text{ or } (\ell, i) \in E\}| \leq t$. Indeed:

- Let i be a channel which is not controlled by $\mathcal{A}$. Then the data that is sent through i is received correctly, and hence it generates no conflict between any other channel j which is not controlled by $\mathcal{A}$. Similarly, any channel j which is not controlled by $\mathcal{A}$ generates no conflict with i. Since the number of channels which are not controlled by $\mathcal{A}$ (including i) is $n - t$, we have that indeed $|\{\ell \mid (i, \ell) \in E \text{ or } (\ell, i) \in E\}| \leq t$.

- If for a channel i we have that $|\{\ell \mid (i, \ell) \in E \text{ or } (\ell, i) \in E\}| > t$, it follows from the argument above that i must be controlled by $\mathcal{A}$. It follows that in particular $\mathcal{S}$ can deduce this. Since we are interested in worst-case adversaries we therefore assume that for any channel i which is controlled by $\mathcal{A}$ we have that $|\{\ell \mid (i, \ell) \in E \text{ or } (\ell, i) \in E\}| \leq t$ as well.

Therefore:

$$|E| \leq \sum_{i \in V(M)} |\{\ell \mid (i, \ell) \in E \text{ or } (\ell, i) \in E\}| \leq 2|M|t.$$

□

Another very important property is that if M is a maximal matching, then at least $|M|$ channels were corrupted by the adversary. Indeed, as we saw before, for each $(i, \ell) \in M$ either i or ℓ (or both) is corrupted by $\mathcal{A}$.

4.2 Generalized Broadcasting

Before we present the protocol we need to introduce another technique that is used during its execution, which is *generalized broadcasting*. The purpose of this technique is to efficiently send information reliably in one communication round without requiring privacy.

Let $n = 2t + b$ for some $1 \leq b \leq t$. Suppose that the adversary controls t channels and that the receiver knows $w \leq t$ of them. Finally, let C be an $[n, k]$-linear code over $\mathbb{F}_q$ with minimum distance $d \geq w + 2(t - w) + 1$. Then *generalized broadcasting* based on C consists of encoding k elements of $\mathbb{F}_q$ into a codeword $\vec{x} \in C$ and then sending over each channel $i \in \{1, \ldots, n\}$ the value x_i.

We prove that generalized broadcasting can be used to send information reliably. Assume that $\mathcal{A}$ during the transmission corrupts some values of $\vec{x}$ in the $t - w$ positions which are not known by the receiver. Then the received vector has at most t errors, of which w are marked by the receiver as erasures. Since $d \geq w + 2(t - w) + 1$ the receiver can therefore correct the w erasures and in addition $t - w$ more errors. Hence, not only does the receiver learn the original vector, but he can also spot all the channels that were corrupted by $\mathcal{A}$ during transmission of the codeword $\vec{x}$.

It is worthwhile to note that for $b = 1$ this is the description of the broadcasting that is introduced in Section 2.2.1, where we choose the code $C = \{(a, \ldots, a) \in \mathbb{F}_q^n \mid a \in \mathbb{F}_q\}$. We note also that if $k \in \Omega(n)$ then this technique allows to send reliably a linear amount of information.

4.3 The Nearly Optimal 2-Round Protocol

4.3.1 Overview of the Protocol

In this section we present our nearly-optimal 2-round protocol. It is based on the protocol presented by Kurosawa and Suzuki in [KS08], but we use some different techniques aiming to minimize the amount of data which is sent during the protocol execution.

Let $\gamma \geq 3$, $n = 2t + 1$ and let q be such that $q^n \gg n^{\gamma-1}$. The latter is the only restriction we need to place on q and it is easy to see that n does not need to be very large to allow any $q \geq 2$.

Remark. By the isomorphism of vector spaces $\mathbb{F}_{q^n} \cong \mathbb{F}_q^n$ we may view each field element that is sent as a vector in $\mathbb{F}_q^n$. In fact, each element of $\mathbb{F}_{q^n}$ is transmitted via n elements of $\mathbb{F}_q$. This turns out to be very useful for the efficiency of the protocol.

In this protocol three Reed-Solomon codes are used, namely C, C_2 and C_3. They have the following properties:

- The secret sharing scheme associated with C is a t-threshold scheme. Indeed, each share associated with a codeword of C will be sent through a different channel and since $\mathcal{A}$ is a t-adversary we need t-privacy. The $(t+1)$-reconstruction is required in the information reconciliation part of the protocol.

- The ramp sharing scheme associated with C_3 encodes secrets of size $t+1$ and has t-privacy and n-reconstruction. C_3 is used for the privacy amplification part of the protocol.

- Unlike the codes C and C_3, the code C_2 is defined during protocol execution and its parameters depend on the strategy of the adversary. C_2 is used to reliably send information with generalized broadcasting (see Section 4.2).

The main idea in this protocol is to apply some additional techniques to the protocol of Kurosawa and Suzuki. As in the optimal 3-round protocol presented in Section 3 the message has size $L = \Omega(n^\gamma)$ over $\mathbb{F}_{q^n}$ and is divided into $n^{\gamma-1}$ message blocks. Furthermore, the property that $q^n \gg n^{\gamma-1}$ allows to decrease the communication overhead for the conflict sending part of the protocol.

4.3.2 The Protocol

Let C be an $[n{+}1, t{+}1]$-Reed-Solomon code over the field $\mathbb{F}_{q^n}$. By Example 2.37 the secret sharing scheme associated with C is a t-threshold scheme.

As we did for the 3-round protocol in Section 3.2, we also define an $[n, t+1]$-Reed-Solomon code C' by considering the first n positions of each codeword of C. As demonstrated earlier, the secret sharing scheme associated with C' is a t-threshold scheme as well.

Let $\gamma \in \mathbb{Z}$, $\gamma \geq 3$ and suppose that $\mathcal{S}$ wants to send to $\mathcal{R}$ a message of size $L = (t+1)n^{\gamma-1}$ over $\mathbb{F}_{q^n}$. In the first communication round, $\mathcal{R}$ acts as in Figure 4.9.

In other words, first $\mathcal{R}$ selects $n^\gamma{+}tn$ codewords of C uniformly at random. The reason for the "$+tn$" is that up to t vector blocks are discarded during the protocol execution. Then, he divides them into $n^\gamma + t$ vector blocks consisting of n vectors each.

Now, for every $\vec{c}^{(j,k)}$ let $f_{j,k} \in \mathbb{F}_{q^n}$ be the defining polynomial of the codeword. In general we have:

$$f_{j,k}(x) = a_0^{(j,k)} + a_1^{(j,k)}x + \cdots + a_t^{(j,k)}x^t;$$

$\mathcal{R}$ defines $\vec{a}^{(j,k)}$ to be the vector:

$$\vec{a}^{(j,k)} = (a_0^{(j,k)}, \ldots, a_t^{(j,k)}) \in \mathbb{F}_{q^n}^{t+1}.$$

$\mathcal{R}$ acts as follows.

- He selects $n^\gamma + tn$ codewords of C uniformly at random;
- He divides them into $n^{\gamma-1} + t$ blocks of n codewords each, labelling each block with an index j. From here on $\vec{c}^{(j,k)}$ denotes the k-th codeword in the j-th block.
- For each pair (j, k) he initializes the vectors $\vec{a}^{(j,k)} \in \mathbb{F}_{q^n}^{t+1}$ as the vector containing the coefficients of the defining polynomial of the codeword $\vec{c}^{(j,k)}$.
- For every $j \in \{1, \dots, n^2 + t\}$, $k \in \{1, \dots, n\}$ and $i \in \{1, \dots, n\}$ $\mathcal{R}$ sends through channel i the element $c_i^{(j,k)}$, keeping secret the element $c_{n+1}^{(j,k)}$. In addition, he sends through each channel i the vector $\vec{a}^{(j,i)}$.

Figure 4.9: First communication round.

It is clear that for any (j, k) the vector $\vec{a}^{(j,k)}$ uniquely defines the polynomial $f_{j,k}$, and hence uniquely defines the codeword $\vec{c}^{(j,k)}$. Note that the vector $\vec{a}^{(j,i)}$ that is sent through channel i uniquely defines a codeword which is equal to the vector $\vec{c}^{(j,k)}$ with $k = i$. From here on, we call $\vec{c}^{(j,k)\prime}$ the vector $(c_1^{(j,k)}, \dots, c_n^{(j,k)}) \in \mathbb{F}_{q^n}^n$. By definition we have that $\vec{c}^{(j,k)\prime} \in C'$.

The table below shows how encoding and sending are performed in this step for each vector block j. Also here we omit the index "j" in the notation for simplicity.

$a_0^{(1)}$ $a_1^{(1)}$ $\vdots$ $a_{t-1}^{(1)}$ $a_t^{(1)}$	$a_0^{(2)}$ $a_1^{(2)}$ $\vdots$ $a_{t-1}^{(2)}$ $a_t^{(2)}$	$\dots$ $\dots$ $\ddots$ $\dots$ $\dots$	$a_0^{(n-1)}$ $a_1^{(n-1)}$ $\vdots$ $a_{t-1}^{(n-1)}$ $a_t^{(n-1)}$	$a_0^{(n)}$ $a_1^{(n)}$ $\vdots$ $a_{t-1}^{(n)}$ $a_t^{(n)}$	
$c_1^{(1)}$	$c_2^{(1)}$	$\dots$	$c_{n-1}^{(1)}$	$c_n^{(1)}$	$c_{n+1}^{(1)}$
$c_1^{(2)}$	$c_2^{(2)}$	$\dots$	$c_{n-1}^{(2)}$	$c_n^{(2)}$	$c_{n+1}^{(2)}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
$c_1^{(n-1)}$	$c_2^{(n-1)}$	$\dots$	$c_{n-1}^{(n-1)}$	$c_n^{(n-1)}$	$c_{n+1}^{(n-1)}$
$c_1^{(n)}$	$c_2^{(n)}$	$\dots$	$c_{n-1}^{(n)}$	$c_n^{(n)}$	$c_{n+1}^{(n)}$
n shares and n vectors					1 secret

Now, suppose that $\mathcal{S}$ after the first communication round received for each block j the vectors $\vec{d}^{(j,k)} \in \mathbb{F}_{q^n}^n$ and $\vec{w}^{(j,i)} \in \mathbb{F}_{q^n}^n$, where $\mathcal{R}$ transmitted

$\vec{c}^{(j,k)'}$ and $\vec{a}^{(j,i)}$ respectively. In the second communication round $\mathcal{S}$ acts as in Figure 4.10.

$\mathcal{S}$ does the following.

- He computes a maximal set B of linearly independent vectors modulo C' for the whole error-vector space and the set $R = R(B)$.
- He broadcasts B and R.
- For each vector block $j \in J$ he acts as follows.
  - He computes the vectors $\vec{v}^{(j,i)} \in C'$ as the unique codewords defined by the vectors $\vec{a}^{(j,i)}$.
  - He computes all the conflicts $((i,\ell), u_{i,\ell})$ generated by the vectors $\vec{d}^{(j,i)}$ and $\vec{v}^{(j,i)}$, as defined in Section 4.1.
- He defines a graph G as in Section 4.1 from all the computed conflicts, and finds a maximal matching M.
- He defines $w = |M|$.
- He defines an $[n, w+1]$-Reed-Solomon code C_2 over $\mathbb{F}_{q^n}$.
- He defines the set $J = \{j_1, \ldots, j_{n^{\gamma-1}}\}$ with the first $n^{\gamma-1}$ indices j, according to the usual order on the natural numbers, of the set $\{1, \ldots, n^{\gamma} + t\} \setminus R$.
- For every $j \in J$ he encodes all the conflicts into m codewords of C_2, say $\vec{x}^{(j,1)}, \ldots, \vec{x}^{(j,m)}$.
- For each $j \in J$ and each channel i he sends to $\mathcal{R}$ the elements $x_i^{(j,1)}, \ldots, x_i^{(j,m)}$ through channel i.

Figure 4.10: Second communication round.

$\mathcal{S}$ during this step computes a maximal set B of linearly independent vectors modulo C' for the whole error-vector space spanned by all the errors carried by the vectors $\vec{d}^{(j,k)}$ and then broadcasts it. As we saw in Section 3.1, the set B allows $\mathcal{R}$ to recover all the channels that were corrupted by $\mathcal{A}$ during the first communication round.

In addition $\mathcal{S}$ computes all the conflicts generated by the vectors $\vec{v}^{(j,i)}$ and $\vec{d}^{(j,k)}$ with $k = i$. He then encodes them into codewords of a code C_2 whose parameters depend on the size of the maximal matching of the conflict graph. In Section 4.3.4 we prove that $\mathcal{S}$ is able to send data reliably with C_2. Note that from this point on the blocks j with $j \in R$ are not considered anymore in the protocol execution.

Now, $\mathcal{S}$ applies privacy amplification and sends the message to $\mathcal{R}$, as described in Figure 4.11.

$\mathcal{S}$ does the following.

- For each $j \in J$ he defines a vector $\vec{y}^{(j)} \in \mathbb{F}_{q^n}^n$, where for each $i \in \{1, \ldots, n\}$ we have that $y_i^{(j)}$ is the secret corresponding to the set of shares expressed by $\vec{v}^{(j,i)}$.
- He defines an $[n+t, n]$-Reed-Solomon code C_3 over $\mathbb{F}_{q^n}$.
- For each $j \in J$ he considers $\vec{y}^{(j)}$ as the first n elements of a codeword $(\vec{y}^{(j)}, \vec{z}^{(j)}) \in C_3$, where $\vec{z}^{(j)} \in \mathbb{F}_{q^n}^{t+1}$.
- He joins all the $\vec{z}^{(j)}$ together in order to obtain a private key $K \in \mathbb{F}_{q^n}^L$.
- $\mathcal{S}$ encrypts the message with K and broadcasts it to $\mathcal{R}$.

Figure 4.11: Privacy amplification and sending of the message

For each block j $\mathcal{S}$ considers the secret elements carried by the vectors $\vec{v}^{(j,k)}$, obtaining a vector of n field elements. He then extracts $t+1$ field elements from them using C_3. In Section 4.3.3 we prove that these elements are perfectly private. In total $\mathcal{S}$ has $L = (t+1)n^{\gamma-1}$ perfectly private field elements, which are used as a key for a one-time pad encryption. This concludes the second communication round.

Finally $\mathcal{R}$ reconciles his information with $\mathcal{S}$ and computes the message with the procedure listed in Figure 4.12.

$\mathcal{R}$ does the following.

- He uses all the conflicts received to recover all the codewords $\vec{v}^{(j,i)}$.
- He applies privacy amplification to the secrets carried by these codewords as $\mathcal{S}$ did in Figure 4.11, and therefore obtains K.
- He subtracts K from the encrypted message.

Figure 4.12: Message reconstruction for $\mathcal{R}$

First, $\mathcal{R}$ uses the conflicts received during the second communication round to recover all the vectors $\vec{v}^{(j,i)}$. In Section 4.3.4 we explain in detail how this works. He then applies privacy amplification to these vectors and obtains K. Finally he uses K to decrypt the message.

This ends the description of the protocol. In the next two sections we discuss respectively the privacy and the reliability of the protocol.

4.3.3 Privacy of the Protocol

In this section we analyze the privacy of our 2-round protocol. We recall that a protocol is perfectly private when the secret message is independent of the view of the adversary $\mathcal{A}$, or in other words if $\mathcal{A}$ at the end of the protocol execution has not even a partial knowledge about the message.

First, we study the first communication round. Suppose that channel i is not controlled by $\mathcal{A}$. Then, after the first communication round $\mathcal{A}$ knows t positions for each codeword $\vec{c}^{(j,i)}$. Since C is a t-threshold scheme, it follows that $\mathcal{A}$ does not have enough information to recover the secret $c_{n+1}^{(j,i)}$.

Suppose, instead, that channel i is controlled by $\mathcal{A}$. Then in addition to t entries of the codeword $\vec{c}^{(j,i)}$, $\mathcal{A}$ learns also the vector $\vec{a}^{(j,i)}$. Hence, he can recover the full codeword $\vec{c}^{(j,i)}$ and therefore the secret $c_{n+1}^{(j,i)}$.

During the sending of B $\mathcal{A}$ learns some additional information about the blocks j with $j \in R$; this additional information, though, does not change his view since $\mathcal{S}$ and $\mathcal{R}$ will not use the blocks j with $j \in R$ any further in the protocol.

It is straightforward to see that encoding all the conflicts into the codewords $\vec{x}^{(j,1)}, \ldots, \vec{x}^{(j,m)} \in C_2$ and then sending these vectors is equivalent to sending the conflicts using a generalized broadcasting based on C_2. We therefore need to prove that the sending of all the conflicts does not change the view of $\mathcal{A}$.

The vectors encoded into codewords of C_2 consist of the conflicts that $\mathcal{S}$ computed in every block j with $j \notin R$. Suppose that the conflict $((i, \ell), u_{i,\ell})$ was sent. Then:

- if $\mathcal{A}$ controls channel i, then he already knows the element $u_{i,\ell}$;
- if $\mathcal{A}$ does not control channel i, then he controls channel ℓ and the value $d_\ell^{(j,i)}$ was corrupted by him; but in particular this means that he already knew the value $u_{i,\ell} = v_\ell^{(j,i)} = c_\ell^{(j,i)}$.

Hence, the conflicts can even be computed independently by $\mathcal{A}$ before they are transmitted by $\mathcal{S}$.

Finally, we must prove that the extracted key K is perfectly private. The ramp sharing scheme associated with C_3 with secrets of size $t+1$ has

t-privacy and n-reconstruction. Now, consider the vector $\vec{y}^{(j)}$. By the properties of C_3, these are enough to extract the latter $t+1$ entries, which are used for privacy amplification. Indeed, from Example 2.41 with $\ell = t+1$ we know that C_3 has t-privacy for secrets of size $(t+1)$ and $\mathcal{A}$ learns at most t values.

4.3.4 Reliability of the Protocol

In this section we prove that our 2-round protocol has perfect reliability, i.e., that at the end of the protocol execution $\mathcal{R}$ is able to output the message correctly. As for privacy, we analyze the reliability for each round.

Suppose that $\mathcal{S}$ after the first communication round received the vectors $\vec{d}^{(j,k)} \in \mathbb{F}_{q^n}^n$ and the vector $\vec{w}^{(j,i)} \in \mathbb{F}_{q^n}^{t+1}$ from each channel i. Since every $\vec{c}^{(j,k)'} \in C'$, if $\mathcal{A}$ corrupted some entries of this vector then $d(\vec{c}^{(j,k)'}, \vec{d}^{(j,k)}) \leq t < d(C')$, and $\vec{d}^{(j,k)} \notin C'$.

If $\mathcal{A}$ controls channel i however, he can corrupt the vector $\vec{a}^{(j,i)}$ and forward to $\mathcal{S}$ a vector $\vec{w}^{(j,i)} \neq \vec{a}^{(j,i)}$. In this case $\mathcal{A}$ can modify any number of entries in this vector. On the other hand, when channel i is not controlled by $\mathcal{A}$, we have $\vec{w}^{(j,i)} = \vec{a}^{(j,i)}$.

Now, during the second communication round the maximal set B of linearly independent vectors modulo C' is sent. We know from Section 3.1 that B allows $\mathcal{R}$ to determine all the channels that were corrupted by $\mathcal{A}$ during the first round and in particular on the blocks $j \notin R$. Since $|R| \leq t$ there are more than $n^{\gamma-1}$ such blocks by definition. This means that J is well defined. The blocks j with $j \in R$ are discarded.

It remains to prove that $\mathcal{R}$ is also able to compute the secret key K. Since privacy amplification runs deterministically on the elements to which it is applied, it suffices to prove that $\mathcal{R}$ can recover all the n^γ secrets used by $\mathcal{S}$. For this, he needs to know all the vectors $\vec{v}^{(j,i)}$.

We can assume that $\mathcal{R}$ received B and R correctly. We can also assume that for each $j \in J$ he received the vectors $\vec{x}^{(j,1)}, \dots, \vec{x}^{(j,m)}$ correctly. Indeed, suppose that $\mathcal{A}$ corrupted some of their entries. From B $\mathcal{R}$ is able to determine all the channels that were corrupted by $\mathcal{A}$ during the first round on the blocks j with $j \notin R$ (and, in particular, for $j \in J$). As we saw in Section 4.1, these channels are in total at least w. On the other hand, there are at most $t-w$ channels that $\mathcal{A}$ controls which are still unknown to $\mathcal{R}$. Now, $\mathcal{R}$ can erase the corrupted channels that he knows from the communication. Since

$$d(C_2) = n - (w+1) + 1 = n - w = (2t+1) - w = 2(t-w) + w + 1,$$

he can correct w erasures and $t-w$ random errors from each codeword, and in this way he recovers the vectors $\vec{x}^{(j,1)}, \dots, \vec{x}^{(j,m)}$ with perfect correctness. As we remarked in Section 4.3.3 this is equivalent to use a generalized broadcasting based on C_2.

Suppose that channel i was not corrupted by the adversary. Then $\mathcal{S}$ and $\mathcal{R}$ share the vector $\vec{c}^{(j,i)}$. $\mathcal{S}$ obtains it from the vector $\vec{a}^{(j,i)}$.

Suppose, instead, that channel i was corrupted by the adversary. Then $\mathcal{R}$ considers a set of $t+1$ channels which were not corrupted by the adversary during the first phase. Let ℓ be one of these indices. There are two possibilities:

- either there is not a conflict $((i,\ell), u_{i,\ell})$, and hence $\vec{v}^{(j,i)} = \vec{c}^{(j,i)}$;
- or there is a conflict $((i,\ell), u_{i,\ell})$, and hence $\vec{v}^{(j,i)} = u_{i,\ell}$.

Therefore, $\mathcal{R}$ knows $t+1$ entries of the vector $\vec{v}^{(j,i)}$ and this is enough for him to recover the entire vector. This implies that $\mathcal{R}$ can compute all the vectors $\vec{v}^{(j,i)}$ and this concludes our analysis on the reliability of the protocol.

4.3.5 Communication Overhead of the Protocol

Now we analyze the communication overhead of the Protocol. We recall that $\mathcal{S}$ effectively sends a message in $\mathbb{F}_{q^n}^L$ with $L = (t+1)n^{\gamma-1}$.

- During the first communication round $\mathcal{R}$ sends to $\mathcal{S}$ $n^\gamma + tn$ codewords of C', for a total of $n^{\gamma+1} + tn^2$ field elements. Moreover, $n^\gamma + tn$ vectors $\vec{a}^{(j,i)}$ are sent, for a total of $n^{\gamma+1} + (t+1)n^\gamma + (3t+2)tn$ field elements.
- B has size at most t, hence up to t codewords are broadcast, for a total of tn^2 field elements. The set R contains at most t indices and each index requires at most $\log_q(n^{\gamma-1}+t)$ elements of $\mathbb{F}_q$ to be sent. This can be considered a negligible cost.
- As we saw in Section 4.1 for each $j \in J$ at most $2wt$ conflicts are sent. Each conflict $((i,\ell), u_{i,\ell})$ requires $2\log_q(n) + n$ elements over $\mathbb{F}_q$, which for n large enough can be simplified to n. Hence, $2wt$ elements of $\mathbb{F}_{q^n}$ are sent with the generalized broadcasting, which then has a cost of $2wt \cdot \frac{n}{w+1} \leq n^2$. This holds for each $j \in J$, so in total $\mathcal{S}$ sends about $n^{\gamma+1}$ field elements.
- Finally, $\mathcal{S}$ broadcasts its encrypted message of size L, which requires $(t+1)n^\gamma$ field elements.

Hence in total $2n^{\gamma+1} + 2(t+1)n^\gamma + (5t+3)tn$ field elements were sent (up to negligible differences), for sending a message of size $L = (t+1)n^{\gamma-1}$ over $\mathbb{F}_{q^n}$.

Therefore, as γ grows asymptotically the information rate of the protocol is

$$\frac{2n^{\gamma+1} + 2(t+1)n^{\gamma} + (5t+3)tn}{(t+1)n^{\gamma-1}} \leq 6n + \frac{6}{n^{\gamma-3}} = 6n + O\left(\frac{1}{n^{\gamma-3}}\right),$$

as promised.

Remark. We can make two small improvements to the protocol, which were not included in the protocol description because they do not influence the performance much, while on the other hand would have made the analysis on the overhead more complicated.

- Since we assumed that $q^n \gg n^{\gamma-1}$, it is not too restrictive to assume that $q^n \geq n^{\gamma} + (t+1)n^{\gamma-1} + n^2 - t^2 - n - t$. Hence, we can decide not to discard any vector block. Indeed, in total $\mathcal{S}$ and $\mathcal{R}$ share $n^{\gamma} + tn$ secrets, of which $tn^{\gamma-1} + t^2 + t$ are known by $\mathcal{A}$. Then they can apply privacy amplification to the whole vector using a Reed-Solomon code with parameters $[n^{\gamma} + (t+1)n^{\gamma-1} + n^2 - t^2 - n - t, n^{\gamma} + tn]$; this would allow them to generate a vector of size $(t+1)n^{\gamma-1} + (t+1)n - t^2 - n - t$ which is perfectly private.

- During the sending of the message, $\mathcal{S}$ can actually use generalized broadcasting instead of the normal broadcasting. The efficiency of this modification however depends on w, and there are optimal adversarial strategies which allow an adversary to minimize the improvement of the overhead.

4.4 Case $n = 2t + b$

We can generalize our protocol for $n = 2t + 1$ to the more relaxed case $n = 2t + b$, where $b \in \mathbb{Z}$. Note that b can be linear in n, i.e., $b = \varepsilon t$ for some $0 < \varepsilon \leq 1$.

As before, we pick a q for which $q^n \gg n^{\gamma-1}$. We define the code C to be an $[n+b, t+b]$-Reed-Solomon code over $\mathbb{F}_{q^n}$. As we saw in Example 2.41 the ramp sharing scheme associated with C with secrets of size b is a $(t, t+b)$-ramp scheme.

Pick, as for the case $n = 2t + 1$, the $[n, t+b]$-Reed-Solomon code C' consisting of the vectors formed by first n positions of the codewords in C. The secret sharing scheme associated with C' has also t-privacy and $(t+b)$-reconstruction.

Suppose that $\mathcal{S}$ wants to send to $\mathcal{R}$ a message of size $L = b(t+b)n^{\gamma-1}$ over $\mathbb{F}_{q^n}$. The protocol runs precisely as for the case $n = 2t + 1$, with the following differences.

- At the beginning of the first communication round $\mathcal{R}$ selects $n^\gamma + tn$ codewords of C as before. This time, however, each codeword carries b secrets instead of 1.
- The vectors $\vec{a}^{(j,i)}$ belong to $\mathbb{F}_{q^n}^{t+b}$, since the defining polynomials of the codewords of C' have degree at most $t+b$.
- The Reed-Solomon code C_2 that is used for generalized broadcasting during the second communication round has parameters $[n, w+b]$. Indeed, with such parameters the minimum distance of C_2 is:

  $$d(C_2) = n - (w+b) + 1 = 2t + b - w - b + 1 = 2(t-w) + w + 1.$$

  Therefore, C_2 can handle w erasures and up to $t-w$ random errors.
- Note that if $b = \Omega(n)$ then $\mathcal{S}$ is able to send a linear amount of data with C_2 even if $w = 0$, i.e., if no conflicts are found.
- The vectors $\vec{v}^{(j,i)}$ give a total of bn^γ secrets, of which $btn^{\gamma-1}$ are known by $\mathcal{A}$.
- The Reed-Solomon code C_3 used for privacy amplification has parameters $[n+t+b, n]$. Indeed, the privacy threshold of the ramp sharing scheme associated with this code with secrets of size $t+b$ is $t = n-(t+b)$, hence $\mathcal{A}$ has no partial knowledge about the secrets.
- After privacy amplification $\mathcal{S}$ and $\mathcal{R}$ share a common secret key K of size $b(t+b)n^{\gamma-1}$.
- $\mathcal{S}$ sends the message using generalized broadcasting. This is now essential to keep communication low.

Observe how this version of the protocol is exactly equal to the protocol for $n = 2t+1$ when we set $b = 1$.

Now we analyze the communication overhead of this protocol.

- During the first communication round $n^\gamma + tn$ codewords of C' and $n^\gamma + tn$ vectors $\vec{a}^{(j,i)} \in \mathbb{F}_{q^n}^{t+b}$, for an amount of $n^{\gamma+1} + (t+b)n^\gamma + tn^2 + (t+b)tn$ field elements.
- The broadcasting of the maximal set of linearly independent vectors modulo C' has the cost of the broadcasting of t codewords of length n, which is tn^2. The communication costs of R and S are negligible.

- As for the case $n = 2t + 1$, $\mathcal{S}$ for each $j \in J$ must send at most $2wt$ conflicts. Hence, the communication cost for this part, thanks to generalized broadcasting, is of $2wtn^{\gamma-1} \cdot \frac{n}{w+b} \leq n^{\gamma+1}$.
- Finally, $\mathcal{S}$ broadcasts its encrypted message of size $L = b(t + b)n^{\gamma-1}$ using generalized broadcasting, which has a cost of $b(t+b)n^{\gamma-1} \cdot \frac{n}{w+b} \leq (t + b)n^{\gamma}$.

Everything was sent in order to send a message of size $b(t+b)n^{\gamma-1}$. Therefore, applying to the denominator the inequality $t + b \geq \frac{n}{2}$ where needed, the communication overhead of this protocol is:

$$\frac{2n^{\gamma+1} + 2(t + b)n^{\gamma} + 2tn^2 + (t + b)tn}{b(t + b)n^{\gamma-1}} \leq \frac{4n}{b} + \frac{2n}{b} + \frac{2}{bn^{\gamma-3}} + \frac{1}{bn^{\gamma-3}} =$$
$$= \frac{6n}{b} + \frac{3}{bn^{\gamma-3}} = 6\frac{n}{b} + O\left(\frac{1}{bn^{\gamma-3}}\right),$$

as expected.

References

[ACH06] S. Agarwal, R. Cramer, and R. de Haan, *Asyptotically Optimal Two-Round Perfectly Secure Message Transmission*, Advances in Cryptology—CRYPTO '06, Lecture Notes in Computer Science, vol. 4117, Springer-Verlag, Berlin, 2006, pp. 389–401.

[DDWY93] D. Dolev, C. Dwork, O. Waarts, and M. Yung, *Perfectly Secure Message Transmission*, Journal of the ACM **40** (1993), no. 1, 17–47.

[FFGHV07] M. Fitzi, M. Franklin, J. Garay, and S. Harsha Vardhan, *Towards Optimal and Efficient Perfectly Secure Message Transmission*, Theory of Cryptography Conference: TCC '07, Lecture Notes in Computer Science, vol. 4392, Springer, Berlin, 2007, pp. 311–322.

[Haa09] R. de Haan, *Algebraic Techniques for Low Communication Secure Protocols*, Ph.D. thesis, 2009, URL: `https://openaccess.leidenuniv.nl/handle/1887/13603`.

[KS08] K. Kurosawa and K. Suzuki, *Truly Efficient 2-Round Perfectly Secure Message Transmission Scheme*, Advances in Cryptology—EUROCRYPT '08, Lecture Notes in Computer Science, vol. 4965, Springer-Verlag, Berlin, 2008, pp. 324–340.

[NPRS04] A. Narayanan, C. Pandu Rangan, and K. Srinathan, *Optimal Perfectly Secure Message Transmission*, Advances in Cryptology—CRYPTO '04, Lecture Notes in Computer Science, vol. 3152, Springer-Verlag, Berlin, 2004, pp. 545–561.

[Sha48] C. E. Shannon, *A Mathematical Theory of Communication*, Bell System Technical Journal **27** (1948), 379–423, 623–656.

[Sha79] A. Shamir, *How to Share a Secret*, Communications of the ACM **22** (1979), no. 11, 612–613.

A Appendix: A 2-Round Protocol Using Generic Codes

Until now, all the linear codes that have been used for defining protocols were Reed-Solomon codes. The reason for this is that Reed-Solomon codes have optimal parameters, they can be defined for every length n (given a large enough field) and are equipped with fast decoding algorithms.

One may ask, however, if protocols for PSMT can be defined using other kinds of codes, for example generic codes. What we show in this appendix is that the answer is yes, but we lose both communication and computational efficiency. While the fact that Reed-Solomon codes are MDS-codes allowed previous protocols to work with t-threshold schemes, the secret sharing schemes associated with generic codes are not in general threshold schemes. Hence, the approach given in this appendix is not possible for the case $n = 2t + 1$. Therefore, we must assume that $n = (2 + \varepsilon)t$ where, as usual, n is the number of channels, t is the number of channels controlled by the adversary and $0 < \varepsilon \leq 1$ with $\varepsilon t \in \mathbb{Z}$.

Here we give an overview of all the issues that come up when we weaken our assumptions on the field size and on the structures of the codes that we use, and present some ideas to overcome them. We present a 2-round protocol based on the protocol we gave in Section 4.4. Even though it is structured in the same way, some techniques are adapted to the use of generic codes. For this reason this protocol is only sketched Section A.2, as we mainly focus on the different technical solutions.

A.1 Uniformity Property for Codes

In this section we analyze an important property for secret sharing schemes associated with a linear code. If a code C has length n, then it can have s-*uniformity* for some $s \leq n$.

Definition A.1. *Let $s \in \mathbb{Z}$, $s \leq n$ and $C \in \mathbb{F}_q^n$ be a linear code of length n. Then the secret sharing scheme associated with C has* s-uniformity *if for every subset $\{i_1, \ldots, i_s\} \subseteq \{1, \ldots, n\}$ and for every choice of $a_1, \ldots, a_s \in \mathbb{F}_q$ there exists a codeword $\vec{c} \in C$ s.t. for every $j \in \{1, \ldots, s\}$ we have $c_{i_j} = a_j$.*

s-uniformity is very important, for example, when in the privacy amplification step of a perfectly secure message transmission protocol we need to see a random vector as a part of a codeword of a bigger code which allows us to extract the secret encryption key.

The following theorem proves a very simple sufficiency condition for a linear code to have s-uniformity.

Theorem A.2. *Let C be a linear code over $\mathbb{F}_q$ with dual C^* and suppose that $d(C^*) \geq s+1$. Then C has s-uniformity.*

Proof. Let $G \in M_{k\times n}(\mathbb{F}_q)$ be a generator matrix for C. Then G^T is a check matrix for C^*. $d(C^*) \geq s+1$, so by Lemma 2.7 every set of s rows of G^T (i.e., columns of G) is linearly independent. This implies that for every s columns $G_{i_1}, \ldots, G_{i_s}$ of G the map:

$$\begin{aligned} \varphi \colon \mathbb{F}_q^k &\to \mathbb{F}_q^s \\ \vec{v} &\mapsto \vec{v} \cdot (G_{i_1}, \ldots, G_{i_s}) \end{aligned}$$

is surjective. Hence, for every set of values $a_1, \ldots, a_s \in \mathbb{F}_q$ in the positions $i_1, \ldots, i_s$, one finds a $\vec{v}$ that is mapped there by φ. Therefore, the codeword $\vec{v} \cdot G$ contains $a_1, \ldots, a_s$ on the positions $i_1, \ldots, i_s$. This proves s-uniformity. □

In this appendix n-uniformity is required as a property of the code used for privacy amplification, as we see below in Section A.3.3. Theorem A.2 is used in Section A.5 for the discussion on the existence of the code that is used.

A.2 Sketch of the Protocol

In this section we introduce a sketch of the 2-round protocol that is based on the use of generic codes. We also give an overview of the issues that come up when the lack of structure that characterizes generic codes does not allow the use of the standard techniques. As stated in the introduction of this appendix, this protocol is mainly based on the protocol given in Section 4.4.

Let $n = (2+\varepsilon)t$ and $\mathbb{F}_q$ be a finite field whose size does not depend on n. Suppose furthermore that $\mathcal{S}$ wants to send to $\mathcal{R}$ a message M of size $\Omega(n^4)$.

During the first communication round $n^3 + tn$ codewords are randomly sampled by $\mathcal{R}$ and then divided into $n^2 + t$ blocks consisting of n vectors. $\mathcal{R}$ then sends these vectors component-wise through the n channels. In addition, for each of these codewords he defines a vector which uniquely defines it and sends it through a corresponding channel. Therefore this is in concept the same first communication round that is performed in our 2-round protocol using Reed-Solomon codes. However, the use of Reed-Solomon codes allows us to encode an optimal amount of secret values, namely εt, while keeping the properties of t-privacy and $(n-t)$-reconstruction. When using generic codes this optimality is not certain anymore, so we use a construction that allows for these properties. This construction is explained in Section A.3.1.

The second communication round now proceeds as follows. First $\mathcal{S}$, from all the vectors that he received during the first round, computes a maximal set of linearly independent vectors modulo the code used for the error-vector space and broadcasts it to $\mathcal{R}$. In Section 3.1 we show that this part can be executed efficiently with generic codes as well. He then defines the set of all conflicts generated by the received vectors. These conflicts however cannot be defined as in Section 4.1 because their size depends on n. Hence, a different way for defining the conflicts is used. This approach is introduced in Section A.4.

Furthermore, all these conflicts are sent with generalized broadcasting in order to keep the overhead constant. While with Reed-Solomon codes we can encode exactly an amount of $n-d+1$ field elements for each codeword (where d is the minimum distance of the code that is used), this is not trivial for generic codes. In fact, such an optimal amount is not reached in general. Hence, in Section A.3.2 we briefly explain how to perform generalized broadcasting with a suitable generic code.

Now, $\mathcal{S}$ applies privacy amplification and sends the message across. First, he extracts the secrets that are encoded by all the received codewords and he arranges them into vectors in $\mathbb{F}_q^n$. Privacy amplification now proceeds differently from that introduced in the 2-round protocol using Reed-Solomon codes. Indeed, the optimal parameters of Reed-Solomon codes allowed to meet two important requirements. The first is that each of the secret vectors could be seen as part of a codeword of a suitable code, so that privacy amplification could be applied. The second is that a key of maximal size $(1+\varepsilon)t$ could be extracted while keeping the properties of t-privacy and n-reconstruction. These two requirements are not met in general by generic codes. Therefore, in Section A.3.3 we analyze how privacy amplification is performed in this step.

$\mathcal{S}$ then sends the encrypted message across. $\mathcal{R}$ performs information reconciliation and mimics the privacy amplification performed by $\mathcal{S}$. After this he is able to output the original message. This concludes the protocol.

A.3 Perfectly Secure Message Transmission Over Finite Fields

In this section we analyze the issues that come up when we define a 2-round protocol using generic codes that were pointed out in Section A.2. In this setting we let $n = 2t + b = (2+\varepsilon)t$ with $0 < \varepsilon \leq 1$ and $b = \varepsilon t \in \mathbb{Z}_{\geq 1}$ and we let $\mathbb{F}_q$ be the fixed finite field that is used. In the end it turns out that we need generic codes with specific parameters that are given throughout the section. The existence of such codes is discussed in Section A.5.

A.3.1 Establishment of Correlations

In the protocol given in Section 4.4 an $[n+b, t+b]$-Reed-Solomon code is used for encoding b secrets and sending n shares. The ramp scheme associated with this code has t-privacy and $n-t = t+b = (1+\varepsilon)t$-reconstruction. When we use generic codes however we have the following considerations.

- We still require t-privacy and $(1+\varepsilon)t$-reconstruction for the code that is used during protocol execution. Indeed, the need for t-privacy follows from the fact that each component of any of its codewords is sent through a different channel. Moreover, this scheme requires $(n-t)$-reconstruction for the information reconciliation part of the protocol.

- In general a generic code does not meet the Singleton-bound. We cannot therefore in general encode the optimal amount of εt elements in each codeword. Hence, we relax the hypothesis on the size of the secret and we require that δt secrets instead of εt are encoded, for some $0 < \delta \leq \varepsilon$.

- When we consider the first n positions of a Reed-Solomon code we obtain another Reed-Solomon code. This in the protocol presented in Section 4.4 allow us to encode $b = \varepsilon t$ field elements as the last b positions of a codeword, while the first n positions are treated as shares and still belong to a secret sharing scheme with suitable properties. However, even though when we pick the first n positions of a linear code we indeed obtain a linear code, in general no assumption can safely be made on the parameters of such a shortened code. The construction for ramp schemes given by Theorem 2.42 is therefore used to get a proper encoding.

Hence, for this part of the protocol we use a linear code $\hat{C_1}$ with the following two properties, using the construction given in Theorem 2.42.

- There exists a subcode $C_1 \subset \hat{C_1}$ such that $\dim(\hat{C_1}) - \dim(C_1) = \delta t$ with δ as described above.

- The resulting ramp sharing scheme associated with the code $\hat{C_1}$ with the subcode C_1 has t-privacy and $(1+\varepsilon)t$-reconstruction.

Furthermore, in the protocol given in Section 4.4 for each codeword a vector containing the coefficients of its defining polynomial is sent. In this setting such "defining polynomials" do not exist, because codewords of generic codes in general do not have that structure. We therefore define for each of these codewords a vector of size $(1+\varepsilon)t$ over $\mathbb{F}_q$ which does not require any particular structure, but still uniquely defines it. This may be, for example, the vector containing the first $(1+\varepsilon)t$ elements of the codeword.

A.3.2 Generalized Broadcasting

In the protocol given in Section 4.4 we perform generalized broadcasting in order to send reliably information about the conflicts while keeping a constant overhead. This requires a Reed-Solomon code whose parameters depend on the strategy of the adversary. In the protocol based on generic codes a generalized broadcasting is performed as well.

If we however let the parameters of the generic code that is used depend on the strategy of the adversary then the analysis on the existence of such a code becomes too difficult. Hence, we define it a priori, independently of the adversary strategy. This implies that we should set the minimum distance to be at least $2t+1$, so that it can correct up to t errors for each codeword.

Since we are in the setting where $n=(2+\varepsilon)t$, by the Singleton-bound we have that the dimension of such a code can be at most εt. Our parameters are not tight in general, so as in Section A.3.1 we relax our hypothesis on the parameters and we require that the dimension is αt for some $0<\alpha\leq\varepsilon$.

Hence, for this protocol using generic codes we define a linear code C_2 with the following properties:

- $d(C_2)\geq 2t+1$, and
- $\dim(C_2)=k_2=\alpha t$ with α as described above.

Remark. Currently no algorithm for efficient decoding with generic codes is known. Hence, the part of the protocol which involves generalized broadcasting is computationally inefficient.

A.3.3 Privacy Amplification

The protocol given in Section 4.4 uses for privacy amplification an $[n+b+t, n]$-Reed-Solomon code. As for the establishment of correlations (Section A.3.1) and for generalized broadcasting (Section A.3.2), the fact that parameters for generic codes are not tight forces us to make further assumptions. We have the following considerations:

- First of all we need to be able to apply privacy amplification to any possible random vector in $\mathbb{F}_q^n$. For this we require for the linear code that we use in this step to have the following property: every possible vector in $\mathbb{F}_q^n$ consists in the first n positions of a codeword of this code.

This is closely related to n-uniformity, so by Theorem A.2 we require the minimum distance of the code to be at least $n + 1$.

- Since the parameters are not tight we cannot in general set the reconstruction threshold to be n. Hence, we relax the hypothesis on the reconstruction threshold and require for it to be strictly larger than n, say $n + \mu t$ for some $0 < \mu \leq \varepsilon$.

- If we set $n + \mu t$ as reconstruction threshold then the vector in $\mathbb{F}_q^n$ to which we want to apply privacy amplification does not contain enough information for reconstruction. A vector in $\mathbb{F}_q^{\mu t}$ should hence be defined and appended to the original vector in order to allow reconstruction. $\mathcal{S}$ and $\mathcal{R}$ must share this vector in order to both perform privacy amplification, so in the protocol description we require that $\mathcal{S}$ defines it and then broadcasts it to $\mathcal{R}$ during the second communication round. The adversary hence knows this vector. For this reason we also require $(1 + \mu)t$-privacy for this step.

- When we use a Reed-Solomon code for privacy amplification we can encode a secret of maximal length $(1 + \varepsilon)t$. Again, though, our parameters are not tight. On the one hand, we saw above that we need $(1 + \mu)t$-privacy. On the other hand we require n-uniformity. This implies that given the $(1 + \mu)t$ values known by the adversary, at most $n - (1 + \mu)t = (1 + \varepsilon - \mu)t$ more values are uncorrelated to them. Hence, this is the maximal size of a secret that can be encoded. As for code $\hat{C}_1$ in Section A.3.1 we therefore relax the hypothesis on the size of the secret and we assume that we can encode a secret of size λt for some $0 < \lambda \leq 1 + \varepsilon - \mu$.

- Similar to the Reed-Solomon code of Section 4.4, which has length $n + t + b$, we require the length of our generic code to be $n + \mu t + \lambda t = (2 + \varepsilon + \mu + \lambda)t$.

- As for the code $\hat{C}_1$, in order to get a proper encoding we use the construction given in Theorem 2.42 to define our ramp sharing scheme.

In conclusion, for the privacy amplification part of our protocol we define a linear code $\hat{C}_3$ with the following properties:

- Its length is $(2 + \varepsilon + \mu + \lambda)t$.

- There exists a subcode $C_3 \subset \hat{C}_3$ such that $\dim(\hat{C}_3) - \dim(C_3) = \lambda t$ with λ as described above.

- The resulting ramp sharing scheme associated with the code $\hat{C}_3$ with the subcode C_3 has $(1+\mu)t$-privacy and $(2+\varepsilon+\mu)t$-reconstruction with μ as described above.

A.4 Conflicts

During the second communication round of the 2-round protocol a graph is computed from all the conflicts generated by the received vectors. In this new setting these conflicts cannot be transmitted in the same manner. Indeed, a vector $((i,\ell),u_{i,\ell})$ has size $\Omega(\log(n))$ over $\mathbb{F}_q$, and therefore the whole set of conflicts cannot be sent without having an overhead of $\Omega(\log(n))$. Hence each set of conflicts contained in a codeword is encoded into a vector of different form.

Assume that during the first communication round a set of n vectors $\vec{c}^{(1)},\dots,\vec{c}^{(n)} \in \hat{C}_1$ is sent by one party (that we here assume is $\mathcal{R}$) in such a way that for every $i \in \{1,\dots,n\}$ and for every $k \in \{1,\dots,n\}$ the value $c_i^{(k)}$ is sent through channel i. Assume moreover that for every k a vector $\vec{a}^{(k)} \in \mathbb{F}_q^{t+\varepsilon t}$ is defined, such that $\vec{a}^{(k)}$ uniquely defines $\vec{c}^{(k)}$ by the reconstruction property of $\hat{C}_1$. Suppose finally that for every i the vector $\vec{a}^{(k)}$ with $k=i$ is sent through channel i.

Assume that for $i \in \{1,\dots,n\}$ respectively $\vec{d}^{(i)} \in \mathbb{F}_q^n$ and $\vec{w}^{(i)} \in \mathbb{F}_q^{t+\varepsilon t}$ are received. Then, initially the other party (that we here assume is $\mathcal{S}$) computes the vectors $\vec{v}^{(i)}$ as the unique codewords of $\hat{C}_1$ defined by the vectors $\vec{w}^{(i)}$.

Suppose that $\mathcal{S}$ needs to send all the information about the values $v_\ell^{(i)}$ with $v_\ell^{(i)} \neq d_\ell^{(i)}$. This would allow $\mathcal{R}$ to recover all the vectors $\vec{v}^{(i)}$, as we saw in the 2-round protocol based on Reed-Solomon codes. As remarked above $\mathcal{S}$ cannot encode the conflicts in the same way here. Hence, he defines for every $i \in \{1,\dots,n\}$ the vector $\vec{z}^{(i)} \in \mathbb{F}_q^{2n}$ as follows:

- $z_\ell^{(i)} = z_{n+\ell}^{(i)} = 0$ if $v_\ell^{(i)} = d_\ell^{(i)}$;
- $z_\ell^{(i)} = v_\ell^{(i)}$, $z_{n+\ell}^{(i)} = d_\ell^{(i)}$ if $v_\ell^{(i)} \neq d_\ell^{(i)}$.

The length of these vectors is linear in n, hence they can be sent reliably using generalized broadcasting with an overhead of $\Omega(1)$.

On the one hand this technique for sending conflicts does not require graphs and their matchings and allows for a constant communication overhead. On the other hand since these vectors must be sent with the generalized broadcasting they represent a computationally inefficient part of the protocol.

A.5 Existence of Suitable Generic Codes

In this section we discuss the existence of the codes $\hat{C}_1$, C_2 and $\hat{C}_3$ which are used in the protocol. In general, bounds from Coding Theory state that there do not exist codes for all combinations of parameters. Reed-Solomon codes have optimal parameters, but they cannot be defined for $q < n$.

We treat each of the three cases separately, even though the procedure is basically the same for each. First of all, we need the following definition.

Definition A.3. *Let* $0 < \lambda < \frac{q-1}{q}$. *Then the* q-ary entropy function H_q *is defined as:*

$$H_q(\lambda) = \lambda \log_q(q-1) - \lambda \log_q(\lambda) - (1-\lambda)\log_q(1-\lambda).$$

For our purposes, we express the q-ary entropy function in the following equivalent form:

$$H_q(\lambda) = \log_q\left(\frac{(q-1)^\lambda}{\lambda^\lambda(1-\lambda)^{(1-\lambda)}}\right).$$

A result which is very important for this analysis is the following.

Theorem A.4 ([Haa09])**.** *Let* C *be an* $[n,k]$*-linear code over* $\mathbb{F}_q$, $d(C)$ *be the minimum distance of* C *and let* $d = \lambda n$ *with* $0 < \lambda < \frac{q-1}{q}$. *Then:*

$$P(d(C) < d) < q^{k+n(H_q(\lambda)-1)}.$$

We prove that our codes $\hat{C}_1$, C_2, $\hat{C}_3$ exist with a non-zero probability for appropriate choices of q, ε, δ, α, μ and λ.

First we analyze the existence of $\hat{C}_1$. We note that the ramp sharing scheme associated with $\hat{C}_1$ and C_1 has (at least) t-privacy and $(1+\varepsilon)t$ reconstruction, while encoding secrets of size δt with $0 < \delta \leq \varepsilon$. Hence, these two codes must have the following properties:

- $n - d(\hat{C}_1) + 1 \leq t + \varepsilon t \;\Rightarrow\; d(\hat{C}_1) \geq t + 1 = n\left(\frac{1}{2+\varepsilon}\right) + 1;$
- $d(C_1^*) - 1 \geq t \;\Rightarrow\; d(C_1^*) \geq t + 1 = n\left(\frac{1}{2+\varepsilon}\right) + 1;$
- $\dim(\hat{C}_1) - \dim(C_1) = \hat{k_1} - k_1 = \delta t.$

By Theorem A.4, we have:

$$P\left(d(\hat{C}_1) \leq n\left(\tfrac{1}{2+\varepsilon}\right)\right) \leq q^{\hat{k_1}+n\left(H_q\left(\frac{1}{2+\varepsilon}\right)-1\right)}$$
$$P\left(d(C_1^*) \leq n\left(\tfrac{1}{2+\varepsilon}\right)\right) \leq q^{nH_q\left(\frac{1}{2+\varepsilon}\right)-k_1}$$

hence we study the inequalities:

$$1 - P\left(d(\hat{C}_1) \leq n\left(\tfrac{1}{2+\varepsilon}\right)\right) \geq 1 - q^{\hat{k_1}+n\left(H_q\left(\frac{1}{2+\varepsilon}\right)-1\right)} > 0$$
$$1 - P\left(d(C_1^*) \leq n\left(\tfrac{1}{2+\varepsilon}\right)\right) \geq 1 - q^{nH_q\left(\frac{1}{2+\varepsilon}\right)-k_1} > 0$$

For the first, we have:

$$\begin{aligned}
&1 - q^{\hat{k_1}+n\left(H_q\left(\frac{1}{2+\varepsilon}\right)-1\right)} > 0 \iff \\
&q^{\hat{k_1}+n\left(H_q\left(\frac{1}{2+\varepsilon}\right)-1\right)} < 1 \iff \\
&q^{\hat{k_1}} < \left(\frac{q(1+\varepsilon)^{\frac{1+\varepsilon}{2+\varepsilon}}}{(2+\varepsilon)(q-1)^{\frac{1}{2+\varepsilon}}}\right)^n. \\
&\text{Since } \left(\frac{q(1+\varepsilon)^{\frac{1+\varepsilon}{2+\varepsilon}}}{(2+\varepsilon)(q-1)^{\frac{1}{2+\varepsilon}}}\right)^n > \left(\frac{q^{\frac{1+\varepsilon}{2+\varepsilon}}}{3}\right)^n = q^{n\left(\frac{1+\varepsilon}{2+\varepsilon}-\log_q(3)\right)},
\end{aligned}$$

the inequality is satisfied for $\hat{k_1} < n\left(\frac{1+\varepsilon}{2+\varepsilon} - \log_q(3)\right)$.

For the second we have:

$$\begin{aligned}
&1 - q^{nH_q\left(\frac{1}{2+\varepsilon}\right)-k_1} > 0 \iff \\
&q^{nH_q\left(\frac{1}{2+\varepsilon}\right)-k_1} < 1 \iff \\
&q^{k_1} > \left(\frac{(2+\varepsilon)(q-1)^{\frac{1}{2+\varepsilon}}}{(1+\varepsilon)^{\frac{1+\varepsilon}{2+\varepsilon}}}\right)^n. \\
&\text{Since } \left(\frac{(2+\varepsilon)(q-1)^{\frac{1}{2+\varepsilon}}}{(1+\varepsilon)^{\frac{1+\varepsilon}{2+\varepsilon}}}\right)^n < \left(3q^{\frac{1}{2+\varepsilon}}\right)^n = q^{n\left(\frac{1}{2+\varepsilon}+\log_q(3)\right)},
\end{aligned}$$

the inequality is satisfied for $k_1 > n\left(\frac{1}{2+\varepsilon} + \log_q(3)\right)$.

Combining the two inequalities and requiring that:

$$\hat{k_1} - k_1 = \delta t = n\left(\frac{\delta}{2+\varepsilon}\right)$$

we get that codes with parameters compatible with those of $\hat{C}_1$ and C_1 exist with probability greater than 0 for sufficiently large n when q, ε and δ satisfy the following property:

$$\frac{\varepsilon-\delta}{2+\varepsilon} > \log_q(9).$$

Now, we study the existence of the code C_2. We recall that the code C_2 should have the following property: $d(C_2) \geq 2t+1 = n\left(\frac{2+\varepsilon}{2}\right)+1$, with

$k_2 \geq \alpha t = n\left(\frac{\alpha}{2+\varepsilon}\right)$. We want that:

$$1 - P\left(d(C_2) \leq n\left(\frac{2+\varepsilon}{2}\right)\right) > 0,$$

hence applying Theorem A.4 we study the inequality:

$$1 - q^{k_2 + n\left(H_q\left(\frac{2}{2+\varepsilon}\right)-1\right)} > 0.$$

Now,

$$\begin{aligned}
&1 - q^{k_2 + n\left(H_q\left(\frac{2}{2+\varepsilon}\right)-1\right)} > 0 \iff \\
&q^{k_2 + n\left(H_q\left(\frac{2}{2+\varepsilon}\right)-1\right)} < 1 \iff \\
&q^{k_2} < \left(\frac{2^{\frac{2}{2+\varepsilon}}\varepsilon^{\frac{\varepsilon}{2+\varepsilon}}q}{(2+\varepsilon)(q-1)^{\frac{2}{2+\varepsilon}}}\right)^n. \\
&\text{Since } \left(\frac{2^{\frac{2}{2+\varepsilon}}\varepsilon^{\frac{\varepsilon}{2+\varepsilon}}q}{(2+\varepsilon)(q-1)^{\frac{2}{2+\varepsilon}}}\right)^n > \left(\frac{2^{\frac{2}{3}}\varepsilon q^{\frac{\varepsilon}{2+\varepsilon}}}{3}\right)^n = q^{n\left(\frac{\varepsilon}{2+\varepsilon}+\log_q\left(\frac{\sqrt[3]{4}\varepsilon}{3}\right)\right)},
\end{aligned}$$

the inequality is satisfied for $k_2 < n\left(\frac{\varepsilon}{2+\varepsilon} + \log_q\left(\frac{\sqrt[3]{4}\varepsilon}{3}\right)\right)$. Hence, for sufficiently large n and parameters q, ε and α with:

$$\begin{gathered}
\frac{\alpha}{2+\varepsilon} < \frac{\varepsilon}{2+\varepsilon} + \log_q\left(\frac{\sqrt[3]{4}\varepsilon}{3}\right) \quad \text{i.e.,} \\
\frac{\varepsilon - \alpha}{2+\varepsilon} > \log_q\left(\frac{3}{\sqrt[3]{4}\varepsilon}\right)
\end{gathered}$$

a code C_2 with the desired parameters exists with probability greater than 0.

Finally, we study the existence of the codes $\hat{C}_3$ and C_3. The codes $\hat{C}_3$ and C_3 used in our protocol have the following properties:

- Their length is $N = n + (\mu+\lambda)t = (2+\varepsilon+\mu+\lambda)t$, with $0 < \mu \leq \varepsilon$, $0 < \lambda \leq 1+\varepsilon-\mu$;
- The reconstruction threshold of the ramp sharing scheme based on them is at most $n+\mu t = (2+\varepsilon+\mu)t$, hence $d(\hat{C}_3) > \lambda t = N\left(\frac{\lambda}{2+\varepsilon+\mu+\lambda}\right)$;
- The privacy threshold is at least $(1+\mu)t$, hence $d(C_3^*) > (1+\mu)t = N\left(\frac{1+\mu}{2+\varepsilon+\mu+\lambda}\right)$;

- $\dim(\hat{C_3}) - \dim(C_3) = \hat{k_3} - k_3 = \lambda t.$

We follow a similar approach to the one we used for the codes $\hat{C_1}$ and C_1. We want to prove that such codes $\hat{C_3}$ and C_3 exists with probability greater than 0, hence we study the two inequalities:

$$1 - q^{\hat{k_3} + N\left(H_q\left(\frac{\lambda}{2+\varepsilon+\mu+\lambda}\right)-1\right)} > 0$$
$$1 - q^{NH_q\left(\frac{1+\mu}{2+\varepsilon+\mu+\lambda}\right)-k_3} > 0$$

For the first, we have:

$$1 - q^{\hat{k_3} + N\left(H_q\left(\frac{\lambda}{2+\varepsilon+\mu+\lambda}\right)-1\right)} > 0 \iff$$
$$q^{\hat{k_3} + N\left(H_q\left(\frac{\lambda}{2+\varepsilon+\mu+\lambda}\right)-1\right)} < 1 \iff$$
$$q^{\hat{k_3}} < \left(\frac{q\lambda^{\frac{\lambda}{2+\varepsilon+\mu+\lambda}}(2+\varepsilon+\mu)^{\frac{2+\varepsilon+\mu}{2+\varepsilon+\mu+\lambda}}}{(2+\varepsilon+\mu+\lambda)(q-1)^{\frac{\lambda}{2+\varepsilon+\mu+\lambda}}}\right)^N.$$

Since $\left(\frac{q\lambda^{\frac{\lambda}{2+\varepsilon+\mu+\lambda}}(2+\varepsilon+\mu)^{\frac{2+\varepsilon+\mu}{2+\varepsilon+\mu+\lambda}}}{(2+\varepsilon+\mu+\lambda)(q-1)^{\frac{\lambda}{2+\varepsilon+\mu+\lambda}}}\right)^N > \left(\frac{\sqrt[5]{4}\min(1,\lambda)q^{\frac{2+\varepsilon+\mu}{2+\varepsilon+\mu+\lambda}}}{5}\right)^N =$
$= q^{N\left(\frac{2+\varepsilon+\mu}{2+\varepsilon+\mu+\lambda} + \log_q\left(\frac{\sqrt[5]{4}\min(1,\lambda)}{5}\right)\right)},$

the inequality holds for:

$$\hat{k_3} < N\left(\frac{2+\varepsilon+\mu}{2+\varepsilon+\mu+\lambda} + \log_q\left(\frac{\sqrt[5]{4}\min(1,\lambda)}{5}\right)\right).$$

For the second inequality we have:

$$1 - q^{NH_q\left(\frac{1+\mu}{2+\varepsilon+\mu+\lambda}\right)-k_3} > 0 \iff$$
$$q^{NH_q\left(\frac{1+\mu}{2+\varepsilon+\mu+\lambda}\right)-k_3} < 1 \iff$$
$$q^{k_3} > \left(\frac{(2+\varepsilon+\mu+\lambda)(q-1)^{\frac{1+\mu}{2+\varepsilon+\mu+\lambda}}}{(1+\mu)^{\frac{1+\mu}{2+\varepsilon+\mu+\lambda}}(1+\varepsilon+\lambda)^{\frac{1+\varepsilon+\lambda}{2+\varepsilon+\mu+\lambda}}}\right)^N.$$

Since $\left(\frac{(2+\varepsilon+\mu+\lambda)(q-1)^{\frac{1+\mu}{2+\varepsilon+\mu+\lambda}}}{(1+\mu)^{\frac{1+\mu}{2+\varepsilon+\mu+\lambda}}(1+\varepsilon+\lambda)^{\frac{1+\varepsilon+\lambda}{2+\varepsilon+\mu+\lambda}}}\right)^N < \left(5q^{\frac{1+\mu}{2+\varepsilon+\mu+\lambda}}\right)^N =$
$= q^{N\left(\frac{1+\mu}{2+\varepsilon+\mu+\lambda} + \log_q(5)\right)},$

the inequality holds for:

$$k_3 > N\left(\frac{1+\mu}{2+\varepsilon+\mu+\lambda} + \log_q(5)\right).$$

Combining the two inequalities and requiring that:

$$\hat{k_3} - k_3 = \lambda t = N\left(\frac{\lambda}{2+\varepsilon+\mu+\lambda}\right),$$

we obtain that a code with parameters compatible with those of C_3 exist with probability greater than 0 for sufficiently large n when q, ε, μ and λ satisfy the following property:

$$\frac{1+\varepsilon-\lambda}{2+\varepsilon+\mu+\lambda} > \log_q\left(\frac{25}{\sqrt[5]{4}\min(1,\lambda)}\right).$$

Furthermore, we require that $\hat{C_3}$ has n-uniformity. By Theorem A.2 we have that for $\hat{C_3}$ having n-uniformity it suffices that:

$$d(\hat{C_3}^*) \geq n+1 = N\left(\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}\right)+1.$$

Hence, after applying Theorem A.4 we study the inequality:

$$1 - q^{NH_q\left(\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}\right)-\hat{k_3}} > 0.$$

We have:

$$\begin{aligned}
&1 - q^{NH_q\left(\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}\right)-\hat{k_3}} > 0 \iff \\
&q^{NH_q\left(\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}\right)-\hat{k_3}} < 1 \iff \\
&q^{\hat{k_3}} > \left(\frac{(2+\varepsilon+\mu+\lambda)(q-1)^{\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}}}{(2+\varepsilon)^{\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}}(\mu+\lambda)^{\frac{\mu+\lambda}{2+\varepsilon+\mu+\lambda}}}\right)^N. \\
&\text{Since } \left(\frac{(2+\varepsilon+\mu+\lambda)(q-1)^{\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}}}{(2+\varepsilon)^{\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}}(\mu+\lambda)^{\frac{\mu+\lambda}{2+\varepsilon+\mu+\lambda}}}\right)^N < \left(\frac{5q^{\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}}}{\min(1,\mu+\lambda)}\right)^N = \\
&= q^{N\left(\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}+\log_q\left(\frac{5}{\min(1,\mu+\lambda)}\right)\right)},
\end{aligned}$$

the inequality holds for:

$$\hat{k_3} > N\left(\frac{2+\varepsilon}{2+\varepsilon+\mu+\lambda}+\log_q\left(\frac{5}{\min(1,\mu+\lambda)}\right)\right).$$

Combining this inequality with the upper bound for $\hat{k_3}$ obtained previously, we have that $\hat{C_3}$ has n-uniformity with probability bigger than 0 if the following condition on q, ε, μ and λ holds:

$$\frac{\mu}{2+\varepsilon+\mu+\lambda} > \log_q\left(\frac{25}{\sqrt[5]{4}\min(1,\lambda)\min(1,\mu+\lambda)}\right).$$

It is easy to see that the four conditions for the existence of the codes $\hat{C}_1$, C_2 and $\hat{C}_3$ are compatible with each other, i.e., for sufficiently large n there exist q, ε, δ, α, μ and λ which satisfy all the three conditions. This concludes our analysis.