Contribution à la cryptanalyse de primitives cryptographiques fondées sur la théorie des codes

Ayoub Otmani

UNIVERSITÉ DE CAEN BASSE-NORMANDIE

MÉMOIRE D'HABILITATION À DIRIGER DES RECHERCHES

Spécialité Informatique

Contribution à la cryptanalyse de primitives cryptographiques fondées sur la théorie des codes

Ayoub OTMANI

6 décembre 2011

RAPPORTEURS

Daniel AUGOT	Directeur de Recherche, INRIA
Kazukuni KOBARA	Principal research scientist, National Institute AIST (Japon)
Ruud PELLIKAAN	Professor, Eindhoven University of Technology (Pays-Bas)

JURY

Daniel AUGOT	Directeur de Recherche, INRIA
Claude CARLET	Professeur, Université Paris VIII
Kazukuni KOBARA	Principal research scientist, National Institute AIST (Japon)
Ruud PELLIKAAN	Professor, Eindhoven University of Technology (Pays-Bas)
Brigitte VALLÉE	Directrice de Recherche, CNRS
Pascal VÉRON	Maître de conférences, Université du Sud Toulon-Var
Gilles ZÉMOR	Professor, Université de Bordeaux I

Contribution to the Cryptanalysis of Code-Based Primitives

Ayoub OTMANI

December 6, 2011

Contents

Chapter 1

Preamble

A large part in the design of secure cryptographic primitives consists in identifying hard algorithmic problems. Despite the fact that several problems have been proposed as a foundation for public-key primitives, those effectively used are essentially classical problems coming from integer factorization and discrete logarithm. On the other hand, coding theory appeared with the goal to solve an important and challenging problem: *how to transmit reliably information in noisy environments?* It turns out that this task is closely related to the famous problem of decoding a random linear code. It is widely admitted as a hard problem that has led McEliece in 1978 to propose the first code-based public-key encryption scheme. The key concept of the scheme is to focus on codes that come up with an efficient decoding algorithm. He also advocated the use of classical binary Goppa codes. Since then, it still belongs to the very few cryptosystems which remain unbroken. All the existing attacks that recover either the private key or basically the plaintext from a single ciphertext are exponential in time.

This thesis is primarily interested in studying the security of code-based primitives. The first category of primitives we analyzed is composed of variants of the McEliece cryptosystem. A *variant* is any public-key encryption schemes that replace classical binary Goppa codes with other codes provided that they can be decoded efficiently. In the past several proposals were suggested but most of them did not withstand attacks that recover the private key. Another motivation for this kind of change is to remedy the main disadvantage of McEliece's scheme of having too large public keys. One trend in the area of the code-based cryptography is to use highly structured codes which can be described by matrices totally defined from only few rows. Our works show that this phenomenon also apply to many compact variants because it is possible to recover practically the private key, either by mounting dedicated attacks, or by devising attacks of algebraic nature that are general enough to also apply to the original McEliece cryptosystem. This result represents a new algebraic framework to assess the security of the McEliece cryptosystem and a first step towards the design of new attacks based on the solving of algebraic systems (Gröbner bases, *etc.*)

Furthermore, this approach can be used to study a famous problem encountered in code-based cryptography called the problem of the *Goppa code Distinguishing* problem. It asks whether there is an efficient way to distinguish a Goppa code from a randomly drawn linear code. It represents an important assumption which supports the use of Goppa codes in cryptography. It also makes possible to link the problem of decoding a random linear code to that of decoding a Goppa code. Thanks to it, it can provide a security proof for the McEliece cryptosystem and other cryptographic primitives based on Goppa codes. We show that it is possible to efficiently solve it as long as the code rate is sufficiently high. We show more precisely that it is possible to differentiate between an alternate code, a Goppa code and a random linear code with high probability. The solving is possible through the construction of a linear system deduced from an algebraic system that any Goppa code and, more generally, any alternant code must satisfy. It is observed that the rank of this system has different values depending on the code we consider. We are even able to predict its value and also provide an explanation.

Finally, we investigate the security of a signature scheme proposed by Kabatianskii-Krouk-Smeets. It relies on two random linear codes, one is public whereas the other has to be secret. We first evaluate the security of the scheme against a passive attack. Then, we show that it can be completely broken thanks to the construction of an auxiliary linear code from the public key. From it, the private key of the scheme is then recovered by looking for low-weight codewords. Although, the time complexity of our attack is exponential in the length of the codes, our analysis shows that the attack is sensitive to the rates of codes and therefore can be practical if the values of the rates are very close.

The thesis is divided in two parts. The first one is an introduction to the area of the coding theory and code-based cryptography. It also provides the state of the art in that field. It exposes the important challenges that are encountered in code-based cryptography, and in particular with the McEliece cryptosystem. The second part deals specifically with my recent contributions that appeared in proceedings of international conferences.

Part I

Code-Based Cryptography

Chapter 2

Algorithmic Issues

In this chapter, we describe important algorithmic problems arising from the theory of linear error-correcting codes. These problems will serve as a foundation for building and assessing the security of code-based cryptographic primitives.

The key notion that we will require first is that of *linear code* over a finite field $\mathbb{F}_q$ with q elements. Such an object is basically a vector subspace of $\mathbb{F}_q^n$. The *dimension* k of a (linear) code is its dimension as a linear space. The code $\mathscr{C}$ obviously admits a basis of cardinality k. any $k \times n$ matrix $\boldsymbol{G}$ obtained from a basis of $\mathscr{C}$ is called a *generator matrix*, or equivalently:

$$\mathscr{C} = \left\{ \boldsymbol{u}\boldsymbol{G} : \boldsymbol{u} \in \mathbb{F}_q^k \right\}.$$

The *rationale* of a linear code is to convey *reliably and efficiently* an information between a *transmitter* and a *recipient* through a noisy link. This link is also termed *channel*. The information is represented as a sequence $\boldsymbol{u} = (u_1, \ldots, u_k)$ of k symbols taken from a finite alphabet $\mathbb{F}_q$. Any attempt of sending $\boldsymbol{u}$ will inevitably results in a modification of a fraction of the symbols u_j. Actually the situation is worse because these modifications are not *a priori* predictable. So any realization in $\mathbb{F}_q^k$ can be in theory a possible event. The probability of obtaining one will completely depend on the channel. The latter can actually be seen as a random process $\eta : X \to Y$ whose behaviour is described by transition probabilities $P\left(\eta(z) = y \mid z = x\right)$ for each x in X and y in Y. The X and Y are the *input* and *output* alphabets respectively. Generally, it is implicitly assumed that $X \subset Y$ and there exists a one-to-one correspondence between $\mathbb{F}_q$ and X.

Our focus will be on the q-ary *symmetric channel of error probability* $0 \leqslant p < 1/2$ where $X = Y$, both of cardinality q, and such that $P(\eta(z) = x \mid z = x) = 1 - p$ and $P(\eta(z) \neq x \mid z = x) = \frac{p}{q-1}$. This kind of channel is memoryless which means that the probability of obtaining one symbol at the output does not depend on the previous symbols transmitted in the past. Mathematically, it is equivalent to say that the events are independent.

One would believe that the problem of reliably transmitting information can be solved by simply repeating the symbols a given number of times. Recovering the initial symbol can be done by looking at which symbol appears most. This technique is called *majority-vote decision*. In this context, the transmitted sequences are of the form $u \cdots u$ where u belongs to an alphabet, say $\{0, 1\}$, is repeated for instance $n = 2m + 1$ times for some integer $m > 1$. The symbol u represents the information to be sent, so that $k = 1$. One can prove that indeed the error probability of the majority-vote decision algorithm can be made arbitrary small provided that n is sufficiently large. So *why is it not acceptable?*

The main problem with this approach rests on its *inefficiency* namely the cost of transmitting and processing one single symbol tends to infinity. This is clearly unrealistic in practical scenarios. Actually, this cost is captured by the ratio $R \stackrel{\text{def}}{=} k/n$ also called the *rate* of the coding system. In reality, this rate is imposed by the context and therefore cannot be modified, and in particular decreased to be as small as wished.

A classical way to deal with this matter is to choose a generator matrix $\boldsymbol{G}$ of a linear code $\mathscr{C}$ of length n and dimension $k < n$ over $\mathbb{F}_q$, and instead of directly sending the information $\boldsymbol{u}$ belonging to $\mathbb{F}_q^k$, one transmits rather the sequence $\boldsymbol{c} \stackrel{\text{def}}{=} \boldsymbol{u}\boldsymbol{G}$. The operation of transforming $\boldsymbol{u}$ into $\boldsymbol{c}$ is known as *encoding* of information. The task of the *receiver*, upon receiving the sequence $\boldsymbol{r}$, which is the result of the transmission of $\boldsymbol{c}$ amidst the noise, is to recover $\boldsymbol{u}$. This phase of recovering $\boldsymbol{u}$ is called the *decoding*. Therefore, the receiver has to define a *decision rule* ensuring him that, with high probability, he has made the *good* choice. It also means that it is paramount for the receiver to detect errors when they occur. A way to achieve this is to equip the ambient space $\mathbb{F}_q^n$ with an *inner product* defined for any $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_n)$ of $\mathbb{F}_q^n$ as $\boldsymbol{x} \cdot \boldsymbol{y} \stackrel{\text{def}}{=} \sum_{j=1}^{n} x_j y_j$. It permits to define the *dual* code of $\mathscr{C}$ as the *linear* space $\mathscr{C}^\perp \stackrel{\text{def}}{=} \left\{ \boldsymbol{y} \in \mathbb{F}_q^n : \forall \boldsymbol{c} \in \mathscr{C},\ \boldsymbol{y} \cdot \boldsymbol{c} = 0 \right\}$. Clearly, the dimension of $\mathscr{C}^\perp$ is $n - k$. Furthermore, any $(n - k) \times n$ generator matrix of $\mathscr{C}^\perp$ characterizes $\mathscr{C}$ because we have:

$$\mathscr{C} = \left\{ \boldsymbol{c} \in \mathbb{F}_q^n : \sum_{j=1}^{n} c_j \boldsymbol{H}_j = \boldsymbol{0} \right\}$$

where $\boldsymbol{H}_j$ denotes the j-th column of $\boldsymbol{H}$. Such a matrix $\boldsymbol{H}$ is called a *parity-check matrix* of $\mathscr{C}$. It becomes straightforward for the receiver to know if there exists an error in $\boldsymbol{r}$. It simply checks whether the quantity $\sum_{j=1}^{n} r_j \boldsymbol{H}_j$ called the *syndrome* of $\boldsymbol{r}$, is equal to $\mathbf{0}$. If ever an error is detected, it remains to find the corresponding information. A reasonable[1] decoding rule is to choose *the most likely word* $\hat{\boldsymbol{c}}$ among $\mathscr{C}$, or formally, $\hat{\boldsymbol{c}}$ maximizes the probability $P(\boldsymbol{r}|\boldsymbol{c})$ when $\boldsymbol{c}$ describes $\mathscr{C}$. Note that since we have assumed the symbols are conveyed by means of a memoryless channel we therefore have:

$$P(\boldsymbol{r} \mid \boldsymbol{c}) = \prod_{j=1}^{n} P\Big(\eta(z) = r_j \mid z = c_j\Big).$$

An obvious method to perform this search is to check all q^k words of $\mathscr{C}$ which gives a procedure whose time complexity is $\mathcal{O}\left(nq^k\right) = \mathcal{O}\left(nq^{Rn}\right)$. This is of course unsatisfactory.

This discussion explains that one important challenge of coding theory is to find, given a fixed rate $R < 1$, a family of linear codes $(\mathscr{C}_n)_{n\geqslant 1}$ of length n and dimension $k = Rn$ where $\mathscr{C}_n$ admits a polynomial time in n decoding algorithm such that the decoding error probability is arbitrary small[2] as n tends to infinity.

2.1 Minimum Distance Decoding

The optimal decoding rule for a code $\mathscr{C}$ consists in maximizing the probability $P(\boldsymbol{r} \mid \boldsymbol{c})$ when $\boldsymbol{c}$ describes $\mathscr{C}$. Actually, one can prove that for most common channels this resort to *searching for the closest word to* $\boldsymbol{r}$ in $\mathscr{C}$ for an appropriate metric. For instance, in the case of the q-ary symmetric channel, the metric to choose is the Hamming's one. The Hamming *distance* (or metric) denoted by $\mathsf{dist}(\boldsymbol{x}, \boldsymbol{y})$ between any two vectors $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_n)$ from $\mathbb{F}_q^n$ is the cardinality of the set $\{j \in \{1, \ldots, n\} : x_j \neq y_j\}$. The *weight* $\mathsf{wt}(\boldsymbol{x})$ of any $\boldsymbol{x}$ in $\mathbb{F}_q^n$ is $\mathsf{dist}(\boldsymbol{x}, \mathbf{0})$ *i.e.* the cardinality of $\{j : x_j \neq 0\}$. We are now able to give the following definition where we have assumed that $\mathscr{C}$ is a code over $\mathbb{F}_q$ of length n and dimension k.

Definition 1. *A* minimum distance decoding *of* $\mathscr{C}$ *is a mapping* $D : \mathbb{F}_q^n \longrightarrow \mathscr{C}$ *such that the following holds*[3] *for any* $\boldsymbol{z}$ *in* $\mathbb{F}_q^n$*:*

$$\forall \boldsymbol{c} \in \mathscr{C}, \;\; \mathsf{dist}\left(\boldsymbol{z}, D(\boldsymbol{z})\right) \leqslant \mathsf{dist}\left(\boldsymbol{z}, \boldsymbol{c}\right)$$

We have seen that a naive approach to solve the minimum distance decoding for a given received word $\boldsymbol{r}$ is exponential in n. One may ask whether a better strategy can be devised. Before, we will reformulate the problem in order study it in the realm of the theory of complexity. First, let us set $\boldsymbol{s} \stackrel{\text{def}}{=} \sum_{j=1}^{n} \boldsymbol{r}_j \boldsymbol{H}_j$. By remarking that $\boldsymbol{s} = \sum_{j=1}^{n} (\boldsymbol{r}_j + \boldsymbol{c}_j)\boldsymbol{H}_j$ for any $\boldsymbol{c}$ in $\mathscr{C}$, we see that minimum distance decoding is equivalent to looking for a vector $\boldsymbol{e}$ in $\mathbb{F}_q^n$ of *minimum weight* such that:

$$\boldsymbol{s} = \sum_{j=1}^{n} \boldsymbol{e}_j \boldsymbol{H}_j. \tag{2.1}$$

This new way of tackling the problem enables to define an associated decision problem called the *Syndrome Decoding problem*.

Definition 2 (Syndrome Decoding Decision Problem)**.** *The inputs are an* $(n-k)\times n$ *matrix* $\boldsymbol{H}$ *with entries in* $\mathbb{F}_q$ *with* $1 \leqslant k < n$*, an integer* t *such that* $t < n$ *and a vector* $\boldsymbol{s}$ *in* $\mathbb{F}_q^{n-k}$*. The question is* does there exist $\boldsymbol{e}$ in $\mathbb{F}_q^n$ such that $\mathsf{wt}(\boldsymbol{e}) \leqslant t$ and $\boldsymbol{s} = \sum_{j=1}^{n} \boldsymbol{e}_j \boldsymbol{H}_j$?

Proposition 1 ([BMvT78])**.** *The Syndrome Decoding problem is* NP*-Complete.*

Observe that the matrix $\boldsymbol{H}$ is a part of the input but we can also investigate the problem without assuming it, that is to say, we suppose that $\boldsymbol{H}$ is given once and for all and an arbitrary large preprocessing is performed on it. The problem remains however a hard problem [BN90].

2.2 Bounded Distance Decoding

The NP-Completeness results about minimum distance decoding show it is illusory to expect that a general polynomial time minimum distance decoding exists. But the situation can be different if some constraints are relaxed. In fact, since its introduction by Shannon [Sha48], the coding theory has spawned a large amount of works dealing with the construction of family of codes provided with polynomial time decoding algorithms. This is possible because of two main reasons. First, some specific structure are added to linear codes and secondly, some limitations are put on the type of errors that can be corrected. It is important to keep in mind that, whatever the decoding procedure is, some errors would necessarily lead to take either wrong decisions (decoding

[1] Actually it is the optimal rule because it achieves the smallest decoding error probability.

[2] Shannon showed in [Sha48] than the rate R has to be smaller than or equal to a quantity called the *capacity* $C < 1$ of the channel. The real change is in reality to find capacity-approaching codes for which the decoding error probability can be made arbitrary small.

[3] Let us observe that we necessarily have $D(\boldsymbol{c}) = \boldsymbol{c}$ for any $\boldsymbol{c}$ in $\mathscr{C}$.

error), or simply no decision at all (decoding *failure*). So one has to identify to which class an error should belong so that it can be decoded correctly.

The answer is to interpret decoding inside a geometric setting. The minimum distance decoding of a received word $\boldsymbol{r}$ consists in finding the nearest codeword $\boldsymbol{c}$ in $\mathscr{C}$. Obviously, when the solution is unique then the decoder has no choice. But when there are several solutions, one arbitrary choice has to be made leading necessarily in some cases to wrong decisions. It is therefore important to define regions where the decoding is *unique*. The notion of *minimum distance* of a code $\mathscr{C}$ will serve us to do so.

Definition 3. *The* minimum distance *is* $d \stackrel{def}{=} \min\left\{\mathsf{dist}(\boldsymbol{x},\boldsymbol{y}) : \boldsymbol{x} \in \mathscr{C}, \boldsymbol{y} \in \mathscr{C}, \boldsymbol{x} \neq \boldsymbol{y}\right\}$.

One can easily see that $d = \min\left\{\mathsf{wt}(\boldsymbol{c}) : \boldsymbol{c} \in \mathscr{C}, \boldsymbol{c} \neq \boldsymbol{0}\right\}$ when $\mathscr{C}$ is a linear code. Furthermore, if we define $t \stackrel{\text{def}}{=} \left\lfloor \frac{d-1}{2} \right\rfloor$ then for any $\boldsymbol{c}$ in $\mathscr{C}$, we have $\mathcal{B}(\boldsymbol{c},t) \cap \mathscr{C} = \{\boldsymbol{c}\}$ where by definition $\mathcal{B}(\boldsymbol{c},t) \stackrel{\text{def}}{=} \left\{\boldsymbol{z} \in \mathbb{F}_q^n : \mathsf{dist}(\boldsymbol{c},\boldsymbol{z}) \leqslant t\right\}$. Hence, if the number of errors is less than or equal to t and if we have an efficient way for searching for codewords of $\mathscr{C}$ in balls of radius t then the decoding is unique and correct. The parameter t is called the *packing radius*. Furthermore, the method that consists in searching for a codeword within a given distance $\ell \geqslant 1$ around the received word $\boldsymbol{r}$ is called an *ℓ-bounded distance decoding* as explained in the following definition.

Definition 4. *Let ℓ be an integer $\geqslant 1$. An* ℓ-bounded distance decoding *is the mapping* $f_\ell : \bigcup_{c \in \mathscr{C}} \mathcal{B}(\boldsymbol{c},\ell) \longrightarrow \mathscr{C}$ *such that for any $\boldsymbol{z}$ in $\cup_{c\in\mathscr{C}}\mathcal{B}(\boldsymbol{c},\ell)$:*

$$\forall \boldsymbol{c} \in \mathscr{C}, \;\; \mathsf{dist}\left(\boldsymbol{z}, f_\ell(\boldsymbol{z})\right) \leqslant \mathsf{dist}\left(\boldsymbol{z},\boldsymbol{c}\right)$$

If it happens that $\cup_{\boldsymbol{c}\in\mathscr{C}}\mathcal{B}(\boldsymbol{c},t)$ forms a partition of $\mathbb{F}_q^n$, the code $\mathscr{C}$ is called *perfect*. For such codes, the t-bounded minimum distance decoding could not fail at outputting the correct codeword of $\mathscr{C}$ when of course the number of errors is less than or equal to t.

An easy way to find a codeword within distance t from $\boldsymbol{r}$ is to exhaustively search for $\boldsymbol{e}$ in $\mathcal{B}(\boldsymbol{0},t)$ until $\sum_{j=1}^{n}(r_j + e_j)\boldsymbol{H}_j = \boldsymbol{0}$. Since the number of elements in $\mathcal{B}(\boldsymbol{r},t)$ is $\sum_{i=1}^{t}(q-1)^i\binom{n}{i}$ and by observing that $\sum_{i=1}^{t}(q-1)^i\binom{n}{i} \leqslant q^{nh_q(\frac{t}{n})}$ where $h_q(x)$ is the entropy function defined over $[0,1]$ with $h_q(x) \stackrel{\text{def}}{=} x\log_q(q-1) - x\log_q(x) - (1-x)\log_q(1-x)$, this obvious method has a time complexity $\mathcal{O}\left(nq^{nh_q(\frac{t}{n})}\right)$. The complexity of this basic procedure depends on the value t which in turn is about half of the minimum distance of the code. Hence, in order to obtain a more complexity, we have to ask first whether it is easy to compute the minimum distance. The answer is unfortunately negative as shown by [Var97].

Definition 5 (Minimal Weight Decision Problem)**.** *The inputs are an $(n-k)\times n$ matrix $\boldsymbol{H}$ with entries in $\mathbb{F}_q$ with $1 \leqslant k < n$, an integer t such that $t < n$. The question is* does there exist a *nonzero* $\boldsymbol{c}$ in $\mathbb{F}_q^n$ such that: $\mathsf{wt}(\boldsymbol{c}) \leqslant t$ and $\sum_{j=1}^{n} c_j\boldsymbol{H}_j = \boldsymbol{0}$?

Proposition 2 ([Var97])**.** *The Minimal Weight Problem is NP-Complete*[4].

Since the existence of a general polynomial time algorithm that computes the exact value is unlikely to exist, one may ask what it is its *typical* value.

Definition 6. *A given property is said to be* typical *for a code if the probability that a code picked uniformly at random satisfied it tends to 1 when the length n tends to $+\infty$. We will also say this feature is satisfied for* almost all *codes.*

We are now able to state the following proposition.

Proposition 3 ([Bar98])**.** *The minimum distance d for almost all codes of rate R satisfies $d \geqslant n\delta_{\mathrm{GV}}(R)$ where $\delta_{\mathrm{GV}}(R)$ is the* relative Gilbert-Varshamov distance *which is defined as the smallest positive root of the equation:*

$$h_q(x) + R - 1 = 0. \tag{2.2}$$

However, the bounded distance decoding within the packing radius t is most of the time pessimistic because generally for *most* $\boldsymbol{v}$ in $\mathbb{F}_q^n$ the ball $\mathcal{B}(\boldsymbol{v},\ell)$ when $\ell > t$ will contain at most one codeword from $\mathscr{C}$. In fact, the only interest about t is the guarantee of *always* having at most one codeword in $\mathcal{B}(\boldsymbol{v},t)$ for *any* $\boldsymbol{v}$. But it does not prevent from a decoding failure. Hence, the important question is to known what is the radius ℓ that guarantees a bounded distance decoding with similar performance as minimum distance decoding.

That is precisely the role of an other important numerical parameter called the *covering radius* ρ. It is the smallest radius for which for any $\boldsymbol{v}$ in $\mathbb{F}_q^n$ the ball $\mathcal{B}(\boldsymbol{v},\rho)$ contains *at least* one codeword of $\mathscr{C}$. Clearly we have $t \leqslant \rho$ and one can see that an alternative definition of ρ is as follows in which $\mathsf{dist}(\boldsymbol{z},\mathscr{C})$ denotes for any $\boldsymbol{z}$ from $\mathbb{F}_q^n$ the following quantity:

$$\mathsf{dist}(\boldsymbol{z},\mathscr{C}) \stackrel{\text{def}}{=} \min\left\{\mathsf{dist}(\boldsymbol{z},\boldsymbol{c}) : \boldsymbol{c} \in \mathscr{C}\right\}.$$

[4]The problem is still NP-Complete [BMvT78] if in the statement of Proposition 2, it is required to have a word of weight exactly t.

Definition 7. *The* covering radius *of $\mathscr{C}$ is* $\rho \stackrel{def}{=} \max\left\{\mathsf{dist}(\boldsymbol{z},\mathscr{C}) : \boldsymbol{z} \in \mathbb{F}_q^n\right\}$.

The covering radius is important because it measures the largest number of errors within which it is still possible to get an error-free minimum distance decoding. This means that if the error pattern has weight strictly greater than ρ then minimum distance decoding will *always* return a wrong codeword. Hence computing ρ permits to known the best achievement one could expect from $\mathscr{C}$. The intuition would suggest computing the covering radius is more difficult than decoding since for the naive approach would look for the closest vector in $\mathscr{C}$ for *each* $\boldsymbol{r}$ of $\mathbb{F}_q^n$. In a sense, one has to do q^n decodings. So it raises the questions of knowing the complexity of computing the covering radius in one hand and, in the other hand, its *typical* value when a code is picked randomly. To answer to first point, it is possible to define a decision problem that measures the difficulty of computing the covering radius. Again, the problem is likely to be difficult as shown by Proposition 4.

Definition 8 (Upper bound on covering radius problem)**.** *Given a linear code $\mathscr{C}$ and $t \geqslant 0$. Is it true that:*

$$\forall \boldsymbol{z} \in \mathbb{F}_q^n, \exists \boldsymbol{c} \in \mathscr{C}, \ \ \mathsf{dist}\left(\boldsymbol{z},\boldsymbol{c}\right) \leqslant t?$$

Proposition 4 ([McL84])**.** *The upper bound on covering radius problem is Π_2^p-Complete*

As for the second question, the answer is given by the following proposition.

Proposition 5 ([Bar98])**.** *The covering radius for almost codes of rate R is $n\delta_{\mathrm{GV}}(R)\left(1+o(1)\right)$ where $o(1)$ tends to 0 when n tends to $+\infty$.*

This results explains why it suffices to focus on searching for a codeword within distance $n\delta_{\mathrm{GV}}(R)$ from the received word $\boldsymbol{r}$. As a consequence, the complexity of the obvious way that consists in looking for a codeword in $\mathcal{B}\left(\boldsymbol{r}, n\delta_{\mathrm{GV}}(R)\right)$ is $\mathcal{O}\left(nq^{n(1-R)}\right)$. Let us observe that when $R < 1/2$ the exhaustive search is faster than the bounded distance decoding within distance $n\delta_{\mathrm{GV}}(R)$.

2.3 General Decoding Methods

The decoding of a random linear code received a lot of attention over the past years from a practical point of view. Several works [McE78, LB88, Leo88, Ste88, vT90, CC94, CC95, Dum96, CC98, CS98, BLP08] propose to solve it for an arbitrary *binary* linear code $\mathscr{C}$ of length n and dimension k that is given either as a generator matrix $\boldsymbol{G}$ or a parity-check matrix $\boldsymbol{H}$. The question to solve is then, given a word $\boldsymbol{r}$ from $\mathbb{F}_2^n$ and an integer $t \geqslant 1$, find a codeword $\boldsymbol{c}$ from $\mathscr{C}$ such that $\mathsf{dist}(\boldsymbol{r},\boldsymbol{c}) \leqslant t$. This boils down to seeking $\boldsymbol{e}$ in $\mathbb{F}_2^n$ such that $\mathsf{wt}(\boldsymbol{e}) \leqslant t$ and $\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e}$.

2.3.1 Information Set decoding

General approach. One of the most natural approach is to locate a set $U \subset \{1,\ldots,n\}$ of cardinality k such that the restriction $\boldsymbol{G}_U$ over U of $\boldsymbol{G}$ is *invertible and* $\boldsymbol{e}_U = \boldsymbol{0}$. Indeed, it can readily be checked that the following equality holds:

$$\boldsymbol{c} = \left(\boldsymbol{r}_U \boldsymbol{G}_U^{-1}\right)\boldsymbol{G}$$

where of course $\boldsymbol{r}_U$ is the restriction of $\boldsymbol{r}$ to the positions belonging to U. Such a set U is called an *information set*. The simple observation first made by McEliece in [McE78] shows that the method of randomly picking a set U of cardinality k where $\boldsymbol{G}_U$ is invertible has a success probability P_{IS} given by:

$$P_{IS} \stackrel{\text{def}}{=} \frac{\binom{n-t}{k}}{\binom{n}{k}} = \frac{(n-t)!}{k!(n-t-k)!}\frac{k!(n-k)!}{n!} = \frac{\binom{n-k}{t}}{\binom{n}{t}}.$$

If we define $\tau \stackrel{\text{def}}{=} t/n$ and $R \stackrel{\text{def}}{=} k/n$ then there exist [MS86, p. 309] two positive constants a and b such that P_{IS} can be upper-bounded by:

$$a \cdot 2^{-n\left(h_2(\tau)-(1-R)h_2\left(\frac{\tau}{1-R}\right)\right)} \leqslant P_{IS} \leqslant b \cdot 2^{-n\left(h_2(\tau)-(1-R)h_2\left(\frac{\tau}{1-R}\right)\right)}.$$

This method performs for each candidate of U a Gaussian elimination over a $k \times k$ matrix. So the time complexity is about $k^\omega/P_{IS} = \mathcal{O}\left(n^\omega 2^{n\left(h_2(\tau)-(1-R)h_2\left(\frac{\tau}{1-R}\right)\right)}\right)$ where $2 < \omega \leqslant 3$ is the "linear algebra constant".

This general approach for searching a set U where the restriction of the generator matrix is invertible is termed *Information Set Decoding*, or ISD for short. Many works used it with the goal to improve its efficiency. There exist two ways for enhancing ISD principle: one aims at avoiding to perform too many Gaussian eliminations, and the other attempts to increase the probability of getting a good candidate for the set U. Let us observe that if we set $\boldsymbol{s} \stackrel{\text{def}}{=} \sum_{j=1}^{n} r_j \boldsymbol{H}_j$ where $\boldsymbol{H}$ is an $(n-k)\times n$ parity-check matrix of $\mathscr{C}$ then the problem to solve is to find $\boldsymbol{e}$ from $\mathbb{F}_2^n$ of weight t such that (2.1) holds. It is equivalent to to find a set $E \subset \{1,\ldots,n\}$ of cardinality $\leqslant t$ such that we have:

$$\sum_{j\in E} \boldsymbol{H}_j = \boldsymbol{s} \tag{2.3}$$

where $\boldsymbol{H}_j$ is j-th column of $\boldsymbol{H}$. We now present the most important algorithms based on ISD. They all assume that, up to a permutation of the columns, $\boldsymbol{H}$ is in systematic form $\boldsymbol{H} = \begin{pmatrix} \boldsymbol{I}_{n-k} & \boldsymbol{A} \end{pmatrix}$ where $\boldsymbol{A}$ is a $(n-k) \times k$ submatrix of $\boldsymbol{H}$ and $\boldsymbol{I}_{n-k}$ is the identity matrix of size $(n-k)$. For the sake of simplicity, we will always assume from now that $U = \{n-k+1, \ldots, n\}$. In particular, when $\boldsymbol{r}$ is error-free over U ($\boldsymbol{e}_U = \boldsymbol{0}$) then the following equality happens:

$$\boldsymbol{s} = \begin{pmatrix} \boldsymbol{I}_{n-k} & \boldsymbol{A} \end{pmatrix} \boldsymbol{e}^T = \boldsymbol{I}_{n-k} \cdot \boldsymbol{e}^T = \boldsymbol{e}^T.$$

where $\boldsymbol{a}^T$ denotes the transpose of $\boldsymbol{a}$. In that case $\boldsymbol{s}$ satisfies $\mathsf{wt}(\boldsymbol{s}) \leqslant t$. Hence, the whole goal of ISD is to find a parity-check matrix $\boldsymbol{H}$ in systematic form so that $\boldsymbol{s}$ is exactly the error vector and a way to recognise it is through the condition $\mathsf{wt}(\boldsymbol{H}\boldsymbol{r}^T) \leqslant t$.

Lee-Brickell's approach. The constraint $\boldsymbol{e}_U = \boldsymbol{0}$ on the set U is replaced by the one where the weight of $\boldsymbol{e}_U$ is a small value p. The algorithm enumerates all the sets $P \subset \{n-k+1, \ldots, n\}$ of cardinality p and it halts if the following conditions holds:

$$\mathsf{wt}\left(\boldsymbol{s} + \sum_{j \in P} \boldsymbol{A}_j\right) \leqslant t - p. \tag{2.4}$$

The success probability of this algorithm denoted by P_{LB} is:

$$P_{LB} = \frac{\binom{t}{p}\binom{n-t}{k-p}}{\binom{n}{k}} = \frac{\binom{k}{p}\binom{n-k}{t-p}}{\binom{n}{t}}.$$

The running time of the algorithm is basically the cost for enumerating all $\binom{k}{p}$ linear combinations of the columns of $\boldsymbol{A}$ multiplied by $(n-k)^\omega / P_{LB}$.

Leon's approach. Leon's idea [Leo88] follows Lee-Brickell's constraint but it also requires to look for some random $L \subset \{1, \ldots, n-k\}$ of cardinality $\ell \leqslant n-k$ such that $\boldsymbol{e}_L = \boldsymbol{0}$. For simplicity we assume $L = \{1, \ldots, \ell\}$. The reason for imposing such a condition is to reduce the costs of the Gaussian elimination and the computation of the weight of words. The algorithm is interested in parity-check matrices of the following form:

$$\boldsymbol{H} = \left(\begin{array}{c|c|c} \boldsymbol{X} & \boldsymbol{0} & \boldsymbol{T} \\ \hline \boldsymbol{Y} & \boldsymbol{I}_{n-k-\ell} & \boldsymbol{B} \end{array}\right) \quad \text{with} \quad \boldsymbol{A} = \left(\begin{array}{c} \boldsymbol{T} \\ \hline \boldsymbol{B} \end{array}\right).$$

We emphasize that $\boldsymbol{T}$ is the $\ell \times k$ submatrix of $\boldsymbol{A}$ obtained by keeping only the rows in L, and $\boldsymbol{B}$ is the $(n-k-\ell) \times k$ submatrix of $\boldsymbol{A}$ obtained with the remaining rows. The algorithm first picks a random set L and then enumerates sets $P \subset \{n-k+1, \ldots, n\}$ of cardinality p. Let us denote by $R \subset \{1, \ldots, n-k\}$ the complement of L. We will also denote by $\boldsymbol{s}_L$ and $\boldsymbol{s}_R$ the restrictions of $\boldsymbol{s}$ to the positions in L and R respectively. Whenever it encounters a set P such that $\boldsymbol{s}_L + \sum_{j \in P} \boldsymbol{T}_j = \boldsymbol{0}$, it next checks whether:

$$\mathsf{wt}\left(\boldsymbol{s}_R + \sum_{j \in P} \boldsymbol{B}_j\right) \leqslant t - p. \tag{2.5}$$

The advantage of this algorithm is threefold. First, a Gaussian elimination is performed on a smaller square matrix of size $n-k-\ell$. Secondly, the linear combinations are executed on a relatively small matrix $\boldsymbol{T}$. Eventually, the weight computation is done only when a combination has a good chance to give the solution. As a side remark, let us notice that $\boldsymbol{H}$ is treated as if it is equal to the following $(n-k) \times (n-\ell)$ matrix:

$$\left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{T} \\ \hline \boldsymbol{I}_{n-k-\ell} & \boldsymbol{B} \end{array}\right).$$

It the condition (2.5) holds the algorithm stops otherwise the algorithm continues with other random sets U and L. The probability P_L that this event occurs is given by:

$$P_L = \frac{\binom{n-t}{\ell}\binom{t}{p}\binom{n-\ell-t}{k-p}}{\binom{n}{k}} = \frac{\binom{n-k}{\ell}\binom{k}{p}\binom{n-k-\ell}{t-p}}{\binom{n}{t}}$$

Stern's approach. Stern proposed in [Ste88] to use "Birthday Paradox" techniques in Leon's approach in order to enhance the probability of getting a good candidate for L. After having chosen L, it chooses randomly two disjoint subsets U_1 and U_2 of $\{n-k+1, \ldots, n\}$ of the same size $k/2$. Hence, these two sets form a partition. The algorithm builds then two lists $\mathcal{L}_1$ and $\mathcal{L}_2$ obtained by considering all the subsets $P_1 \subset U_1$ and $P_2 \subset U_2$ of cardinal $p/2$:

$$\mathcal{L}_1 \stackrel{\text{def}}{=} \left\{ \boldsymbol{s}_L + \sum_{j \in P_1} \boldsymbol{T}_j : P_1 \subset U_1 \right\} \quad \text{and} \quad \mathcal{L}_2 \stackrel{\text{def}}{=} \left\{ \sum_{j \in P_2} \boldsymbol{T}_j : P_2 \subset U_2 \right\}.$$

For each collision between $\mathcal{L}_1$ and $\mathcal{L}_2$ obtained with $P_1^\star \subset U_1$ and $P_2^\star \subset U_2$, that is to say $\boldsymbol{s}_L + \sum_{j\in P_1} \boldsymbol{T}_j = \sum_{j\in P_2} \boldsymbol{T}_j$, the algorithm checks whether the following condition (2.9) holds:

$$\mathsf{wt}\left(\boldsymbol{s}_R + \sum_{j\in P_1^\star \cup P_2^\star} \boldsymbol{B}_j\right) \leqslant t-p. \tag{2.6}$$

The algorithm halts if it is the case. If none of the collisions in the lists give a valid solution, the algorithm chooses other sets U and L, perform another Gaussian elimination and build two new lists with other randomly built sets U_1 and U_2. The success probability P_t of this method is then:

$$P_t = \frac{\binom{t}{p/2}\binom{n-t}{k/2-p/2}}{\binom{n}{k/2}} \frac{\binom{t-p/2}{p/2}\binom{n-t-(k/2-p/2)}{k/2-p/2}}{\binom{n-k/2}{k/2}} \frac{\binom{n-t-(k-p)}{\ell}}{\binom{n-k}{\ell}}.$$

Furthermore, the average number of operations N_t when the sets U (U_1 and U_2) and L are fixed is:

$$N_t = (n-k)^3/2 + k(n-k)^2 + p\ell\binom{k/2}{p/2} + p(n-k-\ell)\frac{\binom{k/2}{p/2}^2}{2^\ell}. \tag{2.7}$$

The value of N_t comprises the cost of the Gaussian elimination, the construction of one list and the cost of comparing words when collisions occur. Th memory complexity is given the size of one list which is roughly $\ell\binom{k/2}{p/2}$. Stern's approach improves time complexity of decoding at the cost of increasing the memory complexity. Therefore, it requires to elaborate a time-memory trade-off, which is controlled by the parameters p and ℓ.

Canteaut-Chabaud's approach. The method of [CC98] improves upon Stern's approach by reducing the cost of the Gaussian elimination. Firstly, the algorithm begins with a fully systematic parity-check matrix so that $\boldsymbol{X} = \boldsymbol{I}_\ell$ and $\boldsymbol{Y} = \boldsymbol{0}$. Next, it introduces a new small parameter $c \geqslant 1$. The general principle is to keep unchanged at least $(n-k-c)$ columns of $\boldsymbol{I}_{n-k}$ obtained from a previous Gaussian elimination instead of taking at random $n-k$ new columns and then perform another costly Gaussian elimination procedure. Of course, these new c columns are obtained from c randomly picked columns of $\boldsymbol{A}$.

Dumer's approach. This method [Dum91, Dum96] relies on a generalisation of Leon's approach. We keep the same notation as Leon's algorithm. Let us recall that the output of Leon's method is a word $\boldsymbol{e}$ from $\mathbb{F}_2^n$ such that $\boldsymbol{e}_L = \boldsymbol{0}$ and $\mathsf{wt}(\boldsymbol{e}_U) = p$. Dumer's approach directly seeks a set V of cardinal $k+\ell$ such that $\mathsf{wt}(\boldsymbol{e}_V) = p$. Hence, V replaces here the set $U \cup L$. Without loss of generality, we assume that $V = \{n-k-\ell+1, \ldots, n\}$. It first builds a parity-check matrix $\boldsymbol{H}$ with the following form:

$$\boldsymbol{H} = \left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{W} \\ \hline \boldsymbol{I}_{n-k-\ell} & \boldsymbol{Z} \end{array}\right)$$

where $\boldsymbol{W}$ is of size $\ell \times (k+\ell)$ and $\boldsymbol{Z}$ is of size $(n-k-\ell)\times(k+\ell)$. Equivalently, up to a permutation of columns, it corresponds to set $\boldsymbol{W} \stackrel{\text{def}}{=} \left(\begin{array}{cc} \boldsymbol{X} & \boldsymbol{T} \end{array}\right)$ and $\boldsymbol{Z} \stackrel{\text{def}}{=} \left(\begin{array}{cc} \boldsymbol{Y} & \boldsymbol{B} \end{array}\right)$ in Leon's approach. The goal now is to find a set $P \subset \{n-k-\ell+1, \ldots, n\}$ of cardinality p such that $\boldsymbol{s}_L + \sum_{j\in P} \boldsymbol{W}_j = \boldsymbol{0}$, and then it checks if the following inequality holds:

$$\mathsf{wt}\left(\boldsymbol{s}_R + \sum_{j\in P} \boldsymbol{Z}_j\right) \leqslant t-p. \tag{2.8}$$

The way for obtaining a set P is the same as Stern's one. It randomly picks two disjoint subsets V_1 and V_2 of $\{n-k-\ell+1, \ldots, n\}$ with the same cardinality $(k+\ell)/2$. The rest of the algorithm is exactly the same as Stern's algorithm. The algorithm builds $\mathcal{L}_1$ and $\mathcal{L}_2$ by considering all the subsets $P_1 \subset V_1$ and $P_2 \subset V_2$ of cardinal $p/2$:

$$\mathcal{L}_1 \stackrel{\text{def}}{=} \left\{\boldsymbol{s}_L + \sum_{j\in P_1} \boldsymbol{W}_j : P_1 \subset V_1\right\} \quad \text{and} \quad \mathcal{L}_2 \stackrel{\text{def}}{=} \left\{\sum_{j\in P_2} \boldsymbol{W}_j : P_2 \subset V_2\right\}.$$

For each collision between $\mathcal{L}_1$ and $\mathcal{L}_2$ obtained with $P_1^\star \subset V_1$ and $P_2^\star \subset V_2$, the algorithm tests if:

$$\mathsf{wt}\left(\boldsymbol{s}_R + \sum_{j\in P_1^\star \cup P_2^\star} \boldsymbol{Z}_j\right) \leqslant t-p. \tag{2.9}$$

It halts when it is the case. The success probability denoted by P_D is given by:

$$P_D = \frac{\binom{t}{p/2}\binom{n-t}{(k+\ell)/2-p/2}}{\binom{n}{(k+\ell)/2}} \frac{\binom{t-p/2}{p/2}\binom{n-t-(k+\ell)/2+p/2}{(k+\ell)/2-p/2}}{\binom{n-(k+\ell)/2}{(k+\ell)/2}}.$$

2.3.2 Conclusion

We have seen that the problem of decoding a random linear code is NP-hard. On the other hand, all the existing decoding algorithms have exponential complexity. Although these results substantiate the fact that minimum distance decoding is a hard problem even if one allows an infinite preprocessing they do not provide enough information about its *real* difficulty. The input of the problem deals with a so wide range of instances that some of them, the *worst cases*, happen to be very hard to solve. But we know in fact nothing about the others and, more importantly, we have no information about the proportion of hard instances among all the possible ones. A more relevant framework for assessing accurately the difficulty of real world problems is the theory of average-case complexity [Lev86]. Unfortunately, the theory of error-correcting codes currently lacks arguments of that kind and it is still an open problem to show that minimum distance decoding of linear codes is an NP-Complete problem on the *average*.

2.4 Code Equivalence Problem

The classification of codes represents an important body of coding theory. Its aims at identifying codes that can be viewed as similar objects when we focus on certain features. Hence, it requires the notion of equivalence. It is well-known that the symmetric group $\mathfrak{S}_n$ acts on $\mathbb{F}_q^n$ through the group action $\boldsymbol{v}^\sigma \stackrel{\text{def}}{=} (v_{\sigma^{-1}(1)}, \ldots, v_{\sigma^{-1}(n)})$ defined for any σ in $\mathfrak{S}_n$ and $\boldsymbol{v}$ in $\mathbb{F}_q^n$. Now, let us assume that we have two linear codes $\mathscr{A}$ and $\mathscr{B}$ over $\mathbb{F}_q$ both of dimension k and length n. We say that $\mathscr{A}$ is *equivalent* to $\mathscr{B}$ if there exists σ in $\mathfrak{S}_n$ such that $\mathscr{A} = \mathscr{B}^\sigma$ with $\mathscr{B}^\sigma \stackrel{\text{def}}{=} \left\{ \boldsymbol{b}^\sigma : \boldsymbol{b} \in \mathscr{B} \right\}$.

The code equivalence is an important tool for classifying codes, especially according to the minimum distance. Indeed, it is obvious that equivalent codes have the same minimum distance and covering radius. Hence in terms of decoding performance, they can be viewed as identical objects. In cryptography, this notion as we will see is important and requires to assess the complexity of solving it. It is straightforward to define a decision problem related it.

Definition 9 (Code equivalence problem)**.** *Given two $k \times n$ generator matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ with $k \leqslant n$. Are there a $k \times k$ matrix $\boldsymbol{S}$ and an $n \times n$ permutation matrix $\boldsymbol{P}$ such that $\boldsymbol{B} = \boldsymbol{SAP}$?*

The complexity of solving this problem is studied in [PR97] which shows that if the code equivalence problem is NP-Complete then the polynomial time hierarchy collapses. On the other hand, the code equivalence problem is not easy either because it also proved in [PR97] that the *Graph Isomorphism* problem reduces to it. These facts may substantiate the fact that finding the permutation between two equivalent codes may be easy in some cases. This was studied in [Sen00] which proposes an algorithm called the *Support Splitting Algorithm* whose *heuristic* time complexity is:

$$\mathcal{O}\left(n^3 + 2^h n^2 \log n\right)$$

where h is the dimension of the *hull*. The hull of a code $\mathscr{A}$ is defined by $\mathscr{A} \cap \mathscr{A}^\perp$. In the case of random linear codes, the hull is most of the time trivial, or is of very small dimension.

Chapter 3

Algebraic Coding

Classical Generalised Reed-Solomon Codes.

Definition 10 (Generalised Reed-Solomon codes)**.** *Let k and n be integers such that $1 \leqslant k < n \leqslant q$ where q is a power of a prime number. Let $\boldsymbol{x} = (x_1, \ldots, x_n)$ where the x_j are distinct elements of $\mathbb{F}_q$ and let $\boldsymbol{y} = (y_1, \ldots, y_n)$ where the y_j are nonzero elements of $\mathbb{F}_q$. The* Generalized Reed-Solomon *code consists of all vectors of the form:*

$$\Big(P(x_1), \ldots, P(x_1)\Big)$$

where $P(z)$ ranges over all polynomials of degree $\leqslant k-1$ with coefficients from $\mathbb{F}_q$.

Generalised Reed-Solomon (GRS) codes represent an important class of Maximum Distance Separable (MDS) codes. Indeed, it is well-known that minimum distance is equal to $n-k+1$. Furthermore, we can prove that the dual of a GRS code is also a GRS code. This property will be useful to give an other way of defining GRS codes. For that purpose, we will require the following definition.

Definition 11. *Let n be an integer such that $n \leqslant q$. For any integer $r \geqslant 1$ and for any $\boldsymbol{x}$ and $\boldsymbol{y}$ from $\mathbb{F}_q^n$, we denote by $\boldsymbol{V}_r(\boldsymbol{x}, \boldsymbol{y})$ the following $r \times n$ matrix*

$$\boldsymbol{V}_r(\boldsymbol{x}, \boldsymbol{y}) \stackrel{def}{=} \begin{pmatrix} y_1 & \cdots & y_n \\ y_1 x_1 & \cdots & y_n x_n \\ \vdots & & \vdots \\ y_1 x_1^{r-1} & \cdots & y_n x_n^{r-1} \end{pmatrix}. \tag{3.1}$$

Proposition 6. *A code $\mathscr{G}$ is a Generalised Reed-Solomon code over $\mathbb{F}_q$ of length $n \leqslant q$ and dimension if an only if there exist $\boldsymbol{x} = (x_1, \ldots, x_n)$ where the x_j's are distinct elements of $\mathbb{F}_q$ and $\boldsymbol{y} = (y_1, \ldots, y_n)$ where the y_j's are nonzero elements of $\mathbb{F}_q$ such that $\boldsymbol{V}_r(\boldsymbol{x}, \boldsymbol{y})$ is a parity-check matrix of $\mathscr{G}$ with $r \stackrel{def}{=} n-k$.*

The real interest of GRS codes is the existence of polynomial-time decoding algorithms that correct t errors as long as $2t \leqslant n-k$. One of them is the Berlekamp-Massey algorithm which runs in time $\mathcal{O}(n^2)$, and the other is the Berlekamp-Welsh algorithm which runs in $\mathcal{O}(n^3)$. This is an important feature but they suffer from a fundamental disadvantage. The field $\mathbb{F}_q$ has to have at least n elements. As n grows, this leads to particularly less and less inefficient encoding and decoding algorithms. A way to overcome this limitation is to consider the subcode obtained with the codewords for which each entry belongs to a subfield of $\mathbb{F}_q$. These codes are called either *subfield subcodes* or *alternant* codes.

Definition 12 (Alternant code)**.** *Let m and n be integers such that $m \geqslant 1$ and let $n \leqslant q^m$. The* alternant *code $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ of order r and length n over $\mathbb{F}_q$ associated to the n-tuple $\boldsymbol{x}$ of distinct elements from $\mathbb{F}_{q^m}$ and the n-tuple $\boldsymbol{y}$ of nonzero elements from $\mathbb{F}_{q^m}$ is:*

$$\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y}) = \Big\{ \boldsymbol{c} \in \mathbb{F}_q^n : \boldsymbol{V}_r(\boldsymbol{x}, \boldsymbol{y}) \boldsymbol{c}^T = \boldsymbol{0} \Big\}. \tag{3.2}$$

It is clear that an alternant code is also the linear space $\mathscr{S} \cap \mathbb{F}_q^n$ where $\mathscr{S}$ is the GRS code over $\mathbb{F}_{q^m}$ whose parity-check matrix is $\boldsymbol{V}_r(\boldsymbol{x}, \boldsymbol{y})$. We also have the following lower-bounds on the dimension and the minimum distance of an alternant code.

Proposition 7. *The dimension k and the minimum distance d of an alternant code $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ satisfy $k \geqslant n - rm$ and $d \geqslant r$.*

As an alternant code $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ of order r is a subset of a GRS code, it inherits the same decoding algorithm. It is therefore able to decode $\frac{r}{2}$ as long as $\boldsymbol{x}$ and $\boldsymbol{y}$ are known. Eventually, we recall on another very famous alternant code called Goppa code.

Definition 13 (Goppa codes)**.** *The Goppa code $\mathscr{G}(\boldsymbol{x}, \gamma)$ over $\mathbb{F}_q$ associated to a polynomial $\gamma(x)$ of degree r over $\mathbb{F}_{q^m}$ and an n-tuple $\boldsymbol{x}$ of distinct elements of $\mathbb{F}_{q^m}$ satisfying $\gamma(x_i) \neq 0$ with $1 \leqslant i \leqslant n$ is the alternant code $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ of order r where $y_i \stackrel{def}{=} \gamma(x_i)^{-1}$.*

Among all the possible Goppa codes, the collection of binary Goppa code ($q = 2$) defined by means of a polynomial $\gamma(z)$ without multiple roots represents without any doubt the most important alternant codes. The reason is the fact that the minimum distance is twice as large as the minimum distance of an alternant code.

Proposition 8. *Let $\gamma(z)$ be a polynomial with coefficients from $\mathbb{F}_{2^m}$ without multiple roots. Let $\boldsymbol{x}$ be an n-tuple of elements from $\mathbb{F}_{2^m}$ such that $\gamma(x_i) \neq 0$. Let us denote $\gamma(z)^2$ by $\gamma^2(z)$. We then have $\mathscr{G}(\boldsymbol{x}, \gamma) = \mathscr{G}(\boldsymbol{x}, \gamma^2)$ and $\mathscr{G}(\boldsymbol{x}, \gamma)$ comes up in* [Pat75] *with a decoding algorithm that corrects t errors in $\mathcal{O}(t^2 m^2)$ operations.*

Projective Generalised Reed-Solomon Codes.

It will be more convenient sometimes to work with the projective plane $\overline{\mathbb{F}}_{q^m} \stackrel{\text{def}}{=} \mathbb{F}_{q^m} \cup \{\infty\}$ and to consider the class of *projective* GRS codes which are slightly more general. A projective alternant code has a parity-check matrix of the form $\boldsymbol{V}_r(\boldsymbol{x}, \boldsymbol{y})$ where $\boldsymbol{x}$ is an n-tuple of distinct elements from $\overline{\mathbb{F}}_{q^m}$ and $\boldsymbol{y}$ is an n-tuple of nonzero elements from $\mathbb{F}_{q^m}$. When $x_i = \infty$, the i-th column of $\boldsymbol{V}_r(\boldsymbol{x}, \boldsymbol{y})$ is equal to $(0, \ldots, 0, y_i)^T$. We can also define a projective alternant and Goppa code thanks to the convention $\gamma(\infty) \stackrel{\text{def}}{=} \gamma_r$ for $\gamma(z) = \sum_{i=0}^{r} \gamma_i z^i$.

Chapter 4

McEliece Cryptosystem

An important tool in public-key cryptography is the notion of *one-way* function. Loosely speaking, a function $f : X \to Y$ is one-way if it is easy to compute $f(x)$ for any x in X but it is hard for (almost) all y in Y to find x in X such that $y = f(x)$. Another important concept is the one of *trapdoor*. It represents a quantity that facilitates the inversion of a function. Anyone who knows it is able to compute the pre-image of any element in the codomain. When a function is both one-way and has a trapdoor then it is called a *trapdoor one way-function*. It forms the basis for getting a public-key cryptosystem.

Formally, a public-key cryptosystem should provide three algorithms: a *key generation* algorithm, an *encryption* algorithm E and a *decryption* algorithm D. Given a (security) parameter $\kappa \geq 0$, the key generation algorithm is a probabilistic polynomial-time in κ algorithm. It outputs a pair (pk, sk) of public/private key. It also specifies a finite *set* X of the *plaintexts*. The algorithm E is a (probabilistic) polynomial-time algorithm parameterized by pk such that on input $\boldsymbol{x}$ in X, it outputs $\boldsymbol{c} \stackrel{\text{def}}{=} E_{pk}(\boldsymbol{x})$ also called the *ciphertext* of x. The decryption algorithm D is a deterministic polynomial-time algorithm parameterized by sk such that on input $\boldsymbol{c}$ outputs $\boldsymbol{x} \stackrel{\text{def}}{=} D_{sk}(\boldsymbol{c})$. The cryptosystem should satisfy the *correctness* property which imposes that the decryption must undo the encryption:

$$\forall \boldsymbol{x} \in X,\ D_{sk}\Big(E_{pk}(\boldsymbol{x})\Big) = \boldsymbol{x}.$$

The general problem of decoding random linear codes is a potential candidate for building public-key cryptographic primitives such as an encryption scheme. McEliece in [McE78] was the first the use this problem to propose a public-key cryptosystem. The general idea is to start from a family of codes equipped with a polynomial-time decoding algorithm. The fundamental concept of this proposal is to consider two equivalent representations of a code: one should facilitate the decoding, whereas from the other one, the decoding should be impossible. Although his design principle is general, he explicitly advocated to use binary Goppa codes [Gop70].

4.1 Description

One of the main cryptographic primitives in code-based public-key cryptography is the McEliece encryption [McE78]. We recall that a linear *code* over a finite field $\mathbb{F}_q$ of q elements defined by a $k \times n$ matrix $\boldsymbol{G}$ (with $k \leq n$) over $\mathbb{F}_q$ is the vector space $\mathscr{C}$ spanned by its rows. $\boldsymbol{G}$ is chosen as a full-rank matrix, so that the code is of dimension k. The *rate* of the code is given by the ratio k/n. Code-based public-key cryptography focuses on linear codes that have a polynomial time decoding algorithm. The role of decoding algorithms is to correct errors of prescribed weight. We say that a decoding algorithm corrects t errors if it recovers $\boldsymbol{u}$ from the knowledge of $\boldsymbol{uG} + \boldsymbol{e}$ for all possible $\boldsymbol{e} \in \mathbb{F}_q^n$ of weight t.

We now define the McEliece cryptosystem as it was given in [McE78].

Key generation.

Secret Key. The triplet $(\boldsymbol{S}, \boldsymbol{G}_s, \boldsymbol{P})$ of matrices defined over a finite field $\mathbb{F}_q$ over $q = 2^s$ elements. $\boldsymbol{G}_s$ is a full rank matrix of size $k \times n$, with $k < n$, $\boldsymbol{S}$ is of size $k \times k$ and is invertible. $\boldsymbol{P}$ is a permutation matrix of size $n \times n$. $\boldsymbol{G}_s$ is chosen such that its associated linear code has a polynomial-time decoding algorithm which corrects r errors.

Public key. The $k \times n$ matrix $\boldsymbol{G} \stackrel{\text{def}}{=} \boldsymbol{SG}_s\boldsymbol{P}$.

Encryption. A plaintext $\boldsymbol{u} \in \mathbb{F}_q^k$ is encrypted by choosing a random vector $\boldsymbol{e}$ in $\mathbb{F}_q^n$ of weight r. The corresponding ciphertext is $\boldsymbol{c} = \boldsymbol{uG} + \boldsymbol{e}$.

Decryption. $\boldsymbol{c}' \stackrel{\text{def}}{=} \boldsymbol{c}\boldsymbol{P}^{-1}$ is first computed from the ciphertext $\boldsymbol{c}$. Let us notice that we have:

$$\boldsymbol{c}' = (\boldsymbol{u}\boldsymbol{S}\boldsymbol{G}_s\boldsymbol{P} + \boldsymbol{e})\boldsymbol{P}^{-1} = \boldsymbol{u}\boldsymbol{S}\boldsymbol{G}_s + \boldsymbol{e}\boldsymbol{P}^{-1}.$$

Since the weight of $\boldsymbol{e}\boldsymbol{P}^{-1}$ is r, the decoding algorithm recovers in polynomial time $\boldsymbol{u}\boldsymbol{S}$ and thus, the plaintext $\boldsymbol{u}$ by multiplication by $\boldsymbol{S}^{-1}$.

What is referred to as the McEliece cryptosystem is this scheme with the particular choice of the binary Goppa codes defined by monic irreducible polynomials.

4.2 Best Known Attacks

The minimum requirement for an encryption function E_{pk} is that it should be infeasible from a given ciphertext $\boldsymbol{c}$ and public data[1] like the public key pk, ciphertexts, *etc.* to recover the corresponding plaintext $\boldsymbol{x}$. The one-wayness of the McEliece encryption function is directly linked to the following computational problem.

Definition 14 (McEliece Problem)**.** *Let $\boldsymbol{G}$ be a generator matrix of a binary Goppa code of length $n \leqslant 2^m$ and dimension $k = n - tm$ where m and t are positive integers. Let $\boldsymbol{x}$ be a vector from $(\mathbb{F}_{2^m})^k$ and let $\boldsymbol{e}$ be a vector from $(\mathbb{F}_{2^m})^n$ of weight t and let us set $\boldsymbol{c} \stackrel{def}{=} \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e}$.*

The McEliece Problem *consists in finding $\boldsymbol{x}$ and $\boldsymbol{e}$ only from $\boldsymbol{G}$ and $\boldsymbol{c}$,*

One possible way of solving this problem consists in devising a method that recovers the private key sk. This existence of such a method is devastating because the encryption scheme becomes useless. This kind of attacks are called *key-recovery* attacks.

But, it is also possible to recover a plaintext from a specific ciphertext without resorting to a key-recovery attack. In particular, an attacker against the McEliece scheme would find the plaintext by applying general decoding methods on the public matrix $\boldsymbol{G}$. This category of attacks is called *decoding attacks*.

Key-recovery attacks. From the previous description of the McEliece cryptosystem, one would say that the private key is the triple $(\boldsymbol{S}, \boldsymbol{G}_s, \boldsymbol{P})$. But, one may ask why an attacker who knows these three matrices would be able to decode any ciphertext. Actually, the knowledge of $\boldsymbol{G}_s$ is useless for decoding because all the existing decoding algorithms requires to know in reality the polynomial $\gamma(z)$ and the support $\boldsymbol{x}$. So they are the real secret quantities. The role of $\boldsymbol{P}$ is to increase the number of possible choices of the support $\boldsymbol{x}$. As for the matrix $\boldsymbol{S}$, its role is fuzzy and is linked to the manner of creating the generator matrix $\boldsymbol{G}_s$. It is clear that if a Goppa code and more generally an alternant codes is given under the form of a (rectangular) Vandermonde matrix, a decoder can be easily devised. Hence, the public matrix should not reveal this particular matrix and a way for hiding it is to take another basis of the code. But, an attacker has always the possibility to transform the public matrix $\boldsymbol{G}$ into a systematic one by a Gaussian elimination and this matrix is *unique* up to a permutation. *Consequently, one may view $\boldsymbol{S}$ as the matrix that sends any generator matrix to the one in systematic form.* Hence, without loss of generality, we will always assume that $\boldsymbol{G}$ is systematic. We will also assume that it describes here a binary Goppa code of length $n = 2^m$ obtained from a monic irreducible polynomial $\gamma(z)$ of degree r and coefficients from $\mathbb{F}_{2^m}$. We seek a way to recover $\boldsymbol{x}$ and $\gamma(z)$.

Currently, the best known key-recovery attack relies essentially on an exhaustive search. The idea consists in fixing a support $\boldsymbol{a} = (a_1, \ldots, a_n)$ from $\mathbb{F}_{2^m}^n$ where the a_j are different and enumerating monic polynomials $s(z)$ of degree r with coefficients in $\mathbb{F}_{2^m}^n$ until a code equivalent to the one defined by $\boldsymbol{G}$ is found. This last test is done through SSA algorithm [Sen00]. Empirically, the complexity of this algorithm applied to binary Goppa codes is about $\mathcal{O}(n^3)$ because, with high probability, the hull is trivial. Using the assumption $n = 2^m$, we see that the overall time complexity is therefore about $\mathcal{O}(n^{t+3})$.

This approach can be slightly improved in theory. One can reduce the search space by identifying polynomials that lead to equivalent Goppa codes. The first work to use this principle is [Gib91]. It proposes to define a equivalence relation between polynomials by means of the action of two types of permutations of $\mathbb{F}_{2^m}$, namely the affine permutations $\tau_{a,b}(z) = az + b$ with a from $\mathbb{F}_{2^m} \backslash \{0\}$ and b from $\mathbb{F}_{2^m}$, and the Frobenius automorphisms $z \longmapsto z^{2^\ell}$ with $0 \leqslant \ell \leqslant m-1$. On can easily check that the Goppa codes defined by the polynomials $\gamma(z)$ and $\gamma(az+b)$ are equivalent. Furthermore, if we define the polynomial $\gamma^{2^\ell}(z)$ as:

$$\gamma^{2^\ell}(z) \stackrel{\text{def}}{=} \sum_{i=0}^{r} \gamma_i^{2^\ell} z^i$$

then the Goppa codes defined by $\gamma^{2^\ell}(z)$ and $\gamma(z)$ are also equivalent. More exactly, we have:

$$\mathscr{G}(\boldsymbol{x}, \gamma) = \mathscr{G}(\boldsymbol{x}^{2^\ell}, \gamma^{2^\ell})$$

where $\boldsymbol{x}^{2^\ell} \stackrel{\text{def}}{=} (x_1^{2^\ell}, \ldots, x_n^{2^\ell})$. By composing these two types of transformations, one would expect that about $m(n-1)n$ polynomials would give equivalent Goppa codes.

[1] This kind of attack is called a *Chosen Plaintext Attack* (CPA).

This can be improved by considering extended Goppa codes. We use the property that codes are equivalent if and only if the corresponding extended codes are equivalent. We denote by $\tilde{\mathscr{G}}_r(\boldsymbol{x},\gamma)$ the extended code of $\mathscr{G}_r(\boldsymbol{x},\gamma)$. It requires to introduce a new symbol denoted by ∞ that is added to $\mathbb{F}_{2^m}$. This symbol enables to define projective Goppa codes by means of the projective plane $\overline{\mathbb{F}}_{2^m} \stackrel{\text{def}}{=} \mathbb{F}_{2^m} \cup \{\infty\}$.

We will also focus on special kind of permutations called *fractional transformations*. Let a, b, c, d be elements from $\mathbb{F}_{2^m}$ such that $ad - bc \neq 0$ and let us define $\psi : \overline{\mathbb{F}}_{2^m} \longrightarrow \overline{\mathbb{F}}_{2^m}$ such that $\psi \stackrel{\text{def}}{=} \dfrac{az+b}{cz+d}$. The usual rules are used to evaluate $\psi(z)$ namely $\psi(\infty) = \frac{a}{c}$ and $\psi(-\frac{d}{c}) = \infty$. Then if we use the notation $\boldsymbol{x}^\psi \stackrel{\text{def}}{=} (\psi(x_1),\ldots,\psi(x_n))$ for any vector $\boldsymbol{x}$ from $(\overline{\mathbb{F}}_{2^m})^n$ and by defining $\gamma^\psi(z)$ as being $(cz+d)^r\gamma\big(\psi(z)\big)$, or equivalently,

$$\gamma^\psi(z) = \sum_{i=0}^{r} \gamma_i (az+b)^i (cz+d)^{r-i},$$

we can prove that:

$$\tilde{\mathscr{G}}_r(\boldsymbol{x},\gamma^\psi) = \tilde{\mathscr{G}}_r(\boldsymbol{x}^\psi,\gamma).$$

Again, by composing ψ with any Frobenius automorphism, we obtain about $m(n^2-1)(n^2-n)$ equivalent extended Goppa codes. Using the fact that there are less that n^r/r monic irreducible polynomials of degree r, we see that the exhaustive search can be theoretically reduced to:

$$\frac{n^r}{r} \times \frac{1}{m(n^2-1)(n^2-n)} \times n^3 \simeq \frac{1}{mr} n^{r-1}.$$

In reality, the precise cost of this attack is related to the number of inequivalent Goppa codes. So the classification of Goppa codes is paramount to assess exactly the security of [McE78]. This matter is still an open problem.

Decoding attacks. We know that there exist many general-purpose decoding algorithms in the literature [McE78, LB88, Leo88, Ste88, vT90, CC94, CC95, Dum96, CC98, CS98, BLP08]. All these algorithms strive to find an error-free information set. We know that the complexity can be upper-bounded by $k^\omega \frac{\binom{n}{r}}{\binom{n-k}{r}}$. Clearly, this complexity is always better than that of the best known key-recovery attack. Consequently, the parameters of a McEliece cryptosystem are then generated to resist against decoding attacks.

Reaction attack. We have said that the one-wayness is a necessary criterion for a secure encryption scheme. However it is useless if we do not define exactly what kind of security is desired, or more precisely, what kind of threats against which we want to be protected. Especially, one has to explicitly tell what is the amount of resource that an attacker has at his disposal. For example, we can imagine a scenario [HGS99] where an attacker sends ciphertexts and waits for the receiver's answers. These latter do not need to be the corresponding plaintexts but can just be yes or fail[2] for instance. This type of attack can be implemented against the McEliece encryption. It is however assumed that the decoding algorithm is unable to correct more than r errors. It starts from an intercepted ciphertext and then flips one bit to the other value and sends this modified ciphertext to the receiver. Hence, if the flipped bit corresponds to a position where the initial error vector of weight r is not zero, then its weight becomes $r+1$. Otherwise, its weight decreases to $r-1$. These two different situations will lead to two different answers from the receiver. In one case, he will be able to say yes when the weight is $r-1$. In the other case, his answer will be fail because the weight is beyond its decoding capacity. This attack is also called *reaction-attack* and can be regarded as a weaker version of a *Chosen Ciphertext Attack* (CCA). In this model, an attacker can have the corresponding plaintexts of the ciphertexts of his choice.

Resent-message attack. The paper [Ber97] (See also [VDv02]) treats the situation where plaintext $\boldsymbol{x}$ is encrypted twice by means to the McEliece encryption function. Let us say that the ciphertexts are $\boldsymbol{c}_1 = \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e}_1$ and $\boldsymbol{c}_2 = \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e}_2$ where $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are both of weight r. An easy way to detect resent message is to compute the weight of $\boldsymbol{z} \stackrel{\text{def}}{=} \boldsymbol{c}_1 + \boldsymbol{c}_2$. Clearly, the weight is less than or equal to $2r$ when the same message has been sent because the sum of two ciphertexts corresponding to different plaintexts would be of weight about $n/2$. The next point to observe is a zero position in $\boldsymbol{z}$ comes with high probability from zero positions from $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. This is because the probability p_1 that the same position in $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ equal 1 is $\left(\frac{r}{n}\right)^2$ whereas the probability p_0 that the same position is 0 is $\left(1-\frac{r}{n}\right)^2$. As $p_0 \gg p_1$ an adversary can seek k information bits among the zero positions of $\boldsymbol{z}$ because they are certainly get from positions where $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are both zero. Actually, the number of zeros obtained from two common ones between $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ is $a \stackrel{\text{def}}{=} \frac{1}{2}\left(2r - \mathsf{wt}(\boldsymbol{z})\right)$. In other words, these a positions will be misinterpreted as error-free positions by the adversary. The probability $\mathbf{prob}\left(\mathsf{wt}(\boldsymbol{e}_1 \cap \boldsymbol{e}_2) = a\right)$ that $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ share a positions is:

$$\mathbf{prob}\left(\mathsf{wt}(\boldsymbol{e}_1 \cap \boldsymbol{e}_2) = a\right) = \frac{\binom{r}{a}\binom{n-r}{r-a}}{\binom{n}{r}}.$$

[2] Even no response from the receiver can be interpreted as a fail.

The average value of a is given by:

$$\mathbb{E}\{a\} = \sum_{a=0}^{r} a\,\mathbf{prob}\left(\mathsf{wt}(\boldsymbol{e}_1 \cap \boldsymbol{e}_2) = a\right) = \frac{1}{\binom{n}{r}} \sum_{a=0}^{r} a \binom{r}{a}\binom{n-r}{r-a} = \frac{r^2}{n}.$$

The value of $\mathbb{E}\{a\}$ when computed with cryptographic parameters, is very small. The strategy a attacker can adopt is to randomly choose k among the $n - \mathsf{wt}(\boldsymbol{z})$ zero of $\boldsymbol{z}$ and he flips ℓ positions with $0 \leqslant \ell \leqslant \mathbb{E}\{a\}$ until he founds k positions without errors. The running time of this attack is about:

$$\binom{n - \mathbb{E}\{\mathsf{wt}(\boldsymbol{z})\}}{k}\binom{k}{\mathbb{E}\{a\}} = \binom{n - 2r + 2\frac{r^2}{n}}{k}\binom{k}{\frac{r^2}{n}}$$

since $\mathbb{E}\{\mathsf{wt}(\boldsymbol{z})\} = 2r - 2\mathbb{E}\{a\}$. Again with concrete parameters, the time complexity is relatively small.

The *resent-message* attack and the *reaction attack* show that the one-wayness property is a necessary condition but it does not cover situation where an adversary has more than public data at his disposal. The adequate security notion that should verify any secure encryption function is the resistance against *Adaptive Chosen Ciphertext Attacks* (CCA2). In this model, a polynomial-time attacker is allowed to ask an oracle to decrypt any ciphertext of choice. This is done in two steps[3]. The first step can be regarded as a learning phase in which in he collects as much information as he wants. In the second phase the attacker mounts an attack in order to achieve one precise goal. It can be recovering the private key, inverting the encryption function or simply guessing some bits of a given ciphertext, *etc*. The least ambitious goal is when a (polynomial-time) adversary is unable to guess the ciphertext that corresponds to a plaintext, even if he has chosen it in advance. This means that the knowledge of a particular ciphertext does not give any information about the corresponding plaintext. This property is called *indistinguishability* or also *semantic security*. Note that this notion implies to have a probabilistic encryption. In reality, this notion is formally defined by a game involving a challenger and an adversary. First, the adversary chooses by himself two plaintexts $\boldsymbol{m}_0$ and $\boldsymbol{m}_1$ that he submits to the challenger. The challenger picks at random one of the two plaintexts, encrypts it and sends it to the adversary. The latter has to guess which one of the plaintext has been encrypted. An encryption scheme is said *semantically secure* or *indistinguishable* if there does not exist a polynomial-time algorithm that solves the game with a probability better than $\frac{1}{2}$.

We have now all the ingredients to define what is the best preferred encryption scheme. When an encryption scheme is qualified as *Indistinguishable under an adaptative chosen ciphertext attack* or (IND-CCA2), it means that there does not exist a polynomial-time adversary with the most possible resources that achieves the simplest goal.

One may ask whether the McEliece scheme is IND-CCA2. Unfortunately, the answer is no because an adversary is always able to recognize ciphertexts obtained from plaintexts of his choice. Let assume that he has chosen the plaintexts $\boldsymbol{m}_0$ and $\boldsymbol{m}_1$ from $\mathbb{F}_2^k$ and he receives from the challenger the ciphertext $\boldsymbol{c}$ corresponding under the public key $\boldsymbol{G}$ either to $\boldsymbol{m}_0$ or $\boldsymbol{m}_1$. He then computes $\boldsymbol{c} + \boldsymbol{m}_0\boldsymbol{G}$ and $\boldsymbol{c} + \boldsymbol{m}_1\boldsymbol{G}$. Clearly, one of these two vectors has a weight that equals r and the other has a weight strictly greater than r. Indeed, if we assume for instance that $\boldsymbol{c} = \boldsymbol{m}_0\boldsymbol{G} + \boldsymbol{e}$ where $\boldsymbol{e}$ is of weight r. Then $\boldsymbol{c} + \boldsymbol{m}_0 = \boldsymbol{e}_1$ and $\boldsymbol{c} + \boldsymbol{m}_1\boldsymbol{G} = (\boldsymbol{m}_0 + \boldsymbol{m}_1)\boldsymbol{G} + \boldsymbol{e}$. But the weight of $(\boldsymbol{m}_0 + \boldsymbol{m}_1)\boldsymbol{G} + \boldsymbol{e}$ is necessarily greater than r since the minimum distance of the code defined by $\boldsymbol{G}$ is greater than or equal to $2r + 1$.

We see that the encryption of [McE78] cannot be used as such. Indeed, the error vector $\boldsymbol{e}$ and the plaintext $\boldsymbol{m}$ has to be linked as proposed in [VDv02]. Therefore, the scheme has to be modified "à la" OAEP (Optimal Asymmetric Encryption Padding) [BR95] in order to become a semantically secure scheme. We will see in the following how this can be done.

4.3 Replacing Goppa Codes

Following McEliece's pioneering work, several different public key cryptosystems based on the intractability of decoding a linear code have been proposed [Nie86, GPT91, Sid94, JM96, BL04, BL05, BC07, BBC08, BCGO09, MB09]. We have seen that McEliece's proposal relies on irreducible binary Goppa codes. Except Niederreiter's idea in [Nie86] which brought a significant modification, all these encryption schemes can be seen as a slight variation of [McE78] because they keep the same principle for encryption and decryption but use different family of codes. A *variant* of the McEliece cryptosystem is any public-key cryptosystem that replaces binary Goppa codes by another family of codes. It is clear that McEliece's idea is very generic and hence can be used with any family of codes provided they can be decoded in polynomial time.

There are many reasons for suggesting such a change. Historically, the first one to design a significantly new code-based cryptosystem is Niederreiter [Nie86]. The scheme considers parity-check matrices, and the plaintext messages have to be of weight r. The corresponding ciphertexts are then the syndromes computed thanks to the public parity-check matrix. In terms of security, these two schemes are totally equivalent: any attacker against one can be directly used against the other. Niederreter also prompted to use Generalized Reed-Solomon codes as a possible alternative to binary Goppa codes. This work was then followed by others with the hope to be as secure as McEliece's proposal. Subcodes of Generalized Reed-Solomon codes were also advocated in [BL05]. It was proposed in [Sid94] to use Reed-Muller codes. Algebraic-Geometric codes were suggested in [JM96]. The Gabidulin-Paramonov-Tretjakov (GPT) cryptosystem considered Gabidulin codes devised for the rank-metric. LDPC codes have also been repeatedly suggested in [MRS00, BC07, BBC08] for this use.

[3] In a classical Chosen Ciphertext Attack (CCA) scenario, there is only one step.

Unfortunately, many of these schemes were broken [SS92, Gib95, MS07, OTD08, FM08, Ove08, Wie10, OTD10]. All these attacks result in a total break of the system (the secret key, or an equivalent secret key is recovered from the knowledge of the public key). For instance, Niederreiter's suggestion to use Generalized Reed-Solomon codes turned out to be an insecure solution [SS92]. This also the case with Reed-Muller codes [MS07]. The use of algebraic-Geometric codes becomes less and less secure due to the works of [FM08] which break any cryptosystem with codes on curves of genus $g \leqslant 2$. Recently this result was further extended by [MCMMP11] where any code on curves of genus g and rate R satisfying one of the following conditions as n goes to infinity:

$$\gamma \leqslant R \leqslant \frac{1}{2} - \gamma, \quad \frac{1}{2} + \gamma \leqslant R \leqslant 1 - \gamma, \quad \frac{1}{2} - \gamma \leqslant R \leqslant 1 - 3\gamma, \quad 3\gamma \leqslant R \leqslant \frac{1}{2} + \gamma$$

with $\gamma \stackrel{\text{def}}{=} g/n$ represents an insecure primitive. In particular, the union of these intervals is $[\gamma, 1-\gamma]$ when and only when when $\gamma \leqslant \frac{1}{6}$. Eventually, it is shown in [Wie10] that GRS subcodes represent an insecure choice. As for the GPT cryptosystem, the article [Ove08] proves that it cannot be considered as a trustworthy alternative.

However, these results do not affect the original McEliece cryptosystem. It remains unbroken because of the fact that the best known attacks are exponential. Despite its impressive resistance against a variety of attacks and its fast encryption and decryption, McEliece cryptosystem has not stood up for practical applications. This is most likely due to the large size of the keys Indeed, the public keys are (binary) matrices which can be described at best[4] by $k(n-k) = R(1-R)n^2$ bits with $R \stackrel{\text{def}}{=} k/n$. Currently, it is advocated to use $n \geqslant 2^{10}$ and generally $R \leqslant 3/4$. So the number of bits necessary to store a public key is more than 2^{20} bits.

To overcome this limitation, a new trend initiated in [MRS00] and then followed by [Gab05, BC07, BBC08, BCGO09, MB09, Per11, BLM11] tries to propose cryptosystems with a decreased key size. They all consider codes given by highly structured generator (or parity-check) matrices. The advantage is to have very compact matrices so that they can be described from only few rows. A famous example is the class of cyclic codes. A code $\mathscr{C}$ is *cyclic* if for any $\boldsymbol{c} = (c_1, \ldots, c_n)$ from $\mathscr{C}$, $(c_2, \ldots, c_n, c_1)$ also belongs to $\mathscr{C}$. One can readily see that a cyclic code is stable under any right and left cyclic shift. It is well-known that a cyclic code admits a generator matrix in almost[5] *circulant* form.

Definition 15 (Circulant)**.** *A matrix $\boldsymbol{A}$ of size $n \times n$ is* circulant *if it is of the form:*

$$\boldsymbol{A} = \begin{pmatrix} a_1 & a_2 & \cdots & a_n \\ a_n & a_1 & \cdots & a_{n-1} \\ \vdots & & \ddots & \vdots \\ a_2 & \cdots & \cdots & a_1 \end{pmatrix}.$$

The set of $n \times n$ circulant matrices over $\mathbb{F}_q$ is denoted by $\mathfrak{C}_n(\mathbb{F}_q)$.

Recently, a new kind of codes appeared in [MB09] called *dyadic* codes. These codes are defined for length n of the form $n = 2^\ell$ where $\ell \geqslant 1$. Next, the set of indices is not anymore $\{1, \ldots, n\}$ but $\mathbb{N}_n \stackrel{\text{def}}{=} \{0, \ldots, n-1\}$ and each integer is identified with its binary representation composed of ℓ bits. Finally, for any i and j from $\mathbb{N}_n$, $i \oplus j$ is the integer in $\mathbb{N}_n$ obtained by the bitwise "exclusive-or" from i and j which are of course viewed as sequences of ℓ bits. A code $\mathscr{C}$ is then dyadic if we have for any $\boldsymbol{c} = (c_0, \ldots, c_{n-1})$ and for i in $\mathbb{N}_n$ the following property:

$$\left(c_{(0 \oplus i)}, c_{(1 \oplus i)}, \ldots, c_{((n-1) \oplus i)}\right) \in \mathscr{C}$$

Such codes are also defined by very structured matrices naturally called dyadic matrices.

Remark 1. One sees that $\mathbb{N}_n$ equipped with the $\oplus$ operation is isomorphic to the vector space $(\mathbb{F}_2^\ell, +)$.

Definition 16 (Dyadic)**.** *Let $\boldsymbol{\Delta} = (\Delta_{i,j})$ with $0 \leqslant i \leqslant n-1$ and $0 \leqslant j \leqslant n-1$ be an $n \times n$ matrix. If we denote by $\boldsymbol{d} = (d_0, \ldots, d_{n-1})$ the first row of $\boldsymbol{\Delta}$ then $\boldsymbol{\Delta}$ is* dyadic *if and only if $\Delta_{i,j} = d_{i \oplus j}$. The set of $n \times n$ dyadic matrices over $\mathbb{F}_q$ is denoted by $\mathfrak{D}_n(\mathbb{F}_q)$.*

One realizes immediately the interest of cyclic and dyadic matrices in code-based cryptography: they are completely described thanks to the first row. We will show that these two types of matrices and the corresponding linear codes fit in a wider framework we called Φ-invariant codes.

4.4 Goppa Code Distinguishing Problem

The crucial issue regarding the one-wayness of the McEliece cryptosystem is to have a better idea of the difficulty of the McEliece problem. We have seen that the only known methods that aim to solve it are based either on almost a exhaustive search of the private key or on applying very general decoding methods. Both approaches run in exponential time. This situation is a sense unsatisfactory because there is no certitude that there does not exist a better way to solve it.

[4] when the matrix contains k columns that form the identity matrix.

[5] Strictly speaking a generator matrix has to be of full rank but if we allow linearly dependant rows then a cyclic code admits a circulant generator matrix.

A classical stance in code-based cryptography is to claim that binary Goppa codes look like random linear codes. It amounts to saying that there does not exist a polynomial-time computable quantity which behaves differently depending on whether the code is a Goppa or a random code. Currently, it is an open problem to establish a formal proof that would substantiate the claim that a binary Goppa code is *indistinguishable* from a random code.

This assumption is clearly attractive because it enables to rely on the commonly admitted hardness of decoding a random linear code to say that the McEliece function is hard to invert. This reasoning does make sense also because binary Goppa codes share several common aspects[6] with a randomly picked linear code. Furthermore, all the general decoding algorithms do not exploit the information, even partially, that a matrix describes a Goppa code. Based on this, the authors of [CFS01] defined the *Goppa code distinguishing problem*. Before defining this problem, we introduce some notation. For any n and k integers such that $k \leqslant n$. We denote by $\mathsf{Goppa}(n,k)$ the set of $k \times n$ generator matrices of binary Goppa codes. Similarly, $\mathsf{Random}(n,k)$ is the set of binary $k \times n$ random generator matrices.

Definition 17 (Goppa Code Distinguishing (GD) Problem)**.** *A* distinguisher $\mathcal{D}$ *is an algorithm that takes as input a matrix* $\boldsymbol{G}$ *and returns a bit.* $\mathcal{D}$ *solves the GD problem if it wins the following game:*

- $b \leftarrow \{0,1\}$
- *If* $b = 0$ *then* $\boldsymbol{G} \leftarrow \mathsf{Goppa}(n,k)$ *else* $\boldsymbol{G} \leftarrow \mathsf{Random}(n,k)$
- *If* $\mathcal{D}(\boldsymbol{G}) = b$ *then* $\mathcal{D}$ *wins else* $\mathcal{D}$ *loses.*

Definition 18. *The advantage* $Adv^{GD}(\mathcal{D})$ *of a GD distinguisher* $\mathcal{D}$ *is defined by:*

$$Adv^{GD}(\mathcal{D}) \stackrel{def}{=} \left|\Pr[\mathcal{D}(\boldsymbol{G}) = 1 : \boldsymbol{G} \leftarrow \mathsf{Goppa}(n,k)] - \Pr[\mathcal{D}(\boldsymbol{G}) = 1 : \boldsymbol{G} \leftarrow \mathsf{Random}(n,k)]\right|$$

where $\Pr[\mathcal{D}(\boldsymbol{G}) = 1 : \boldsymbol{G} \leftarrow \mathsf{Goppa}(n,k)]$ *is the probability that* $\mathcal{D}$ *outputs* 1 *when* $\boldsymbol{G}$ *is a random binary generator matrix of a Goppa code, and* $\Pr[\mathcal{D}(\boldsymbol{G}) = 1 : \boldsymbol{G} \leftarrow \mathsf{Random}(n,k)]$ *is the probability that* $\mathcal{D}$ *outputs* 1 *when* $\boldsymbol{G}$ *is a binary random matrix.*

Definition 19. *A function* $\varepsilon(k)$ *is said* negligible *if for any integer* $a > 0$*, there exists an integer* $k_a > 0$ *such that:*

$$\forall k \geqslant k_a, \quad \varepsilon(k) < \frac{1}{k^a}$$

The interest of negligible function is to offer the possibility of keeping the probability that an event occurs negligible even after polynomially many tries. This notion will be fundamental when breaking for instance cryptographic primitives. We are now able to state an important assumption[7].

Assumption 1 ([CFS01])**.** $Adv^{GD}(\mathcal{D})$ *is negligible for any polynomial-time algorithm* $\mathcal{D}$ *that solves the GD problem.*

Note that up to our recent work in [FGUO+11], the only known algorithm that solves the Goppa code Distinguishing problem enumerates binary Goppa codes and tests with SSA algorithm the code equivalence. We have seen that this approach runs in time $\mathcal{O}(\frac{n^{r-1}}{mr})$. Recall finally that for binary Goppa codes, $m \leqslant \log n$ and $r = \frac{1}{m}(1-R)n$.

4.5 Semantically Secure Conversions

The fundamental issue when dealing with cryptographic primitives is to prove its security. Several approaches are possible. The most natural one is to show that the primitive resists to the best known attacks. However, this does not guarantee that there will not appear one day a better attack that renders the primitive insecure. The methodology of *security proof by reduction* appeared to remedy this question by linking a security notion that a cryptographic primitive should verify to an algorithmic problem widely considered as hard. The approach is similar to the one that proves the NP-Completeness of a given problem. Such a "security proof" proves that if an attacker exists then it can be used as a subroutine to solve a hard problem. In other words, such an attacker has little chances to exist.

We have enumerated a list of the existing attacks against [McE78]. We have seen that under the CPA model, the best attack belongs to the family of decoding attacks. But, under the CCA model, the cryptosystems can be broken very easily. These results prompt code-based cryptographer to design conversions that would to an IND-CCA secure encryption scheme. The first article to propose such a conversion for the McEliece cryptosystem is [KI01] which proposes a conversion of [McE78] that is IND-CCA2 in the *Random Oracle Model* under the assumption that the problem of decoding random linear codes is difficult. This works was then followed by [NIKM08] which proposes another modification while providing an IND-CPA secure encryption scheme in the standard model[8] under the assumptions that both decoding random linear codes *and* distinguishing Goppa codes are difficult problems. Finally, it was then improved in [DMQN09] which shows that under the same assumptions, (a modified) McEliece cryptosystem is IND-CCA2 in the standard model.

[6] Like random codes, Goppa codes asymptotically meet the Gilbert-Varshamov bound. They have also a trivial permutation group like random codes.

[7] According to [CFS01], proving or disproving the hardness of the GD problem will have a significant impact: *"Classification issues are in the core of coding theory since its emergence in the 50's. So far nothing significant is known about Goppa codes, more precisely there is no known property invariant by permutation and computable in polynomial time which characterizes Goppa codes. Finding such a property or proving that none exists would be an important breakthrough in coding theory and would also probably seal the fate, for good or ill, of Goppa code-based cryptosystems".*

[8] There is no hash function in this model

Part II

Contributions

Chapter 5

Group Invariant Codes

5.1 Motivation

This chapter is devoted to giving a general framework that encompasses cyclic and dyadic codes. We will see that in fact these matrices can be viewed as the same objects once an appropriate algebra is set. This will permit us to have a better understanding of the encryption schemes based on very structured matrices (circulant and dyadic) from both designer's and cryptanalyst's perspectives.

Firstly, we have seen that all the cryptosystems proposed in [Gab05, BC07, BBC08, BCGO09, MB09, Per11, BLM11] use codes that are stable under the action of a known and easily computable set of mappings, or more exactly a set of permutations. The common point of all these permutations is to belong to a certain group that globally stabilize the codes. This aspect can be formalized more precisely. Let us consider a code $\mathscr{C}$ of length n and dimension k over $\mathbb{F}_q$. We denote by $\mathfrak{S}_n$ the symmetric group of order n and let $\mathbb{N}$ be the set of non-negative integers. Let us assume that there exists a set $\Phi = \{\phi_i : i \in I\}$ where $I \subset \mathbb{N}$ and ϕ_i is from $\mathfrak{S}_n$. We also assume that $\mathscr{C}$ is stable under the action of Φ:

$$\forall \phi \in \Phi, \forall \boldsymbol{c} \in \mathscr{C}, \;\; \boldsymbol{c}^{\phi} \in \mathscr{C}$$

with $\boldsymbol{c}^{\phi}$ meaning $(c_{\phi^{-1}(1)}, \ldots, c_{\phi^{-1}(n)})$. We give now a definition that will be useful for the sequel.

Definition 20. *A code $\mathscr{C}$ is* Φ-invariant *if $\mathscr{C}$ is stable under the action of* Φ.

The set of matrices of size $k \times n$ with $k \geqslant 1$ and $n \geqslant 1$, and entries in a commutative ring $\mathcal{A}$ is denoted by $\mathscr{M}_{k,n}(\mathcal{A})$. Furthermore, we *will always assume* that 0 belongs to I and ϕ_0 is the identity function.

Definition 21. *Let ℓ be the cardinality of I. We define $\Phi : \mathbb{F}_q^n \longrightarrow \mathscr{M}_{\ell,n}(\mathbb{F}_q)$ as the mapping that sends $\boldsymbol{v}$ from $\mathbb{F}_q^n$ to the matrix $\Phi(\boldsymbol{v})$ whose i-th row is $\phi_i(\boldsymbol{v})$.*

A generator matrix of a Φ-invariant $\mathscr{C}$ is then obtained by first picking a codeword $\boldsymbol{a}$ from $\mathscr{C}$ and next checking whether or not the set $\Phi(\boldsymbol{a})$ is a generating set of $\mathscr{C}$ as a vector space. If that is not the case then another $\boldsymbol{b}$ is picked from $\mathscr{C}$. If the linear space spanned by $\Phi(\boldsymbol{a}) \cup \Phi(\boldsymbol{b})$ is not equal to $\mathscr{C}$ then another $\boldsymbol{c}$ is picked, and the process continues so on until enough vectors are picked to generate $\mathscr{C}$. One sees that the representation of $\mathscr{C}$ is just reduced to the vectors $\{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \ldots\}$. The other codewords are reconstructed thanks to Φ.

5.2 $\mathbb{F}_q[G]$-Algebra

The main question with Φ-invariant codes is to exhibit interesting mappings ϕ. There exists a general approach for dealing with this point. We will see for instance that cyclic and dyadic codes actually comply with the following principle. Let us assume that the set of indices $\{1, \ldots, n\}$ is in one-to-one correspondence with a *commutative* group $(G, +)$ of order n. For simplicity, we use the formal summation $\sum_{g \in G} v_g X^g$ where v_g is from $\mathbb{F}_q$ to denote a vector $\boldsymbol{v} = (v_1, \ldots, v_n)$ from $\mathbb{F}_q^n$. This new formalism will enable to view $\mathbb{F}_q^n$ as the ring algebra $\mathbb{F}_q[G]$ with:

$$\mathbb{F}_q[G] \stackrel{\text{def}}{=} \left\{ \sum_{g \in G} v_g X^g : \forall g \in G, v_g \in \mathbb{F}_q \right\}.$$

We recall that $\mathbb{F}_q[G]$ is a commutative ring where the addition of $\sum_{g \in G} v_g X^g$ and $\sum_{g \in G} w_g X^g$ is of course $\sum_{g \in G} (v_g + w_g) X^g$, and the product operation $\times$ is defined as:

$$\left(\sum_{g \in G} v_g X^g \right) \times \left(\sum_{g \in G} w_g X^g \right) = \sum_{g \in G} \left(\sum_{h + \ell = g} v_h w_\ell \right) X^g.$$

We will *exclusively* consider the set of maps $\Phi = \{\phi_g : g \in G\}$ where $\phi_g : G \to G$ is defined by $\phi_g(z) = z + g$. In particular, we have $\ell = n$. Circulant and dyadic matrices fit precisely in that framework as shown by the following examples.

Example 1. One can easily check that for cyclic codes of length n, G is the cyclic group $(\mathbb{Z}_n, +)$ with n elements. In particular, $\mathbb{F}_q[\mathbb{Z}_n]$ is exactly the quotient polynomial ring $(\mathbb{F}_q[X]/(X^n - 1), +, \times)$.

Example 2. Dyadic codes of length n are obtained with $n = 2^\ell$ for some integer $\ell \geqslant 1$ and a group G that is equal to $(\mathbb{F}_2^\ell, +)$.

From now on, $\mathbb{F}_q^n = \mathbb{F}_q[G]$ and we set $\Phi(\mathbb{F}_q[G]) \stackrel{\text{def}}{=} \{\Phi(\boldsymbol{v}) : \boldsymbol{v} \in \mathbb{F}_q[G]\}$ which is clearly a subset of the ring $\mathscr{M}_{n,n}(\mathbb{F}_q)$. In particular, when $G = \mathbb{Z}_n$ then $\Phi(\mathbb{F}_q[G]) = \mathfrak{C}_n(\mathbb{F}_q)$ (Definition 15 in Chapter 4). When $G = \mathbb{F}_q^\ell$ then $\Phi(\mathbb{F}_q[G]) = \mathfrak{D}_n(\mathbb{F}_q)$ with $n = 2^\ell$ (Definition 16 in Chapter 4). The following proposition shows actually that $\Phi(\mathbb{F}_q[G])$ is always a ring because Φ is an injective ring morphism.

Proposition 9. *$(\Phi(\mathbb{F}_q[G]), +, \times)$ is a commutative ring isomorphic to $(\mathbb{F}_q[G], +, \times)$.*

Proof. Let us fix $\boldsymbol{v} = \sum_{g \in G} v_g X^g$ and $\boldsymbol{w} = \sum_{g \in G} w_g X^g$ from $\mathbb{F}_q[G]$, and let us denote their product by $\boldsymbol{z} = \sum_{g \in G} z_g X^g$. The only difficult point is to prove:

$$\Phi(\boldsymbol{z}) = \Phi(\boldsymbol{v}) \times \Phi(\boldsymbol{w}).$$

To do so, we will show that entries of $\Phi(\boldsymbol{z})$ are equal to those of $\Phi(\boldsymbol{v}) \times \Phi(\boldsymbol{w})$. For the sake of simplicity, let us denote by $\boldsymbol{A}$ this last matrix. The entry of $\Phi(\boldsymbol{z})$ at the row indexed by g and at the column indexed k with g and k in G is equal to the entry at column k of the vector $\boldsymbol{z}^{\phi_g}$, that is to say to z_{k-g}. But we know that:

$$z_{k-g} = \sum_{h+\ell=k-g} v_h w_\ell.$$

On the other hand, the entry $a_{g,k}$ verifies the equality:

$$a_{g,k} = \sum_{\ell \in G} v_{\ell-g} w_{k-\ell} = \sum_{h+\ell=k-g} v_h w_\ell = z_{k-g}$$

which concludes the proof. □

Corollary 1. *$(\mathfrak{C}_n(\mathbb{F}_q), +, \times)$ and $(\mathbb{F}_q[X]/(X^n - 1), +, \times)$ are isomorphic.*

This important property about $\Phi(\mathbb{F}_q[G])$ of being a commutative ring can be harnessed to build other interesting matrices. Indeed, we can consider now matrices with entries from $\Phi(\mathbb{F}_q[G])$. Such matrices are actually block matrices where each block is from $\Phi(\mathbb{F}_q[G])$. In the remaining and thanks to Proposition 9, they will be treated as matrices with entries in $\mathbb{F}_q[G]$. The operations of addition and multiplication are then well-defined and clearly satisfy the axioms of a ring. This kind of matrices are used for instance to build *quasi-cyclic* and *quasi-dyadic* codes.

Definition 22. *A linear code of N with $N = bn$ for some integer $b \geqslant 1$ is* quasi-cyclic *(resp.* quasi-dyadic*) of order n if it is defined by a block (generator or parity-check) matrix where each block is an $n \times n$ circulant (resp. dyadic) matrix.*

5.3 Φ-Invariant Codes in Cryptography

The existence of codes invariant under the action of some specific transformations $\Phi = \{\phi_i : i \in I\}$ while being equipped with an efficient decoding algorithm is an important issue in coding theory and cryptography. They represent what is being ardently sought over decades because they combine both time and memory efficiency. Furthermore, it is a challenge to use such codes in cryptography in order to come up with a secure and efficient McEliece-like encryption scheme. Concretely, a cryptographer has to focus on a set of transformation Φ and then identify among the existing families of codes having a decoding algorithm those that contain Φ-invariant codes. Eventually, these schemes has to propose public matrices that are derived from secret Φ-invariant codes. In particular, the transformation $\boldsymbol{S}$ and $\boldsymbol{P}$ applied to the secret matrices have to preserve the Φ-invariant property.

Currently, there are (at least) five McEliece variants with a reduced representation. They can be divided into two branches: one includes variants that use quasi-cyclic codes [Gab05, BC07, BBC08, BCGO09], and the other is based on the use of quasi-dyadic codes [MB09, Per11, BLM11]. The encryption schemes that used precisely this type of codes are the following:

- Quasi-cyclic case: subcode of BCH code ([Gab05]), LDPC code ([BC07]), alternant code ([BCGO09]).
- Dyadic case: classical Goppa code ([MB09]), Srivastava code ([Per11]).

We will focus more particularly in the four proposals [Gab05, BC07, BCGO09, MB09]. We will show how to cryptanalyse them either by specific methods [OTD08, OTD10] or by a more general technique [FOPT10a] that covers the McEliece cryptosystem.

Chapter 6

Cryptanalysis of a Quasi-Cyclic BCH Scheme[1]

6.1 Description

BCH codes are well-known cyclic codes that can be decoded efficiently. Moreover, cyclic codes of length n over $\mathbb{F}_q$ are completely determined by a unique monic polynomial in $\mathbb{F}_q[X]$ that divides X^n-1. Consequently, there is one-to-one correspondence between cyclic codes and divisors of X^n-1. It is thus illusory to use them in a McEliece-like cryptosystem.

The idea of [Gab05] is to consider subcodes of a (cyclic) BCH code $\mathscr{C}_0$ of length N and dimension K. By doing so, it "artificially" increases the number of possible codes, and hence preventing from an attack based on an exhaustive search. It is assumed that $N = nN_0$ for some non-negative integers n and N_0. The key generation algorithm consists in picking randomly k_0 codewords $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{k_0}$ from $\mathscr{C}_0$ where k_0 is the greatest integer such that $k_0 n \leqslant K - n$. For the sake of simplicity, we assume that there exists K_0 such that $K = nK_0$ so that $k_0 = K_0 - 1$. These codewords will serve to define a quasi-cyclic code $\mathscr{C}$ of dimension $k \stackrel{\text{def}}{=} nk_0$. A generator matrix of $\mathscr{C}$ is obtained by repeatedly block shifting each $\boldsymbol{c}_i$. The resulting matrix is a $k \times N$ block matrix $\boldsymbol{G}_s = (\boldsymbol{B}_{i,j})_{\substack{1 \leqslant i \leqslant k_0 \\ 1 \leqslant j \leqslant N_0}}$ where $\boldsymbol{B}_{i,j}$ is an $n \times n$ circulant matrix. To build the public generator matrix, $\boldsymbol{S}$ and $\boldsymbol{P}$ are random block matrices with circulant blocks. Actually, the role of $\boldsymbol{P}$ is to reorder the blocks. There is therefore a permutation π such that $\boldsymbol{G}_s\boldsymbol{P} = \left(\boldsymbol{B}_{i,\pi^{-1}(j)}\right)_{\substack{1 \leqslant i \leqslant k_0 \\ 1 \leqslant j \leqslant N_0}}$. Hence π belongs to the symmetric group $\mathfrak{S}_{N_0}$ of degree N_0 and if we denote by $\boldsymbol{\Pi}$ the $N_0 \times N_0$ matrix associated to π, we have $\boldsymbol{P} = \boldsymbol{\Pi} \otimes \boldsymbol{I}_n$ where $\boldsymbol{I}_n$ is the $n \times n$ identity matrix and $\otimes$ denotes the Kronecker product according to the following definition.

Definition 23. *The Kronecker product of two matrices $\boldsymbol{A} = (a_{i,j})$ and $\boldsymbol{B} = (b_{i,j})$ with entries in a commutative ring $\mathcal{A}$ denoted by $\boldsymbol{A} \otimes \boldsymbol{B}$ is:*

$$\boldsymbol{A} \otimes \boldsymbol{B} \stackrel{\text{def}}{=} (a_{i,j}\boldsymbol{B}).$$

6.2 Key-Recovery Attack

We describe the method we proposed in [OTD10] that recovers the secret permutation π. It exploits three facts:

1. The code $\mathscr{C}_0$ admits a binary $(N-K) \times N$ parity check matrix $\boldsymbol{H}_0$ which can be assumed to be known. There are only a few different primitive BCH codes and we can try all of them. This is a consequence of the fact that the number of such codes is clearly upper-bounded by the number of primitive polynomials of degree m with $N \leqslant 2^m$.

2. Since $\mathscr{C}$ is a subcode of $\mathscr{C}_0$, any n-bit codeword $\boldsymbol{c}$ of $\mathscr{C}$ must satisfy the equation:

$$\boldsymbol{H}_0\boldsymbol{c}^T = \boldsymbol{0}.$$

3. Finding $\boldsymbol{\Pi}$ actually boils down to solving a linear system of n_0^2 unknowns representing the entries of $\boldsymbol{\Pi}^{-1}$ such that:

$$\boldsymbol{H}_0\boldsymbol{\Pi}\boldsymbol{G}^T = \boldsymbol{0}.$$

Each row of $\boldsymbol{G}$ provides $(N-K)$ binary linear equations. Thus the total number of linear equations is $k(N-K) = n^2(K_0 - 1)(N_0 - K_0)$ that are satisfied by N_0^2 unknowns. The parameters proposed in [Gab05] are such that $n > N_0$. Hence, the linear system is heavily over-constrained. For instance, Parameters A give $2,025$ unknowns that satisfy $695,604$ equations, and Parameters B give 529 unknowns and $316,840$ equations. Many of these equations are obviously linearly dependent. The success of this method depends on the size of the solution vector space. An implementation in Magma software actually always gave in both cases a vector space of dimension one which is the secret permutation.

[1] This chapter is partially taken from [OTD10].

6.3 Conclusion

Our attack on the cryptosystem [Gab05] aims at recovering the permutation matrix $\boldsymbol{P}$. We have seen that the role of the permutation matrix $\boldsymbol{P}$ in the McEliece cryptosystem is to increase the number of secret support vectors $\boldsymbol{x}$. The idea of the attack against [Gab05] is to fix a code $\mathscr{C}_0$ for which a decoding algorithm is known and for which the public code is permutation equivalent to it. It then sets up a linear system that the entries of the permutation matrix necessarily satisfy. In the case of the McEliece cryptosystem with a public code of length n and dimension k, this method could not reveal it because the number of equations is $n(n-k)$ and the number of unknowns is n^2.

As for the scheme of [Gab05], there exists essentially a single code $\mathscr{C}_0$. Furthermore, $\boldsymbol{P}$ has to be chosen among a subset of all the possible permutations. It results in a permutation matrix defined by only n_0^2 entries with $n_0 < n$ and secondly, the linear system has many more linear equations that n_0^2. In practise, we obtain a linear system with a single solution.

This result shows that focusing on an extremely large family of codes is not a sufficient condition for ensuring a secure scheme. In particular, the idea of artificially adding diversity by considering subcodes of a given code can be annihilated. Secondly, it demonstrates that any technique of key-size reduction has to take into account of the *real* impact on the number of unknowns that model the cryptosystem and how they can be exploited to devise an attack. Finally, the efficiency of our attack proves that the scheme of [Gab05] could not be repaired.

Chapter 7

Cryptanalysis of a Quasi-Cyclic LDPC Scheme[1]

7.1 Description

LDPC codes are linear codes defined by sparse parity-check matrices. This sparsity is exploited to devise efficient decoding algorithms. Classically, they are decoded by an iterative algorithm which can be assimilated to an instance of the more general Belief Propagation algorithm.

The main challenge when using LDPC codes in cryptography is to prevent an attacker to reconstruct a sparse parity-check matrices. Hence, the public code defined by $\boldsymbol{G}$ should not have low-weight codewords in its dual. This represents clearly a weakness as demonstrated by the work [MRS00]. The other disadvantage of using LDPC codes is the constraint of taking very long codes in order for the iterative decoding algorithm to perform well.

The article [BC07] attempts to solve these two points by first proposing a way to both eliminate the presence of low-weight codewords in the dual and to decrease the size of matrices through the use of quasi-cyclic matrices. As in [BC07], we consider LDPC codes of length $n = pn_0$ and dimension $k = p(n_0 - 1)$ defined by a parity-check matrix $\boldsymbol{H}$ of the following form:

$$\boldsymbol{H} = \begin{pmatrix} \boldsymbol{H}_1 & \cdots & \boldsymbol{H}_{n_0} \end{pmatrix} \tag{7.1}$$

where each matrix $\boldsymbol{H}_j$ is a sparse circulant matrix of size $p \times p$. Without loss of generality, $\boldsymbol{H}_{n_0}$ is of full rank. Each column of $\boldsymbol{H}$ has a fixed weight d_v which is very small compared to the length n. We also assume that one has a good approximation of the number t of correctable errors through iterative decoding of the code defined by $\boldsymbol{H}$.

Unlike the McEliece cryptosystem, the encryption scheme of [BC07] does not take a permutation matrix $\boldsymbol{P}$ but an $n \times n$ invertible matrix $\boldsymbol{Q}$ such that the weight of each row and each column is m which is a integer appropriately chosen. The invertible $k \times k$ matrix $\boldsymbol{S}$ has also a particular constraint: each row and each column is of weight s. The private key consists of the triple $(\boldsymbol{S}, \boldsymbol{H}, \boldsymbol{Q})$. The public key $\boldsymbol{G}$ is the matrix $\boldsymbol{S}^{-1}\boldsymbol{G}'\boldsymbol{Q}^{-1}$ where $\boldsymbol{G}'$ is a generator matrix in row reduced echelon form deduced from $\boldsymbol{H}$. The plaintext space is $\mathbb{F}_2^k$ and the ciphertext space is $\mathbb{F}_2^n$. The encryption of $\boldsymbol{x} \in \mathbb{F}_2^k$ requires to randomly pick an n-bit vector $\boldsymbol{e}$ of weight $t' \leqslant t/m$. The corresponding ciphertext is $\boldsymbol{c} = \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e}$. The decryption step consists in iteratively decoding $\boldsymbol{c}\boldsymbol{Q} = \boldsymbol{x}\boldsymbol{S}^{-1}\boldsymbol{G}' + \boldsymbol{e}\boldsymbol{Q}$ to output $\boldsymbol{z} = \boldsymbol{x}\boldsymbol{S}^{-1}$ and then computing $\boldsymbol{x} = \boldsymbol{z}\boldsymbol{S}$. The crucial point that makes this cryptosystem valid is that the weight of $\boldsymbol{e}\boldsymbol{Q}$ is always less than or equal to $t'm \leqslant t$.

It is suggested in [BC07] to take a matrix $\boldsymbol{Q}$ in diagonal form and to attribute the following values: $p = 4032$, $n_0 = 4$, $d_v = 13$, $m = 7$ and $t = 190$ ($t' = 27$). Finally, each circulant block of $\boldsymbol{S}$ has a column/row weight equals to m so that $s = m(n_0 - 1)$.

From now on, we use the notation $\boldsymbol{S} = (\boldsymbol{s}_{i,j})_{\substack{1 \leqslant i \leqslant n_0 - 1 \\ 1 \leqslant j \leqslant n_0 - 1}}$ and $\boldsymbol{Q} = (\boldsymbol{q}_{i,j})_{\substack{1 \leqslant i \leqslant n_0 \\ 1 \leqslant j \leqslant n_0}}$ where $\boldsymbol{s}_{i,j}$ and $\boldsymbol{q}_{i,j}$ are from $\mathbb{F}_2[\mathbb{Z}_p] \simeq \mathbb{F}_2[X]/(X^p - 1)$. By assumption, $\boldsymbol{q}_{i,j} = 0$ when $i \neq j$. For the sake of simplicity, we set $\boldsymbol{q}_i \stackrel{\text{def}}{=} \boldsymbol{q}_{i,i}$. Since $\boldsymbol{Q}$ is invertible, $\boldsymbol{q}_i$ is also necessarily invertible modulo $X^p - 1$

7.2 Key-Recovery Attack

We have derived in [OTD10] a cryptanalysis that recovers the secret code $\mathscr{C}$ defined by $\boldsymbol{H}$. The attack fully exploits the fact that $\boldsymbol{Q}$ is diagonal. It is straightforward to see that $\boldsymbol{G}'$ verifies:

$$\boldsymbol{G}' = \left(\begin{array}{c|c} & (\boldsymbol{H}_{n_0}^{-1}\boldsymbol{H}_1)^T \\ \boldsymbol{I}_k & \vdots \\ & (\boldsymbol{H}_{n_0}^{-1}\boldsymbol{H}_{n_0-1})^T \end{array} \right).$$

[1] This chapter is taken from the articles [OTD08, OTD10].

In others words, if $\boldsymbol{G}_{\leqslant k}$ is the restriction of $\boldsymbol{G}$ to the first k columns then we have:

$$\boldsymbol{G}_{\leqslant k} = \boldsymbol{S}^{-1} \times \begin{pmatrix} \boldsymbol{q}_1^{-1} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \boldsymbol{q}_{n_0-1}^{-1} \end{pmatrix}.$$

Let us denote the entries of $\boldsymbol{G}_{\leqslant k}^{-1}$ from $\mathbb{F}_2[X]/(X^p-1)$ by $\boldsymbol{r}_{i,j}$ with $1 \leqslant i \leqslant n_0-1$ and $1 \leqslant j \leqslant n_0-1$. Clearly, we have:

$$\boldsymbol{r}_{i,j} = \boldsymbol{q}_i \cdot \boldsymbol{s}_{i,j} \mod (X^p-1). \tag{7.2}$$

On can prove that, as $\boldsymbol{q}$ and $\boldsymbol{s}$ are of low weight, the weight of $\boldsymbol{r}$ is m^2 with high probability [OTD10]. This means that among the nonzero positions of $\boldsymbol{r}$, some should belong to $\boldsymbol{q}_i$ shifted by a certain number of times. So, one possible strategy to recover the polynomial $\boldsymbol{q}_i$ consists in enumerating m-tuples $\sum_{i\in\mathbb{Z}_p} u_i X^i$ that belong to the set of nonzero positions of $\boldsymbol{r}_{i,j}$ until $\boldsymbol{u}^{-1}\boldsymbol{r}_{i,j}$ is of weight m for each integer j such that $1 \leq j \leq n_0-1$. The cost of this method is $O\left(\binom{m^2}{m} \cdot p^2\right)$. It corresponds to $2^{50.3}$ operations for the parameters proposed in [BC07].

Actually, one can perform a better attack as explained in [OTD10]. Let us set $\boldsymbol{d}_{i,j} \stackrel{\text{def}}{=} \boldsymbol{r}_{i,j}\boldsymbol{r}_{i,1}^{-1}$ and consider the quasi-cyclic code $\mathscr{E}_i$ defined by the following generator matrix:

$$\boldsymbol{E}_i = \begin{pmatrix} \boldsymbol{I}_p & \boldsymbol{d}_{i,2} & \cdots & \boldsymbol{d}_{i,n_0-1} \end{pmatrix}.$$

$\mathscr{E}_i$ contains at least p codewords of low weight $(n_0-1)m = 3\times 7 = 21$ since

$$\boldsymbol{s}_{i,1} \times \boldsymbol{E}_i = \begin{pmatrix} \boldsymbol{s}_{i,1} & \boldsymbol{s}_{i,2} & \cdots & \boldsymbol{s}_{i,n_0-1} \end{pmatrix}.$$

It is hence easy to recover $\boldsymbol{s}_{i,1},\ldots,\boldsymbol{s}_{i,n_0-1}$ by applying an algorithm dedicated to the search of low-weight codewords in a linear code. For instance, the time complexity of Stern's algorithm is 2^{32}. After recovering $\boldsymbol{S}$, $\boldsymbol{q}_1,\ldots,\boldsymbol{q}_{n_0-1}$, one has at his disposal the matrix:

$$\boldsymbol{G}' \times \begin{pmatrix} \boldsymbol{I}_p & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \boldsymbol{I}_p & 0 \\ 0 & \cdots & 0 & \boldsymbol{q}_{n_0}^{-1} \end{pmatrix} = \left(\begin{array}{c|c} & \boldsymbol{a}_1 \\ \boldsymbol{I}_k & \vdots \\ & \boldsymbol{a}_{n_0-1} \end{array}\right)$$

with $\boldsymbol{a}_i \stackrel{\text{def}}{=} (\boldsymbol{h}_{n_0}^{-1}\boldsymbol{h}_i)^T \times \boldsymbol{q}_{n_0}^{-1}$ for $1 \leq i \leq n_0-1$, and where for any $\boldsymbol{v}$ from $\mathbb{F}_2[\mathbb{Z}_p]$, $\boldsymbol{z}^T$ represents the unique element of $\mathbb{F}_2[\mathbb{Z}_p]$ that defines the transpose of the circulant matrix defined by $\boldsymbol{v}$.

We recall that $\boldsymbol{h}_1,\ldots,\boldsymbol{h}_{n_0}$ and $\boldsymbol{q}_{n_0}$ are still unknown. Let us observe that $(\boldsymbol{a}_i\boldsymbol{a}_{i'}^{-1})^T = \boldsymbol{h}_i\boldsymbol{h}_{i'}^{-1}$ whenever $\boldsymbol{h}_{i'}$ is invertible. Thus by defining $\boldsymbol{b}_{i,j} \stackrel{\text{def}}{=} (\boldsymbol{a}_i\boldsymbol{a}_j^{-1})^T$ and observing that $\boldsymbol{b}_{i,j}\boldsymbol{h}_j = \boldsymbol{b}_{i,j'}\boldsymbol{h}_{j'} = \boldsymbol{h}_i$, one sees that the code defined by the generator matrix $\boldsymbol{G}_i \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{b}_{1,i} & \cdots & \boldsymbol{b}_{i-1,i} & \boldsymbol{I}_p & \boldsymbol{b}_{i+1,i} & \cdots & \boldsymbol{b}_{n_0-1,i} \end{pmatrix}$ contains low-weight codewords. Indeed, we have:

$$\boldsymbol{h}_i \times \boldsymbol{G}_i = \begin{pmatrix} \boldsymbol{h}_1 & \boldsymbol{h}_2 & \cdots & \boldsymbol{h}_{n_0-1} \end{pmatrix}.$$

The minimum distance of this code is less than or equal to $(n_0-1)d_v$. For instance, the work factor of Stern's algorithm for searching codewords of weight $(n_0-1)d_v = 3\times 13 = 39$ in a code of dimension $p = 4032$ and length $p(n_0-1) = 12096$ is about 2^{37} operations.

Once $\boldsymbol{h}_1,\ldots,\boldsymbol{h}_{n_0-1}$ are recovered, $\boldsymbol{f}_i \stackrel{\text{def}}{=} (\boldsymbol{h}_i^T)^{-1}\boldsymbol{a}_i = (\boldsymbol{h}_{n_0}^{-1})^T\boldsymbol{q}_{n_0}^{-1}$ is computed for any integer i with $1 \leq i \leq n_0-1$. Here again recovering the remaining secret quantities $\boldsymbol{h}_{n_0}$ and $\boldsymbol{q}_{n_0}$ can be done for instance by seeking low-weight codewords in the code defined by the generator matrix $\begin{pmatrix} \boldsymbol{I}_p & \boldsymbol{f}_i \end{pmatrix}$.

7.3 Conclusion

The scheme presented in [BC07] introduced a new technique to generate a public code from a secret LDPC code. Instead of taking a permuation matrix $\boldsymbol{P}$, it rather uses a matrix $\boldsymbol{Q}$ where each row/column has a small weight > 1. It is also based on a sparse secret matrix $\boldsymbol{S}$ for efficiency reasons. The attack against [BC07] that we described here fully exploits that sparsity of $\boldsymbol{S}$. A natural reparation would be to remove this sparsity constraint. Actually, it is an open problem to know if there exists an efficient attack against this new scheme, and more importantly, a challenging issue is to propose a security reduction.

Chapter 8

Cryptosystems Based on Φ-Invariant Alternant Codes

The numerous unsuccessful attempts for designing a secure alternative to the McEliece cryptosystem lead us to believe that Goppa codes represent the only possible choice, or if not so, that any variant should be as close as possible to them. Especially, from a key-size reduction perspective, it is fundamental for example to identify Φ-invariant Goppa codes. Recently two works [BCGO09, MB09] decided to concentrate on alternant codes. They are based on quasi-cyclic alternant codes in [BCGO09] and quasi-dyadic Goppa codes in [MB09]. The approach is quite attractive because it results in an important improvement in the reduction of the public key size. In [BCGO09], the size ranges between $8,000$ and $20,000$ bits, whereas it lies between $4,000$ and $20,000$ bits in [MB09]. The goal of this chapter is briefly to describe these two proposals.

8.1 Quasi-Cyclic Alternant Encryption Scheme

We present here the cryptosystem we designed in [BCGO09]. It consists in building quasi-cyclic alternant codes over $\mathbb{F}_q$ with $2^8 \leqslant q \leqslant 2^{16}$.

It is well-known that Reed-Solomon codes over any field are cyclic. We propose to reorder the coordinates in order to obtain quasi-cyclic Reed-Solomon over a fixed field $\mathbb{F}_q$. Let α be a primitive element of $\mathbb{F}_{q^m}$ and let us set $N = q^m - 1$. We assume that $N = N_0\ell$ and we define $\beta \stackrel{\text{def}}{=} \alpha^{N_0}$. Clearly, β is of order ℓ. Let t be a positive integer and let $\boldsymbol{U}_{2t} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{A}_0 & \cdots & \boldsymbol{A}_{N_0-1} \end{pmatrix}$ be the block parity-check matrix where for any integer j such that $0 \leqslant j \leqslant N_0 - 1$:

$$\boldsymbol{A}_j \stackrel{\text{def}}{=} \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha^j & \alpha^j\beta & \cdots & \alpha^j\beta^{\ell-1} \\ \vdots & \vdots & & \vdots \\ (\alpha^j)^{2t-1} & (\alpha^j\beta)^{2t-1} & \cdots & (\alpha^j\beta^{\ell-1})^{2t-1} \end{pmatrix}. \tag{8.1}$$

Proposition 10 ([BCGO09])**.** *The code of length N defined by the parity-check matrix $\boldsymbol{U}_{2t}$ is quasi-cyclic of order ℓ and dimension K with $N - K = 2t + 1$.*

This code will be used to obtain a quasi-cyclic alternant code of index ℓ over $\mathbb{F}_q$ and length $n \stackrel{\text{def}}{=} n_0\ell$ with $n_0 < N_0$. The idea is to successively apply three operations that preserve the quasi-cyclic feature:

1. Randomly block shortening the code in order to obtain a code of length $n = n_0\ell$ with $n_0 < N_0$.
2. Multiplying each column by a nonzero scalar from $\mathbb{F}_{q^m}$ to get a quasi-cylic generalized Reed Solomon code of length n.
3. Performing the subfield subcode operation over $\mathbb{F}_q$.

We give the different steps of the key generation algorithm of the cryptosystem [BCGO09]. Let n_0 be an integer such that $n_0 < N_0$. This step consists in randomly choosing n_0 different circulant blocks $\boldsymbol{A}_{j_1}, \ldots, \boldsymbol{A}_{j_{n_0}}$ of the parity-check matrix $\boldsymbol{U}_{2t}$. Let us insist that we do not have necessarily $j_1 \leqslant j_2 \leqslant \cdots \leqslant j_{n_0}$. We then consider the code of length $n = \ell n_0$ over $\mathbb{F}_{q^m}$ defined by the following parity-check matrix $\boldsymbol{V}_{2t} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{A}_{j_1} & \boldsymbol{A}_{j_2} & \cdots & \boldsymbol{A}_{j_{n_0}} \end{pmatrix}$. Next, for any integer s such that $1 \leqslant s \leqslant \ell - 1$, let $\boldsymbol{D}_s = (d_{i,j})$ be the $\ell \times \ell$ diagonal matrix such $d_{i,i} = (\beta^s)^{i-1}$, that is to say:

$$\boldsymbol{D}_s = \begin{pmatrix} 1 & & & \\ & \beta^s & & \\ & & \ddots & \\ & & & (\beta^s)^{\ell-1} \end{pmatrix}. \tag{8.2}$$

We consider now an n_0-tuple $\boldsymbol{a} = (a_1, \ldots, a_{n_0})$ of nonzero elements from $\mathbb{F}_{q^m}$ and an integer s with $1 \leqslant s \leq \ell - 1$. Let $\boldsymbol{Q}$ be the $2t \times n$ block parity-check matrix $\boldsymbol{Q} = \begin{pmatrix} \boldsymbol{Q}_1 & \cdots & \boldsymbol{Q}_{n_0} \end{pmatrix}$ with for any i such that $1 \leqslant i \leqslant n_0$:

$$\boldsymbol{Q}_i \stackrel{\text{def}}{=} a_i \boldsymbol{A}_{j_i} \boldsymbol{D}_s. \tag{8.3}$$

This matrix defines a generalized Reed-Solomon code of length n that is quasi-cyclic of order ℓ and dimension $K = N - 2t$.

The public key is then a $k \times n$ generator matrix of the alternant code over $\mathbb{F}_q$ defined by the parity-check matrix $\boldsymbol{Q}$. The integer t is chosen such that k, which is necessarily equal to $n - 2tm$, is multiple of ℓ. Hence, there exist k_0 such that $k = k_0\ell$. The public matrix $\boldsymbol{G}$ is then a block matrix $\boldsymbol{G} = (\boldsymbol{G}_{i,j})$ where each block $\boldsymbol{G}_{i,j}$ is an $\ell \times \ell$ circulant matrix. Finally, the private key is composed of $(j_1, \ldots, j_{n_0})$, the integer s and $(a_1, \ldots, a_{n_0})$.

8.2 Quasi-Dyadic Goppa Code Encryption Scheme

The cryptosystem presented in [MB09] considers particular alternant codes called *quasi-dyadic* Goppa codes. The scheme in [MB09] takes a Goppa code over $\mathbb{F}_q$ of length $n \leqslant q^m$ defined by a polynomial $\gamma(z)$ of degree ℓ with coefficients in $\mathbb{F}_{q^m}$ and for which $\boldsymbol{x}$ is an n-tuple of distinct elements from $\mathbb{F}_{q^m}$ and such that $\boldsymbol{x}$ does not contain a root of $\gamma(z)$. The scheme enforces $\gamma(z)$ to have ℓ distinct roots $\boldsymbol{z} = (z_0, \ldots, z_{\ell-1})$ from $\mathbb{F}_{q^m}$ so that we have:

$$\gamma(z) = \prod_{i=0}^{\ell-1} (z - z_i).$$

In that special case $\mathscr{G}(\boldsymbol{x}, \gamma)$ admits a parity-check matrix $\boldsymbol{C}(\boldsymbol{z}, \boldsymbol{x})$ in Cauchy form [MS86, p. 345], that is to say:

$$\boldsymbol{C}(\boldsymbol{z}, \boldsymbol{x}) \begin{pmatrix} \dfrac{1}{z_0 - x_1} & \cdots & \dfrac{1}{z_0 - x_n} \\ \vdots & & \vdots \\ \dfrac{1}{z_{\ell-1} - x_1} & \cdots & \dfrac{1}{z_{\ell-1} - x_n} \end{pmatrix}.$$

The goal of the scheme [MB09] is build a Goppa code that admits a parity-check matrix that is both a Cauchy matrix and a block matrix where each block is dyadic. An $\ell \times \ell$ matrix $\boldsymbol{\Delta} = (\Delta_{i,j})$ with $0 \leq i \leq \ell - 1$ and $0 \leq j \leq \ell - 1$ is *dyadic* if and only if $\Delta_{i,j} = h_{i \oplus j}$ where $\oplus$ is the bitwise exclusive-or on the binary representation of the indices and $\boldsymbol{h} = (h_0, \ldots, h_{\ell-1})$ is the first row of $\boldsymbol{\Delta}$. Let $\boldsymbol{h} = (h_0, \ldots, h_{N-1})$ be a vector of $\mathbb{F}_{q^m}^N$ with $\ell \leq N$. We denote by $\boldsymbol{\Delta}_\ell(\boldsymbol{h}) = (\Delta_{i,j})$ the $\ell \times N$ matrix such that $\Delta_{i,j} = h_{i \oplus j}$. One can easily observe that $\boldsymbol{\Delta}_\ell(\boldsymbol{h})$ is the juxtaposition of N_0 dyadic matrices of size $\ell \times \ell$ when $N = N_0\ell$ for some integer N_0. Proposition 11 proved in [MB09, Theorem 2] characterizes dyadic Cauchy matrices.

Proposition 11. *A necessary and sufficient condition for $\boldsymbol{\Delta}_\ell(\boldsymbol{h})$ to be a Cauchy matrix $\boldsymbol{C}(\boldsymbol{z}, \boldsymbol{x})$ is that $\mathbb{F}_{q^m}$ is of characteristic 2 and for any i, j in $\{0, \ldots, N-1\}$ we have:*

$$\frac{1}{h_{i \oplus j}} = \frac{1}{h_j} + \frac{1}{h_i} + \frac{1}{h_0}. \tag{8.4}$$

Furthermore, for any $\theta \in \mathbb{F}_{q^m}$ and for any $z_i^* = 1/h_i + \theta$ and $x_j^* = 1/h_j + 1/h_0 + \theta$, the Cauchy matrix $\boldsymbol{C}(\boldsymbol{z}^*, \boldsymbol{x}^*)$ is equal to $\boldsymbol{\Delta}_\ell(\boldsymbol{h})$.

The public generator matrix $\boldsymbol{G}$ is then a $k \times n$ block matrix where each block is an $\ell \times \ell$ dyadic matrix with ℓ being a power of 2. The entries of $\boldsymbol{G}$ belong to $\mathbb{F}_q$ and the integers k and n are chosen such that $n = n_0\ell$ and $k = n - m\ell = \ell(n_0 - m)$ where n_0 is some integer and m defines the extension $\mathbb{F}_{q^m}$. The matrix $\boldsymbol{G}$ is obtained from a secret $\ell \times n$ block parity-check matrix $\boldsymbol{H} = \begin{pmatrix} \boldsymbol{\Delta}_\ell(\boldsymbol{f}_0) & \cdots & \boldsymbol{\Delta}_\ell(\boldsymbol{f}_{n_0-1}) \end{pmatrix}$ where each block $\boldsymbol{\Delta}_\ell(\boldsymbol{f}_j)$ with $0 \leqslant j \leqslant n_0 - 1$ is an $\ell \times \ell$ dyadic matrix and $\boldsymbol{f}_j$ is a vector of $\mathbb{F}_{q^m}^\ell$ such that:

$$\boldsymbol{f}_j = \gamma_j \left(h_{\omega_j \ell \oplus d_j}, h_{(\omega_j \ell + 1) \oplus d_j}, \ldots, h_{((\omega_j + 1)\ell - 1) \oplus d_j} \right)$$

where $\boldsymbol{h} = (h_0, \ldots, h_{N-1})$ is a random vector of $\mathbb{F}_{q^m}^N$ that satisfies Equation (8.4) and such that $N = N_0\ell$ for some integer $N_0 \gg n_0$. The integers ω_j, d_j are chosen such that $0 \leqslant \omega_j \leqslant N_0 - 1$ and $0 \leqslant d_j \leqslant \ell - 1$. The coefficients γ_j are non zero elements of $\mathbb{F}_{q^m}$. Note that the integers ω_j's are different. The secret key consists then of the vectors $\boldsymbol{h}$, $\boldsymbol{\omega} = (\omega_0, \ldots, \omega_{n_0-1})$, $\boldsymbol{d} = (d_0, \ldots, d_{n_0-1})$ and $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_{n_0-1})$.

Chapter 9

Algebraic Cryptanalysis[1]

Algebraic cryptanalysis is a general framework that permits to assess the security of theoretically all cryptographic schemes. So far, such type of attacks has been applied successfully against several multivariate schemes and stream ciphers. The principle is to associate to a cryptographic primitive a set of algebraic equations. The system of equations is constructed in a way to have a correspondence between the solutions of this system, and a secret information like for instance the secret key of an encryption scheme.

We have shown in [FOPT10a] that it is possible construct for the McEliece cryptosystem an algebraic system that a private key has to satisfy. We emphasise that this algebraic approach can be mounted against any McEliece-like cryptosystem as long as the considered codes are alternant codes. It is also important to recall that a Goppa code is a particular alternant code. This will be exploited to propose key-recovery attacks[2] against any cryptosystem based on alternant codes [McE78, BCGO09, MB09].

9.1 Algebraic Key-Recovery Attack

We explain more precisely how we construct the algebraic system. As explained, the McEliece cryptosystem relies on Goppa codes which belong to the class of *alternant* codes and inherit from this an efficient decoding algorithm. For the ease of presentation, we will consider the public key $\boldsymbol{G} = (g_{i,j})$ as a generator matrix of an alternant code over $\mathbb{F}_q$ of length $n \leqslant q^m$ and dimension k obtained by means of the n-tuple $\boldsymbol{x}$ of distinct elements x_j from $\mathbb{F}_{q^m}$ and the n-tuple $\boldsymbol{y}$ of nonzero elements y_j from $\mathbb{F}_{q^m}$. Let us recall $g_{i,j}$ belongs to $\mathbb{F}_q$ and $k \geqslant n - mr$. The key feature about an alternant code is the following fact.

Fact 1. *It is possible to decode $r/2$ errors in polynomial-time with an alternant code of degree r whenever a parity-check matrix is given in the form of $\boldsymbol{V}_r(\boldsymbol{x}, \boldsymbol{y})$.*

This fact has an important consequence for any McEliece-like cryptosystem based on alternant codes. The private key is therefore any vectors $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ from $\mathbb{F}_{q^m}^n$ such that the following equation $\boldsymbol{V}_r(\boldsymbol{x^*}, \boldsymbol{y^*})\boldsymbol{G}^T = \boldsymbol{0}$ holds, or equivalently:

$$\begin{pmatrix} y_1^* & \cdots & y_n^* \\ y_1^* x_1^* & \cdots & y_n^* x_n^* \\ \vdots & & \vdots \\ y_1^* (x_1^*)^{r-1} & \cdots & y_n^* (x_n^*)^{r-1} \end{pmatrix} \begin{pmatrix} g_{1,1} & \cdots & g_{k,1} \\ \vdots & & \vdots \\ g_{1,n} & \cdots & g_{k,n} \end{pmatrix} = \boldsymbol{0}. \tag{9.1}$$

Let $\boldsymbol{X} \stackrel{\text{def}}{=} (X_1, \ldots, X_n)$ and $\boldsymbol{Y} \stackrel{\text{def}}{=} (Y_1, \ldots, Y_n)$ be $2n$ unknowns where X_i corresponds to x_i^* and Y_i to y_i^*. We see that finding the matrix as in (9.1) is equivalent to solving the following system of polynomial equations defined for any integer i and j such that $1 \leqslant i \leqslant k$ and $0 \leqslant j \leqslant r-1$:

$$g_{i,1} Y_1 X_1^j + \cdots + g_{i,n} Y_n X_n^j = 0. \tag{9.2}$$

Let us notice that the solutions are seen in $\mathbb{F}_{q^m}$ whereas the entries $g_{i,j}$ are in the subfield $\mathbb{F}_q$. Furthermore, we see that this polynomial system is highly structured. It is also very sparse as the only monomials occurring are of the form $Y_i X_i^j$. Moreover each block of k equations obtained when i varies while j is fixed, is *bihomogeneous i.e.* homogeneous if the variables of $\boldsymbol{X}$ and $\boldsymbol{Y}$ are considered alone. We define this notion more formally.

Definition 24 ([FSS11])**.** *A polynomial $P(\boldsymbol{X}, \boldsymbol{Y})$ from $\mathbb{F}_{q^m}[\boldsymbol{X}, \boldsymbol{Y}]$ is said to be:*

- bihomogeneous *of* bi-degree (a, b) *if $P(\alpha\boldsymbol{X}, \beta\boldsymbol{Y}) = \alpha^a \beta^b P(\boldsymbol{X}, \boldsymbol{Y})$ for any α and β from $\mathbb{F}_{q^m}$.*
- bilinear *if it is of* bi-degree $(1, 1)$.
- affine bilinear *if there exists a bilinear polynomial $Q(\boldsymbol{X}, \boldsymbol{Y})$ and a nonzero element α from $\mathbb{F}_{q^m}$ such that $P(\boldsymbol{X}, \boldsymbol{Y}) = Q(\boldsymbol{X}, \boldsymbol{Y}) + \alpha$.*

[1] This chapter is taken from the articles [FOPT10a, FOPT10b].

[2] An independent work [UL09] also proposes key-recovery attacks against [BCGO09, MB09].

The existence of an algebraic polynomial system highlights a new way to tackle the security of the McEliece cryptosystem especially regarding the design of better approach for a key-recovery attack. But more importantly, it raises also several important question. The first fundamental issue is about the solving of such a system. *Are the existing methods able to solve it*? *Is it possible to determine the time and memory complexities of these methods?* Furthermore, we have seen that the system is very structured. In particular, the polynomial equations occurring in (9.2) are of bi-degree $(j, 1)$ with $\leqslant j \leqslant r-1$. So, *is it possible to exploit this particular features to devise dedicated and improved solving algorithms?*

Eventually, the number of equations is $rk \geqslant r(n-rm)$. In a cryptographic setting this number can be really high. For instance, the outdated parameters ($m = 10$, $r = 50$ and $n = 2^m$) proposed in the original paper [McE78] furnish at least $50 \times 524 = 26,200$ equations. On the other hand, the number of variables is $2n = 2048$ and more importantly, the degree of the polynomial equations is as high as $r - 1 = 49$. Actually, this analysis is not completely true when we consider binary irreducible Goppa codes as it is advocated in the encryption scheme of [McE78].

Theorem 1. *A binary Goppa code $\mathscr{G}(\boldsymbol{x}, \gamma)$ defined by a polynomial $\gamma(z)$ from $\mathbb{F}_{2^m}[z]$ of degree r without multiple roots is the alternant code $\mathscr{A}_{2r}(\boldsymbol{x}, \boldsymbol{y})$, with $y_i = \gamma(x_i)^{-2}$.*

This result about binary Goppa codes defined by polynomials that has no multiple roots, which is obviously the case for irreducible polynomials, leads to the following fact.

Fact 2 ([Pat75])**.** *There exists a polynomial time algorithm decoding all errors of Hamming weight at most r in a Goppa code $\mathscr{G}(\boldsymbol{x}, \gamma)$ when γ has degree r and has no multiple roots whenever $\boldsymbol{x}$ and $\gamma(z)$ are known.*

Consequently, when a Goppa code defined by n irreducible polynomial is viewed as an alternant defined by $\boldsymbol{x}$ and $\boldsymbol{y}$ with $y_i = \gamma(x_i)^{-2}$ for any i such that $1 \leqslant i \leqslant n$, then they are necessarily solutions of the following polynomial equations obtained for any integer i and j such that $1 \leqslant i \leqslant k$ and $0 \leqslant j \leqslant 2r-1$:

$$g_{i,1} Y_1 X_1^j + \cdots + g_{i,n} Y_n X_n^j = 0. \tag{9.3}$$

This particular system has therefore $2rk$ polynomial equations with a maximum degree $2r-1$. The number of unknowns is still $2n$. For instance, with original parameters of [McE78], the system has $2 \times 50 \times 524 = 52,400$ equations. Unfortunately, the maximum degree becomes extremely high, namely 99. With the current state of the art, it is not clear whether an algebraic attack can be mounted efficiently against the original McEliece cryptosystem.

The fact that a binary Goppa code can have multiple descriptions raises the more general fundamental issue of finding all possible description of an alternant code $\mathscr{A}$, that is to say all $\boldsymbol{x}$ and $\boldsymbol{y}$ such that $\mathscr{A} = \mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$. When the extension field $\mathbb{F}_{q^m}$ is the same as the definition field $\mathbb{F}_q$ *i.e.* if $m = 1$ the problem was solved in [Dür87]. This was the key of the cryptanalysis of McEliece's variant based on generalized Reed-Solomon codes [SS92]. Proposition 12 shows that there exist several solutions by setting one of the Y_i and two values of the X_i and X_j to arbitrary values provided that $Y_i \neq 0$ and $X_i \neq X_j$.

Proposition 12 ([MS86, Chap. 10, p. 305])**.** *Let $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ be a solution of* (9.2)*. Let $a \neq 0$, b, $c \neq 0$ be elements from $\mathbb{F}_{q^m}$, and let us define $a\boldsymbol{x}^* + b \stackrel{def}{=} (ax_1^* + b, \ldots, ax_n^* + b)$ and $c\boldsymbol{y}^* \stackrel{def}{=} (cy_1^*, \ldots, cy_n^*)$. Then $a\boldsymbol{x}^* + b$ and $c\boldsymbol{y}^*$ are also solution.*

The general case is still unsolved. However, the results of [Dür87] basically show that the we have at least one degree of freedom for Y_i and three degrees of freedom for the X_i in the system (9.2). It is quite helpful to introduce the symbol ∞ in order to consider projective alternant codes over $\overline{\mathbb{F}}_{q^m} \stackrel{\text{def}}{=} \mathbb{F}_{q^m} \cup \{\infty\}$. We will now require to focus on special kind of permutations. Let a, b, c, d be elements from $\mathbb{F}_{q^m}$ such that $ad - bc \neq 0$. Let $\psi : \overline{\mathbb{F}}_{q^m} \to \overline{\mathbb{F}}_{q^m}$ be the function defined for any z from $\overline{\mathbb{F}}_{q^m}$ by:

$$\psi(z) \stackrel{\text{def}}{=} \frac{az+b}{cz+d}.$$

The usual rules to evaluate $\psi(z)$ are used. We state now an important proposition where we use the notation $\boldsymbol{x}^\psi \stackrel{\text{def}}{=} \Big(\psi(x_1), \ldots, \psi(x_n)\Big)$ for any $\boldsymbol{x}$ from $(\overline{\mathbb{F}}_{q^m})^n$.

Proposition 13. *Let $\boldsymbol{x}$ be an n-tuple formed by distinct elements of $\overline{\mathbb{F}}_{q^m}$ and let $\boldsymbol{y}$ be an n-tuple of nonzero elements of $\mathbb{F}_{q^m}$. Let us set $\boldsymbol{x}^\psi \stackrel{def}{=} (\psi(x_1), \ldots, \psi(x_n))$ and $\boldsymbol{y}' \stackrel{def}{=} (y'_1, \ldots, y'_n)$ with $y'_j \stackrel{def}{=} (cx_j + d)^{r-1} y_j$. Let r be an integer $\geqslant 1$. We then have the equality:*

$$\mathscr{A}_r(\boldsymbol{x}^\psi, \boldsymbol{y}') = \mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y}).$$

At first glance, the degree of freedom should be less for Goppa codes. Indeed, there is an additional crucial constraint for binary Goppa codes. A solution must verify $Y_i = \gamma(X_i)^{-1}$ for some polynomial with coefficients from $\mathbb{F}_{q^m}$ and of degree r. Surprisingly, we can keep the same degree of freedom by considering a slight change. Indeed, if we consider the extended Goppa code $\tilde{\mathscr{G}}_r(\boldsymbol{x}, \gamma)$ with $\gamma(z) = \sum_{i=0}^r \gamma_i z^i$ then we have the following result. We recall that by convention $\gamma(\infty) = \gamma_r$.

Proposition 14. *Let us define $\gamma^\psi(z) \stackrel{def}{=} (cz+d)^r \gamma\big(\psi(z)\big) = \displaystyle\sum_{i=0}^{r} \gamma_i (az+b)^i (cz+d)^{r-i}$. We have $\tilde{\mathscr{G}}_r(\boldsymbol{x}, \gamma^\psi) = \tilde{\mathscr{G}}_r(\boldsymbol{x}^\psi, \gamma)$.*

These results enable to reduce the number of unknowns by attributing random values to one Y_i and three X_i. However, this highly structured system prompts us to study methods for solving it. It is well-known that the existing algorithms dealing with polynomial systems are essentially based on Gröbner bases [Buc65, CLO01, Fau99, Fau02]. It is therefore natural to understand what can be done with this kind of algorithms. Furthermore, we will see that we can extract an affine bilinear system from (9.2). This is especially interesting since the complexity of solving such equations is well mastered in the generic case [FSS11]. On the other hand, the question remains open for general bihomogeneous equations.

We briefly recall basic facts about the complexity of computing Gröbner bases [Buc65, CLO01, Fau99, Fau02].

9.2 General Complexity of Gröbner Bases

The complexity of computing Gröbner bases depends on the so-called *degree of regularity*. This is roughly the maximal degree of the polynomials appearing during the computation of a Gröbner bases with respect to a Degree Reverse Lexicographical order. This degree of regularity, denoted by D_{reg} in what follows, is the key parameter. Indeed, the cost of computing a Gröbner basis is polynomial in D_{reg}. Precisely, the complexity of computing a Gröbner basis with F_5 – most efficient algorithm so far – is:

$$\mathcal{O}\left(\binom{N+D_{\mathrm{reg}}}{D_{\mathrm{reg}}}^{\omega}\right) \tag{9.4}$$

with $2 < \omega \leqslant 3$ being the "linear algebra constant", and N being the number of variables. This basically correspond to the complexity of reducing a matrix of size $\binom{N+D_{\mathrm{reg}}}{D_{\mathrm{reg}}}$. The degree of regularity depends on the number of variables, the number of equations, the degree of the considered system and the system itself. In general, predicting its value for a given system is a hard problem. However, the behavior of the degree of regularity is well understood [Bar04, BFS04, BFS02, BFSY05] for semi-regular (*resp.* regular) systems (*i.e.* algebraic definition of random systems). In particular, we have:

Proposition 15 (Macaulay's bound)**.** *The degree of regularity of a square (same number of equations and variables) regular quadratic system over $\mathbb{F}_q[\boldsymbol{X}]$ is $1 + n_X$ where n_X is the number of variables in the set of variables $\boldsymbol{X}$.*

It is worth mentioning that however this bound no longer holds if the system has some type of structure. A particular example is that of affine bilinear system [FSS11]. This type of algebraic is also interesting in our context because it is possible to exhibit such a system. A first important result of [FSS11] is that F_5 [Fau02] is already optimal for "generic" (random) affine bilinear systems, *i.e.* all reductions to zero are removed by the F_5 criterion. Another fundamental result is that the degree of regularity of a square generic affine bilinear system is much smaller than the degree of regularity of a generic system.

Proposition 16 ([FSS11])**.** *The degree of regularity of a square generic affine bilinear system in $\boldsymbol{X}$ and $\boldsymbol{Y}$ is bounded by $1 + \min(n_X, n_Y)$ where n_X and n_Y are the number of variables in $\boldsymbol{X}$ and $\boldsymbol{Y}$ respectively.*

This bound is sharp for a generic square affine bilinear system and is much better than the usual Macaulay's bound that we would obtain for a similar square quadratic system with $n_X + n_Y$ quadratic equations in $n_X + n_Y$ variables because generally, it holds that:

$$1 + \min(n_X, n_Y) \ll 1 + n_X + n_Y.$$

One case of particular interest is when the minimum is constant because the computing becomes polynomial in time. It appears that the matrices occurring during the matrix version of F_5 can be divided into smaller matrices thanks to the bilinear structure [FSS11]. To estimate precisely the complexity of solving, we recall the following definitions.

Definition 25 ([FSS11])**.** *Let d_1, d_2 be positive integers.*

- *An ideal is* bihomogeneous *if there exists a set of bihomogeneous generators. The vector space of bihomogeneous polynomials of bi-degree (d_1, d_2) in a polynomial ring R will be denoted by R_{d_1,d_2}. It holds that:*

$$\dim(R_{d_1,d_2}) = \binom{d_1 + n_X}{d_1}\binom{d_2 + n_Y}{d_2}.$$

- *If $\mathcal{I}$ is a bihomogeneous ideal, then I_{d_1,d_2} is the vector space $I \cap R_{d_1,d_2}$.*

- *Let $\mathcal{I}$ be a bihomogeneous ideal of R. The* Hilbert bi-series *is defined by*

$$\mathrm{HS}_{\mathcal{I}}(t_1, t_2) = \sum_{(d_1,d_2)\in\mathbb{N}^2} \dim(R_{d_1,d_2}/I_{d_1,d_2}) t_1^{\alpha} t_2^{\beta}.$$

The Hilbert bi-series defined below allows to study precisely the complexity of a Gröbner basis computation. For (bi-regular) bilinear systems, [FSS11] provides an explicit form of the bi-series.

Theorem 2. *Let $I_m = \langle f_1, \ldots, f_m \rangle \subset R$ be a (bi-regular) bilinear ideal with $m \leq n_{X'} + n_{Y'}$. Then*

$$\mathrm{HS}_{I_m}(t_1, t_2) = \frac{(1-t_1t_2)^m + N_{n_{X'}+1}(t_1,t_2) + N_{n_{Y'}+1}(t_1,t_2)}{(1-t_1)^{n_{X'}+1}(1-t_2)^{n_{Y'}+1}},$$

where

$$N_n(t_1,t_2) = t_1t_2(1-t_2)^n \sum_{\ell=1}^{m-n} (1-t_1t_2)^{m-n-\ell} \left[1 - (1-t_1)^\ell \sum_{k=1}^{n} t_1^{n-k} \binom{\ell+n-k-1}{n-k}\right].$$

From this, we can estimate the size of the matrices occurring at degree D during the matrix-F_5 on a bilinear systems. Indeed, these matrices are of size:

$$\Big(\dim(R_{d_1,d_2}) - [t_1^{d_1}t_2^{d_2}]\mathrm{HS}(t_1,t_2)\Big) \times \dim(R_{d_1,d_2}),$$

with (d_1, d_2) such that $d_1 + d_2 = D$ where $1 \leqslant d_1, d_2 \leqslant D-1$ and $[t_1^{d_1}t_2^{d_2}]\mathrm{HS}(t_1,t_2)$ stands for the coefficient of the term $t_1^{d_1}t_2^{d_2}$ in the Hilbert bi-serie $\mathrm{HS}(t_1,t_2)$. As pointed out, these results hold for a bilinear system. For an affine bilinear, this can be considered as a good (*i.e.* first order) approximation. The idea is that we have to "bi-homogenize" the affine bilinear system which corresponds to add some columns.

9.3 Extraction of a Bilinear System

The method for extracting an (affine) bilinear system from (9.2) works as follows. The first fundamental remark is that k linear equations in the n variables of the block $\boldsymbol{Y}$ occur in (9.2). This implies that all the variables $\boldsymbol{Y}$ can be expressed in terms of $n_{Y'} \geqslant n-k$ variables. We will always assume that the variables $\boldsymbol{Y}'$ only refer to these $n_{Y'}$ *free* variables. The system (9.2) is rewritten only in function of $\boldsymbol{X}$ and $\boldsymbol{Y}'$ *i.e.*, the variables of $\boldsymbol{Y} \setminus \boldsymbol{Y}'$ are substituted by linear combinations involving only variables of $\boldsymbol{Y}'$. For simplicity, we keep calling this new system by (9.2). The number $n_{\boldsymbol{Y}'}$ of variables of $\boldsymbol{Y}'$ is of course $n-k$ and the total number of equations becomes $(r-1)k$. Eventually, this system is still bihomogeneous with bi-degree $(j,1)$ with $1 \leqslant j \leqslant r-1$.

Next, the polynomial system being naturally overdetermined, some equations can be removed. It makes sense then to consider the set of equations whose degree in the variables of $\boldsymbol{X}'$ is a power of q *i.e.* equations of bi-degree $(q^j, 1)$. We obtain another subsystem of having $k \log_q r$ equations. This system is almost a bilinear system over $\mathbb{F}_q^m$. But if each variable X_i with $1 \leqslant i \leqslant n$ is viewed as m q-ary variables say $(X_{i,1}, \ldots, X_{i,m})$ by fixing a basis of $\mathbb{F}_{q^m}$ treated as a $\mathbb{F}_q$-vector space. This new set of variables is denoted by $\boldsymbol{X}'$. This decomposition blows up the number of unknowns because we get m times as many as variables for $\boldsymbol{X}'$. But the number of variables of $\boldsymbol{Y}'$ has not changed. The resulting system denoted by $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ is now $\mathbb{F}_q$-linear with $n_{X'} = mn$ unknowns $\boldsymbol{X}'$ and $n_{Y'} = n-k$ unknowns $\boldsymbol{Y}'$. The number of equations is $k \log_q r$. Since $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ is a bilinear system it is tempting to claim the following conjecture.

Conjecture 1. *The degree of regularity of* $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ *is less than* $1 + \min(n_{X'}, n_{Y'})$.

If the claim is ever true then we would get a degree of regularity that is less then $(n-k) = n(1-R)$. Let us emphasize that the bound [FSS11] only considered systems whose number of equations does not exceed the number of variables. But in our context, by posing $R \stackrel{\text{def}}{=} k/n$ and remarking that $m \leqslant \log_q n$ and $r = \frac{n-k}{m}$, the number of equations is $k \log_q r \geqslant Rn \log_q \left((1-R)\frac{n}{\log_q n}\right)$ while the number of unknowns is $n_{X'} + n_{Y'} = (m+1)n - k \leqslant n(1 - R + \log_q n)$. So, it may happen for some range of parameters that $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ has less equations than the number of variables.

The situation is different for the variants based on Φ-invariant alternant codes, namely quasi-cyclic and quasi-dyadic variants of [BCGO09, MB09]. The constraint on the entries of the public matrix can be directly translated into new linear equations. This allows to reduce considerably the number of variables in (9.2). The reduction is so drastic that it becomes possible to efficiently solve it leading to practical key-recovery attacks. Furthermore, because of the high number of equations, a theoretical explanation can be given. This is based on the following estimation of the space/time complexity for computing a Gröbner basis of $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$.

Proposition 17. *Let us set $D \stackrel{def}{=} \min(n_{X'}, n_{Y'}) + 1$. Assuming Conjecture 1, the time complexity T of computing a DRL-Gröbner basis G_{DRL} of* $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ *is bounded from above by*

$$T = \sum_{\substack{d_1+d_2=D \\ 1 \leqslant d_1, d_2 \leqslant D-1}} \Big(\dim(R_{d_1,d_2}) - [t_1^{d_1}t_2^{d_2}]\mathrm{HS}(t_1,t_2)\Big)^{\omega-1} \dim(R_{d_1,d_2})$$

with $2 \leqslant \omega \leqslant 3$.

9.4 Key-Recovery Attack Against Φ-Invariant Alternant Variants

9.4.1 Quasi-Cyclic Alternant Variant

The scheme presented in [BCGO09] (See Chapter 8) suggests to use block matrices where each block is a circulant matrix. The public code $\mathscr{C}$ is a quasi-cyclic alternant code defined over a field $\mathbb{F}_q = \mathbb{F}_{2^s}$ also considered as a subfield of $\mathbb{F}_{q^m}$ for some integer $m \geqslant 2$. Let α be a primitive element of $\mathbb{F}_{q^m}$, ℓ and N_0 be integers such that $q^m - 1 = N_0\ell$ and eventually β be an element of $\mathbb{F}_{q^m}$ of order ℓ.

The public code $\mathscr{C}$ is obtained[3] from an $r \times n$ parity-check matrix $\boldsymbol{Q}$ over $\mathbb{F}_{q^m}$ which is the juxtaposition of n_0 ($n = \ell n_0$) circulant matrices $\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_{n_0}$ of size $r \times \ell$. Each $\boldsymbol{Q}_{b+1} = \left(Q_{i,j}^{(b)}\right)$ with $0 \leqslant b \leqslant n_0 - 1$, $0 \leqslant i \leqslant r-1$ and $0 \leqslant j \leqslant \ell - 1$ is given by (See Equation (8.3)):

$$Q_{i,j}^{(b)} = \gamma_b \beta^{(d_b+j)e} \left(\alpha^{w_b}\beta^{d_b+j}\right)^i \tag{9.5}$$

where γ_b is a nonzero element of $\mathbb{F}_{q^m}$, d_b is an integer of $\{0, \ldots, \ell-1\}$, e is an integer of $\{0, \ldots, \ell-1\}$ and the w_b's are distinct integers of $\{0, \ldots, N_0 - 1\}$. $\mathscr{C}$ is therefore an alternant code $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ of order r associated to $\boldsymbol{x} = (x_0, \ldots, x_{n-1})$ and $\boldsymbol{y} = (y_0, \ldots, y_{n-1})$ which satisfy for any j in $\{0, \ldots, \ell-1\}$ the following linear relations:

$$x_{b\ell+j} = \alpha^{w_b}\beta^{d_b+j} \tag{9.6}$$

$$y_{b\ell+j} = \gamma_b \beta^{(d_b+j)e}. \tag{9.7}$$

It can be checked that $\mathscr{C}$ has a public generating matrix $\boldsymbol{G}$ which is block circulant of size $k \times n$ with k of the form $k = k_0\ell$ for some integer k_0 (recall that $k \geq n - rm$).

We present now an algebraic attack that recovers $\boldsymbol{x}$ and $\boldsymbol{y}$ by setting up the algebraic system (9.2). This would also give a system with $2n$ unknowns. We can obtain a huge reduction of the number of unknowns by using Equations (9.6) and (9.7) which induce some linear relations between the x_i's and the y_i's. Indeed, we can deduce that:

$$x_{b\ell+j} = x_{b\ell}\beta^j \tag{9.8}$$

$$y_{b\ell+j} = y_{b\ell}\beta^{je}, \tag{9.9}$$

for any integers b and j such that $0 \leqslant b \leqslant n_0 - 1$ and $0 \leqslant j \leqslant \ell - 1$. Let us recall that e verifies $0 \leqslant e \leqslant \ell - 1$. Since in the cases considered in [BCGO09], ℓ is small (less than 100), one may assume that:

Assumption 2. *The secret integer e such that $0 \leqslant e \leqslant \ell - 1$ is known.*

Another important assumption we will make is about the representation of the field $\mathbb{F}_{q^m}$. We can fairly consider α and β as known quantities because any other representation of $\mathbb{F}_{q^m}$ is equivalent to the one fixed by the designer.

Assumption 3. *The primitive element α from $\mathbb{F}_{q^m}$ and the element β of order ℓ are known.*

These two assumptions simplify the description of the algebraic system. By setting up the unknown X_b for $x_{b\ell}$ and Y_b for $y_{b\ell}$ we obtain the following algebraic system.

Proposition 18. *Let $\boldsymbol{G} = (g_{i,j})$ be the $k \times n$ public generator matrix with $k = k_0\ell$ and $n = n_0\ell$. The unknowns $X_0, \ldots, X_{n_0-1}$ and $Y_0, \ldots, Y_{n_0-1}$ satisfy the following polynomial equation obtained for any integers w and i such that $0 \leqslant w \leqslant r-1$ and $0 \leqslant i \leqslant k-1$:*

$$\sum_{b=0}^{n_0-1} g'_{i,b,w} Y_b X_b^w = 0 \quad \textit{with} \quad g'_{i,b,w} \stackrel{\text{def}}{=} \sum_{j=0}^{\ell-1} g_{i,b\ell+j}\beta^{j(e+w)}. \tag{9.10}$$

Proof. We first observe that $\displaystyle\sum_{j=0}^{n-1} g_{i,j}y_j x_j^w = \sum_{b=0}^{n_0-1}\sum_{j=0}^{\ell-1} g_{i,b\ell+j}\ y_{b\ell+j}x_{b\ell+j}^w$, or also:

$$\sum_{j=0}^{n-1} g_{i,j}y_j x_j^w = \sum_{b=0}^{n_0-1}\sum_{j=0}^{\ell-1} g_{i,b\ell+j}y_{b\ell}x_{b\ell}^w\beta^{je}\beta^{jw} = \sum_{b=0}^{n_0-1}\left(\sum_{j=0}^{\ell-1} g_{i,b\ell+j}\beta^{je}\beta^{jw}\right) y_{b\ell}x_{b\ell}^w.$$

By setting X_b for $x_{b\ell}$ and Y_b for $y_{b\ell}$ we obtain the aforementioned system. □

The first important consequence by adding new linear relations is the number of unknowns becomes n_0 for the X_i's and n_0 for the Y_i's. Theoretically by Proposition 12, we would be able to fix two variables, say X_0 and X_1, and one variable Y_j, for instance Y_0, to arbitrary values as long as $X_0 \neq X_1$ and $Y_0 \neq 0$. However, if we do it, we then lose the linear relations between the x_i's given in (9.8). Therefore we can only fix one X_i and one Y_i as stated in Proposition 19 that is straightforward to prove.

Proposition 19. *For any $a \neq 0$ and $c \neq 0$ from $\mathbb{F}_{q^m}$, if $\boldsymbol{x}$ and $\boldsymbol{y}$ are solution to* (9.10) *then $a\boldsymbol{x}$ and $b\boldsymbol{y}$ are also solution.*

[3] By keeping the notation of Chapter 8, the integer $r \stackrel{\text{def}}{=} 2t$.

Hence, we can fix one X_i and one Y_i so that the total number of unknowns is $2(n_0 - 1)$. Let us remark that the resulting system becomes affine bihomogeneous. As for the number of equations, one would believe that for each w from $\{0, \ldots, r-1\}$ we would get k equations. But in reality, there are many redundant equations in Proposition 18. This comes from the block circulant form of $\boldsymbol{G}$. More exactly, $\boldsymbol{G}$ has the following form $\boldsymbol{G} = (\boldsymbol{G}_{i,j})$ with $\boldsymbol{G}_{i,j}$ is an $\ell \times \ell$ circulant matrix for any i and j such that $0 \leqslant i \leqslant k_0 - 1$ and $0 \leqslant j \leqslant n_0$. Using that fact, we therefore have $g_{i\ell+u,b\ell+j} = g_{i\ell,b\ell+((j-u) \mod \ell)}$ for all u and i such that $0 \leqslant u \leqslant \ell - 1$ and $0 \leqslant i \leqslant k_0 - 1$. Hence, we can deduce that:

$$\begin{aligned} g'_{i\ell+u,b,w} &= \sum_{j=0}^{\ell-1} g_{i\ell+u,b\ell+j}\beta^{j(e+w)} = \sum_{j=0}^{\ell-1} g_{i\ell,b\ell+((j-u) \mod \ell)}\beta^{j(e+w)} \\ &= \sum_{j=0}^{\ell-1} g_{i\ell,b\ell+j}\beta^{j(e+w)}\beta^{u(e+w)} = g'_{i\ell,b,w}\beta^{u(e+w)} \end{aligned}$$

We used the fact $\beta^{\ell(e+w)} = 1$ since β is of order ℓ. So for any i such that $0 \leqslant i \leqslant k_0 - 1$, when u describes $\{0, \ldots, \ell - 1\}$, the equations $\sum_{b=0}^{n_0-1} g'_{i\ell+u,b,w} Y_b X_b^w = 0$ are all linearly dependant. The only equations that should be considered are those obtained with i and w such that $0 \leqslant i \leqslant k_0 - 1$ and $0 \leqslant w \leqslant r - 1$ and defined by:

$$\sum_{b=0}^{n_0-1} g'_{1+i\ell,b,w} Y_b X_b^w = 0. \tag{9.11}$$

This means that instead of having rk equations we have only $rk/\ell = k_0 r$ polynomial equations. In particular, the $k/\ell = k_0$ linear equations involving only the unknowns Y_b enable to express k_0 unknowns in function of $n_0 - 1 - k_0$ *free* variables. We denote these remaining unknowns by $\boldsymbol{Y}'$. The variables $\boldsymbol{Y} \setminus \boldsymbol{Y}'$ do not appear anymore henceforth in the system. This study enables to prove the following proposition.

Proposition 20. *It is possible to derive from the polynomial system* (9.10) *an affine bihomogeneous polynomial system that has* $n_{Y'} \stackrel{def}{=} n_0 - 1 - k_0$ *unknowns* Y'_i *and* $n_X = n_0 - 1$ *unknowns* X_i*. The only possible bi-degrees are* $(w, 1)$ *with* $1 \leqslant w \leqslant r - 1$*. Finally, the number of different equations of bi-degree* $(w, 1)$ *is* k_0 *so that the total number is* $(r - 1)k_0$*.*

9.4.2 Quasi-Dyadic Goppa Code Variant

The cryptosystem presented in [MB09] considers particular alternant codes called *quasi-dyadic* Goppa codes. A detailed description of the key generation is given in Chapter 8. We only provide important facts that are useful for recovering the private key. An important fact to know about $\boldsymbol{G}$ is that it is a $k \times n$ matrix over $\mathbb{F}_q$ such that $n = n_0\ell$ and $k \geqslant n - m\ell$ where n_0, ℓ are given integers. We now state an important result.

Proposition 21. *The code defined by the public generator matrix* $\boldsymbol{G}$ *is an alternant code* $\mathscr{A}_\ell(\boldsymbol{x}, \boldsymbol{y})$ *where for any* $0 \leqslant j \leq n_0 - 1$ *and* $0 \leqslant i, i' \leqslant \ell - 1$*, we have the following equations:*

$$\begin{cases} y_{j\ell+i} &= y_{j\ell} \\ x_{j\ell+i} + x_{j\ell} &= x_i + x_0 \\ x_{j\ell+(i\oplus i')} &= x_{j\ell+i} + x_{j\ell+i'} + x_{j\ell} \end{cases} \tag{9.12}$$

where $\oplus$ *is the bitwise exclusive-or on the binary representation of the indices.*

The cryptanalysis of the system consists in defining n_0 unknowns $Y_0, \ldots, Y_{n_0-1}$ that play the role of the y_j's and n unknowns $X_0, \ldots, X_n$ that represent the x_j's. We know specify the system of equations that we obtain directly from Proposition 21.

Proposition 22. *For any* w*,* j*,* i *and* i' *such that* $0 \leqslant w \leqslant \ell - 1$*,* $0 \leqslant j \leqslant n_0 - 1$ *and* $1 \leqslant i, i' \leqslant \ell - 1$*, we have:*

$$\begin{cases} \displaystyle\sum_{j=0}^{n_0-1} Y_j \sum_{l=0}^{\ell-1} g_{i,j\ell+l} X_{j\ell+l}^w &= 0 \\ X_{j\ell+i} + X_{j\ell} + X_i + X_0 &= 0 \\ X_{j\ell+(i\oplus i')} + X_{j\ell+i} + X_{j\ell+i'} + X_{j\ell} &= 0 \end{cases} \tag{9.13}$$

It is possible to simplify (9.13) by observing, thanks to the third equation, that actually many variables X_i's can be expressed in function of some few variables, namely X_{2^j} with $0 \leqslant j \leqslant \log_2(\ell - 1)$ and X_b with $0 \leqslant b \leqslant n_0 - 1$.

Corollary 2. *For any* $1 \leqslant i \leqslant \ell - 1$*, if we write the binary decomposition of* $i = \sum_{j=0}^{\log_2(\ell-1)} \eta_j 2^j$ *then* $X_i = X_0 + \sum_{j=0}^{\log_2(\ell-1)} \eta_j (X_{2^j} + X_0)$.

We will denote by $\boldsymbol{X}'$ the variables X_i that serve to express all the others. We are also able to provide the exact number of unknowns we can fix to arbitrary values. Indeed, thanks to the following lemma, one Y_i can be chosen arbitrarily provided that $Y_i \neq 0$, and two X_i and X_j with $i \neq j$ can be fixed to two distinct random values.

Lemma 1. *If $\boldsymbol{x}$ and $\boldsymbol{y}$ are solution of* (9.13) *then for any $a \neq 0$, b, $c \neq 0$ from $\mathbb{F}_{q^m}$, $a\boldsymbol{x} + b$ and $c\boldsymbol{y}$ are also solution.*

Proof. The only fact to prove is that $(x_0 + b, \ldots, x_{n-1} + b)$ is also a solution of the last two equations in (9.13). It is readily checked since $\mathbb{F}_{q^m}$ is of charateristic two. □

We can now completely give the effective number of equations after elimination of redundant equations.

Proposition 23. *It is possible to derive from the polynomial system* (9.13) *another polynomial system that has $n_{Y'} = m - 1$ unknowns Y'_i and $n_{X'} = n_0 - 2 + \log_2(\ell)$ unknowns X'_i. Furthermore, it has $(\ell - 1)\ell(n_0 - m)$ polynomial equations only involving the terms of the form $Y_i X_i^w$ with $1 \leqslant w \leqslant \ell - 1$.*

Proof. The number of variables Y_j is $(n_0 - 1)$ since we can choose $Y_0 = 1$. As for the variables X_j, we observe that they can all be expressed only in function of X_{2^j} and $X_{i\ell}$ with $0 \leqslant j \leqslant \log_2(\ell - 1)$ and $0 \leqslant i \leqslant n_0 - 1$. So the number of unknowns X_j is $\log_2(\ell - 1) + 1 + n_0 - 2$ since we can fix two different arbitrary values for two variables, say X_0 and X_ℓ (Lemma 1). Using the fact that $\log_2(\ell - 1) = \log_2(\ell) - 1$ since ℓ is a power of 2, we get the claimed number of unknowns. Furthermore, because of the dyadicity of $\boldsymbol{G}$, the equations obtained with $w = 0$ are identical when $\boldsymbol{g}$ describes all the rows of a dyadic block of $\boldsymbol{G}$. This does not appear when $w > 1$. So we have $k/\ell = n_0 - m$ linear equations that involve the Y_i's and $(\ell - 1)k = (\ell - 1)\ell(n_0 - m)$ polynomial equations that contain variables of the form $Y_i X_i^w$ where $1 \leqslant w \leqslant \ell - 1$. □

9.4.3 Strategy for Solving the Algebraic System

The method to solve the systems that we derived from [BCGO09, MB09] consists in extracting the bilinear subsystem of polynomial equations of bi-degree $(q^j, 1)$ as explained in Section 9.3. For the sake of simplicity, we denote it by $\mathsf{McE}_{k,n,r}(\mathbf{X}', \mathbf{Y}')$. The ultimate goal of the attack is to compute the variety (*i.e.* set of solutions) $\mathcal{V}$ associated to $\mathsf{McE}_{k,n,r}(\mathbf{X}', \mathbf{Y}')$. As soon as we have a DRL-Gröbner basis G_{DRL} of $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$, the variety can be obtained in $\mathcal{O}\big((\#\mathcal{V})^\omega\big)$ thanks to a change of ordering algorithm [FGLM93]. Therefore, one has to be sure that the variety $\mathcal{V}$ has few elements. In particular, it is crucial to remove parasite solutions corresponding to $X_i = X_j$ an $Y_j = 0$ for instance. A classical way to do that is to introduce new variables u_{ij} and v_i and add equations of the form:

$$u_{ij}(X_i - X_j) + 1 = 0 \quad \text{and} \quad v_i Y_i + 1 = 0.$$

In practice, we have not added all these equations but only few of them (namely 4 or 5). The reason is that we do not want to add too many new variables. However, these equations and variables can be added to $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ whilst keeping the affine bi-linear structure. To do so, we have to add the v_i to the block $\boldsymbol{X}'$, and the variables u_{ij} to the block $\boldsymbol{Y}'$. So, as we add only few new variables, the complexity of solving $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ with these new constraints is essentially similar to Proposition 17.

Eventually, thanks to the works of [FSS11] on the solving of bilinear systems, we can revisit the strategy previously we used in [FOPT10a]. The approach is as efficient than the "ad-hoc" technique proposed initially in [FOPT10a] but with the advantage that its complexity can be more easily analyzed.

9.4.4 Comparison with Theoretical Results

Wee give the experimental results we obtained in [FOPT10a] for cryptanalyzing the schemes [BCGO09] and [MB09]. We also include a bound on theoretical complexity T_{theo} of computing a Gröbner bases of $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ using $\omega = 2$ as given in Proposition 17 for comparison. This is a optimistic but in the other hand, we are not using the fact that the systems are overdetermined and secondly, we have only considered a subsystem of $\mathsf{McE}_{k,n,r}(\mathbf{X}', \mathbf{Y}')$.

One can see that the theoretical bound T_{theo} provides a reasonable explanation regarding the efficiency of the attack presented in [FOPT10a]. In particular, it is important to remark that the hardness of the attack seems linked to $d \stackrel{\text{def}}{=} \min(n'_X, n'_Y)$. The complexity of the attack clearly increases with this quantity. For the design of future compact variants of McEliece, d should not be too small. Regarding the current state of the art, it is difficult to provide an exact value. Very roughly speaking, $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ can be considered as hard as solving a random (overdetermined) algebraic system with $d = \min(n_{X'}, n_{Y'})$ equations over a big field. With this in mind, we can say that any system with $d \leqslant 20$ should be within the scope of an algebraic attack.

Let us remark that another phenomena can occur. In the particular case of binary quasi-dyadic codes, the Gröbner basis of $\mathsf{biMcE}(\boldsymbol{X}', \boldsymbol{Y}')$ can be easily computed, but the variety associated is too large. This is due to the fact that the Gröbner basis is "trivial" (reduced to one equation) and does not provide then enough information. This is due to the fact that we have used only a subset of the equations of bi-degree $(q^j, 1)$. So, an open question is how we can use cleverly all the equations of $\mathsf{McE}_{k,n,r}(\mathbf{X}', \mathbf{Y}')$ in the binary case.

Table 9.1: Cryptanalysis results for [BCGO09] ($m = 2$).

Challenge	q	ℓ	n_0	$n_{Y'}$	Security	$n_{X'}$	Time (Op., Mem.)	T_{theo}
A_{16}	2^8	51	9	3	80	8	0.06 sec ($2^{18.9}$ op, 115 Meg)	2^{17}
B_{16}	2^8	51	10	3	90	9	0.03 sec ($2^{17.1}$ op, 116 Meg)	2^{18}
C_{16}	2^8	51	12	3	100	11	0.05 sec ($2^{16.2}$ op, 116 Meg)	2^{20}
D_{16}	2^8	51	15	4	120	14	0.02 sec ($2^{14.7}$ op, 113 Meg)	2^{26}
A_{20}	2^{10}	75	6	2	80	5	0.05 sec ($2^{15.8}$ op, 115 Meg)	2^{10}
B_{20}	2^{10}	93	6	2	90	5	0.05 sec ($2^{17.1}$ op, 115 Meg)	2^{10}
C_{20}	2^{10}	93	8	2	110	7	0.02 sec ($2^{14.5}$ op, 115 Meg)	2^{11}
QC_{600}	2^8	255	15	3	600	14	0.08 sec ($2^{16.6}$ op, 116 Meg)	2^{21}

Table 9.2: Cryptanalysis results for [MB09].

Challenge	q	$n_{Y'}$	ℓ	n_0	Security	$n_{X'}$	Time (Op., Mem.)	T_{theo}
Table 2	2^2	7	64	56	128	59	$1,776.3$ sec ($2^{34.2}$ op, 360 Meg)	2^{65}
Table 2	2^4	3	64	32	128	36	0.50 sec ($2^{22.1}$ op, 118 Meg)	2^{29}
Table 2	2^8	1	64	12	128	16	0.03 sec ($2^{16.7}$ op, 35 Meg)	2^8
Table 3	2^8	1	64	10	102	14	0.03 sec ($2^{15.9}$ op, 113 Meg)	2^8
Table 3	2^8	1	128	6	136	11	0.02 sec ($2^{15.4}$ op, 113 Meg)	2^7
Table 3	2^8	1	256	4	168	10	0.11 sec ($2^{19.2}$ op, 113 Meg)	2^7
Table 5	2^8	1	128	4	80	9	0.06 sec ($2^{17.7}$ op, 35 Meg)	2^6
Table 5	2^8	1	128	5	112	10	0.02 sec ($2^{14.5}$ op, 35 Meg)	2^7
Table 5	2^8	1	128	6	128	11	0.01 sec ($2^{16.6}$ op, 35 Meg)	2^7
Table 5	2^8	1	256	5	192	11	0.05 sec ($2^{17.5}$ op, 35 Meg)	2^7
Table 5	2^8	1	256	6	256	12	0.06 sec ($2^{17.8}$ op, 35 Meg)	2^7
Dyadic_{256}	2^4	3	128	32	256	37	7.1 sec ($2^{26.1}$ op, 131 Meg)	2^{29}
Dyadic_{512}	2^8	1	512	6	512	13	0.15 sec ($2^{19.7}$ op, 38 Meg)	2^8

Chapter 10

A Distinguisher For High-Rate McEliece Cryptosystems[1]

10.1 Introduction

We investigate the difficulty of the Goppa Code Distinguishing (GCD) problem which first appeared in [CFS01]. This is a decision problem that aims at recognizing a generator matrix of a binary Goppa code from a randomly drawn binary matrix. Up to now, it is assumed that no polynomial time algorithm exists that distinguishes a generator matrix of a Goppa code from a randomly picked generator matrix.

The main motivation for introducing the GCD problem is to relate the security of the McEliece public-key cryptosytem [McE78] to the difficulty of decoding a random linear code. Since its apparition, this cryptosystem has withstood many attacks and after more than thirty years now, it still belongs to the very few unbroken public key cryptosystems. This situation substantiates the claim that inverting the encryption function, and in particular recovering the private key from public data, is intractable. The classical methods that are dedicated to inverting the McEliece encryption function without finding a trapdoor all resort to the use of the best general decoding algorithms [LB88, Leo88, Ste88, CC98, BLP08]. All these algorithms, whose time complexity is exponential, attempt to solve the long-standing problem of decoding random linear code [BMvT78]. They also assume (implicitly or explicitly) that there does not exist an algorithm that is able to decode more efficiently McEliece public keys. Let us note that if ever such an algorithm exists, it would permit to solve the GCD problem.

On the other hand, no significant breakthrough has been observed with respect to the problem of recovering the private key [Gib91, LS01]. This has led to claim that the generator matrix of a binary Goppa code does not disclose any visible structure that an attacker could exploit. This is strengthened by the fact that Goppa codes share many characteristics with random code. For instance they asymptotically meet the Gilbert-Varshamov bound, they have a trivial permutation group, *etc.* Hence, the hardness of the GCD problem has become a classical belief, and as a consequence, a *de facto* assumption to prove the semantic security in the standard model (IND-CPA in [NIKM08] and IND-CCA2 in [DMQN09]), and the security in the random oracle model against existential forgery [CFS01, Dal07] of the signature scheme [CFS01].

We present a deterministic polynomial-time distinguisher for high rate codes. This kind of codes are mainly encountered with the public keys of the signature scheme [CFS01]. It is based on the algebraic attack developped against compact variants of McEliece [FOPT10a]. In this approach, the key-recovery problem is transformed into the one of solving an algebraic system. By using a linearization technique, we are able to derive a linear system whose rank is different from what one would expect. More precisely, we observe experimentally that this *defect* in the rank is directly related to the type of codes. We provide explicit formulas for "generic" random, alternant, and Goppa code over any alphabet. We performed extensive experiments to confirm that the formulas are accurate. Eventually, we prove the formula in the random case and give explanations in the case of alternant codes over any field and binary Goppa codes. We insist on the fact that the existence of our distinguisher does not undermine the security of primitives based on Goppa codes, but basically, it proves that the GCD assumption is false for some parameters.

The chapter is organized as follows. In Section 10.2, we introduce the algebraic system that any McEliece cryptosystem must satisfy. In Section 10.3, we construct a linear system deduced from the algebraic system. This defines an algebraic distinguisher. We then provide explicit formulas that predicts the behavior of the distinguisher coming from heavy experimentations. In Section 10.4, we give a proof of its typical behavior in the random case. In Section 10.5.1 and Section 10.5.2, we give explanations of the formulas for alternant and binary Goppa codes. Lastly, we conclude over the cryptographic implications the distinguisher induces.

10.2 Algebraic Cryptanalysis of McEliece-like Cryptosystems

The McEliece cryptosystem relies on binary Goppa codes which belong to the class of *alternant codes*. It is convenient to describe this class through a parity-check matrix over an extension field $\mathbb{F}_{q^m}$ of $\mathbb{F}_q$ over which the code is defined. For alternant

[1] This chapter is a reproduction of the article published in the ITW 2011 conference [FGUO+11].

codes of length $n \leqslant q^m$, there exists a parity-check matrix with a very special form related to rectangular Vandermonde matrices:

$$\boldsymbol{V}_r(\boldsymbol{x},\boldsymbol{y}) \stackrel{\text{def}}{=} \begin{pmatrix} y_1 & \cdots & y_n \\ y_1x_1 & \cdots & y_nx_n \\ \vdots & & \vdots \\ y_1x_1^{r-1} & \cdots & y_nx_n^{r-1} \end{pmatrix} \tag{10.1}$$

where $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_n)$ are in $(\mathbb{F}_{q^m})^n$.

Definition 26 (Alternant code)**.** *The alternant code of order r over $\mathbb{F}_q$ associated to $\boldsymbol{x} = (x_1, \ldots, x_n) \in (\mathbb{F}_{q^m})^n$ where all x_i's are distinct and $\boldsymbol{y} = (y_1, \ldots, y_n) \in \left(\mathbb{F}_{q^m}^*\right)^n$ denoted by $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ is $\left\{\boldsymbol{c} \in \mathbb{F}_q^n \mid \boldsymbol{V}_r(\boldsymbol{x},\boldsymbol{y})\boldsymbol{c}^T = \boldsymbol{0}\right\}$. The dimension k satisfies $k \geqslant n - rm$.*

A key feature about alternant codes of degree r is the fact that there exists a polynomial time algorithm decoding all errors of weight at most $\frac{r}{2}$ *once a parity-check matrix is given in the form* $\boldsymbol{V}_r(\boldsymbol{x},\boldsymbol{y})$.

Definition 27 (Goppa codes)**.** *The Goppa code $\mathscr{G}(\boldsymbol{x},\gamma)$ over $\mathbb{F}_q$ associated to a polynomial $\gamma(z)$ of degree r over $\mathbb{F}_{q^m}$ and a certain n-tuple $\boldsymbol{x} = (x_1, \ldots, x_n)$ of distinct elements of $\mathbb{F}_{q^m}$ satisfying $\gamma(x_i) \neq 0$ for all $i, 1 \leqslant i \leqslant n$, is the alternant code $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ of order r with y_i being defined by $y_i = \gamma(x_i)^{-1}$.*

Goppa codes, viewed as alternant codes, naturally inherit a decoding algorithm that corrects up to $\frac{r}{2}$ errors. But in the case of *binary* Goppa codes, we can correct twice as many errors (Fact 3). The starting point is the following result given in [MS86, p. 341].

Theorem 3. *A binary Goppa code $\mathscr{G}(\boldsymbol{x},\gamma)$ associated to a Goppa polynomial $\gamma(z)$ of degree r without multiple roots is equal to the alternant code $\mathscr{A}_{2r}(\boldsymbol{x},\boldsymbol{y})$, with $y_i = \gamma(x_i)^{-2}$.*

Fact 3 ([Pat75])**.** *There exists a polynomial time algorithm decoding all errors of Hamming weight at most r in a Goppa code $\mathscr{G}(\boldsymbol{x},\gamma)$ when $\gamma(z)$ has degree r and has no multiple roots, if $\boldsymbol{x}$ and $\gamma(z)$ are* known*.*

We are now able to construct an algebraic system as explained in [FOPT10a] for the McEliece cryptosystem. This algebraic system is the main ingredient of the distinguisher. We assume that the public matrix is a $k \times n$ generator matrix $\boldsymbol{G}$. We have seen that the knowledge of $\boldsymbol{V}_r(\boldsymbol{x}^*,\boldsymbol{y}^*)$ permits to efficiently decode. By definition of $\boldsymbol{G}$, we have:

$$\boldsymbol{V}_r(\boldsymbol{x}^*,\boldsymbol{y}^*)\boldsymbol{G}^T = \boldsymbol{0}.$$

Let $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ be $2n$ variables corresponding to the x_i^*'s and the y_i^*'s. Observe that such x_i^*'s and y_i^*'s are a particular solution [FOPT10a] of the following system:

$$g_{i,1}Y_1X_1^j + \ldots + g_{i,n}Y_nX_n^j = 0 \tag{10.2}$$

with $1 \leqslant i \leqslant k$ and $0 \leqslant j \leqslant r-1$, and where the $g_{i,j}$'s are the entries of the known matrix $\boldsymbol{G}$.

Solving this system boils down to finding an equivalent private key. For compact variants [BCGO09, MB09] of [McE78], additional structures permit to drastically reduce the number of variables allowing to solve (10.2) for a large set of parameters in polynomial-time using dedicated Gröbner bases techniques [FOPT10a]. The general case is currently an open problem.

10.3 A Distinguisher of Alternant and Goppa Codes

We present in this part the algebraic distinguisher which is based on the non-linear system (10.2). We can assume that $\boldsymbol{G} = (g_{ij})$ with $1 \leqslant i \leqslant k$ and $1 \leqslant j \leqslant n$ is in reduced row echelon form over its k first positions *i.e.* $\boldsymbol{G} = (\boldsymbol{I}_k \mid \boldsymbol{P})$ where $\boldsymbol{P} = (p_{ij})$ for $1 \leqslant i \leqslant k$, $k+1 \leqslant j \leqslant$ is the submatrix of $\boldsymbol{G}$ formed by its last $n - k = mr$ columns. We describe now a simple way for solving (10.2). For any $i \in \{1, \ldots, k\}$ and $e \in \{0, \ldots, r-1\}$, we can rewrite (10.2) as

$$Y_iX_i^e = \sum_{j=k+1}^{n} p_{i,j}Y_jX_j^e. \tag{10.3}$$

Thanks to the trivial identity $Y_iY_iX_i^2 = (Y_iX_i)^2$, for all i in $\{1, \ldots, k\}$, we get for all $i \in \{1, \ldots, k\}$:

$$\sum_{j=k+1}^{n} p_{i,j}Y_j \sum_{j=k+1}^{n} p_{i,j}Y_jX_j^2 = \left(\sum_{j=k+1}^{n} p_{i,j}Y_jX_j\right)^2.$$

It is possible to reorder this to obtain:

$$\sum_{j=k+1}^{n-1} \sum_{j'>j}^{n} p_{i,j}p_{i,j'} \left(Y_jY_{j'}X_{j'}^2 + Y_{j'}Y_jX_j^2\right) = 0.$$

We can now linearize this system by letting $Z_{jj'} \stackrel{\text{def}}{=} Y_j Y_{j'} X_{j'}^2 + Y_{j'} Y_j X_j^2$. We obtain a system $\mathcal{L}_{\boldsymbol{P}}$ of k linear equations involving the $Z_{jj'}$'s:

$$\mathcal{L}_{\boldsymbol{P}} \stackrel{\text{def}}{=} \left\{ \sum_{j=k+1}^{n-1} \sum_{j'>j}^{n} p_{i,j} p_{i,j'} Z_{jj'} = 0 \;\middle|\; i \in \{1, \ldots, k\} \right\}. \tag{10.4}$$

To solve this system it is necessary that the number of equations is greater than the number of unknowns *i.e.* $k \geqslant \binom{mr}{2}$ with the hope that the rank of $\mathcal{L}_{\boldsymbol{P}}$ denoted by $\mathrm{rank}(\mathcal{L}_{\boldsymbol{P}})$ is almost equal to the number of variables. Observe that the linear systems (10.4) have coefficients in $\mathbb{F}_q$ whereas solutions are sought in the extension field $\mathbb{F}_{q^m}$. But the dimension D of the vector space solution of $\mathcal{L}_{\boldsymbol{P}}$ does not depend on the underlying field because $\mathcal{L}_{\boldsymbol{P}}$ can always be seen as a system over $\mathbb{F}_{q^m}$. Remark that we obviously have $D = \binom{mr}{2} - \mathrm{rank}(\mathcal{L}_{\boldsymbol{P}})$.

We carried out intensive computations with Magma [BCP97] by randomly generating alternant and Goppa codes over the field $\mathbb{F}_q$ with $q \in \{2, 4, 8, 16, 32\}$ for r in the range $\{3, \ldots, 50\}$ and several values of m. Furthermore, in our probabilistic model, a random alternant code is obtained by picking uniformly and independently at random two vectors $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ from $(\mathbb{F}_{q^m})^n$ such that the x_i's are all different and the y_i's are all nonzero. A random Goppa code is obtained by taking a random vector $(x_1, \ldots, x_n)$ in $(\mathbb{F}_{q^m})^n$ with all the x_i's different and a random *irreducible* polynomial $\gamma(z) = \sum_i \gamma_i z^i$ of degree r. In our experiments, it appears that D is amazingly *large even in the case where* $k \geqslant \binom{mr}{2}$. It even depends on whether or not the code with generator matrix $\boldsymbol{G}$ is chosen as a (generic) alternant code or as a Goppa code. Interestingly enough, when $\boldsymbol{G}$ is chosen at random, $\mathrm{rank}(\mathcal{L}_{\boldsymbol{P}})$ is equal to $\min\left\{k, \binom{mr}{2}\right\}$ with very high probability. In particular, the dimension of the solution space is typically 0 when k is larger than the number of variables $\binom{mr}{2}$ as one would expect. This will be proved in Section 10.4.

Although this *defect* in the rank is an obstacle to break the McEliece cryptosystem, it can be used to distinguish the public generator from a random code. But before doing so, let us remark first that although the linear system $\mathcal{L}_{\boldsymbol{P}}$ is defined over $\mathbb{F}_q$, there exists potentially two vector spaces of solutions depending on whether we focus on $\mathbb{F}_{q^m}$ or $\mathbb{F}_q$. We shall see that this ambiguity can be solved through the following definition.

Definition 28. *For any integer $r \geqslant 1$ and $m \geqslant 1$, let us denote by $N \stackrel{def}{=} \binom{mr}{2}$ the number of variables in the linear system $\mathcal{L}_{\boldsymbol{P}}$ as defined in (10.4) and D the dimension of the vector space of solutions of $\mathcal{L}_{\boldsymbol{P}}$. The* normalized dimension *of $\mathcal{L}_{\boldsymbol{P}}$ denoted by Δ is defined as $\Delta \stackrel{def}{=} \frac{D}{m}$.*

Throughout the paper we consider three cases: when the p_{ij}'s are chosen uniformly and independently at random in $\mathbb{F}_q$ then we denote the normalized dimension by Δ_{random}. When $\boldsymbol{G}$ is chosen as a generator matrix of a random alternant (*resp.* Goppa) code of degree r, we denote it by $\Delta_{\text{alternant}}$ (*resp.* Δ_{Goppa}). Our experiments have revealed that the normalized dimension of the vector space over $\mathbb{F}_q$ of the solutions of (10.4) is *predictable* and follows formulas.

Experimental Fact 1 (Alternant Case)**.** *As long as $N - m\Delta_{alternant} < k$, with very high probability the normalized dimension $\Delta_{alternant}$ is equal to $T_{alternant}$ where by definition:*

$$T_{alternant} \stackrel{def}{=} \frac{1}{2}(r-1)\left((2e+1)r - 2\frac{q^{e+1}-1}{q-1}\right) \tag{10.5}$$

and where $e \stackrel{def}{=} \lfloor \log_q(r-1) \rfloor$.

As for the case of random Goppa codes we also obtain formulas different from those of alternant codes. Note however that the Goppa codes are generated by means of a random irreducible $\gamma(z)$ of degree r and hence $\gamma(z)$ has no multiple roots. In particular, we can apply Theorem 3 in the binary case.

Experimental Fact 2 (Goppa Case)**.** *As long as $N - m\Delta_{Goppa} < k$, with very high probability the normalized dimension Δ_{Goppa} is equal to T_{Goppa} where by definition:*

$$T_{Goppa} \stackrel{def}{=} \begin{cases} \frac{1}{2}(r-1)(r-2) = T_{alternant} \text{ for } r < q-1 \\ \\ \frac{1}{2}r\Big((2e+1)r - 2q^e + 2q^{e-1} - 1\Big) \text{ for } r \geqslant q-1 \end{cases} \tag{10.6}$$

and where e is the unique integer such that:

$$q^e - 2q^{e-1} + q^{e-2} < r \leqslant q^{e+1} - 2q^e + q^{e-1}.$$

Based upon these experimental observations, we are now able to define a *distinguisher* between random codes, alternant codes and Goppa codes. This distinguisher will be in particular useful to distinguish McEliece public keys from random matrices.

Definition 29. *Let m and r be integers such that $m \geqslant 1$ and $r \geqslant 1$. Let $\boldsymbol{G}$ be a $k \times n$ matrix whose entries are in $\mathbb{F}_q$ with $n \leqslant q^m$ and $k \stackrel{def}{=} n - rm$. Without loss of generality, we assume that $\boldsymbol{G}$ is systematic i.e. $\boldsymbol{G} = (\boldsymbol{I}_k \mid \boldsymbol{P})$. Let $\mathcal{L}_{\boldsymbol{P}}$ be the linear system associated to $\boldsymbol{G}$ as defined in* (10.4)*, and Δ the normalized dimension of $\mathcal{L}_{\boldsymbol{P}}$. We define the* Random Code Distinguisher *$\mathcal{D}$ as the mapping which takes in input $\boldsymbol{G}$ and outputs b in $\{-1, 0, 1\}$ such that $\mathcal{D}(\boldsymbol{G}) = -1$ if $\Delta = T_{alternant}$, $\mathcal{D}(\boldsymbol{G}) = 0$ if $\Delta = T_{Goppa}$, and $\mathcal{D}(\boldsymbol{G}) = 1$ otherwise.*

10.4 Random Case

The purpose of this section is to study the behavior of D_{random}, namely the dimension of the solution space of $\mathcal{L}_{\boldsymbol{P}}$ when the entries of the matrix $\boldsymbol{P}$ are drawn independently from the uniform distribution over $\mathbb{F}_q$. In this case, we can show that:

Theorem 4. *Assume that $N \leqslant k$ and that the entries of $\boldsymbol{P}$ are drawn independently from the uniform distribution over $\mathbb{F}_q$. Then for any function $\omega(x)$ tending to infinity as x goes to infinity, we have that as mr goes to infinity*

$$\mathbf{prob}\Big(D_{\textit{random}} \geqslant mr\omega(mr)\Big) = o(1).$$

Notice that if choose $\omega(x) = \log(x)$ for instance, then asymptotically the dimension D_{random} of the solution space is with very large probability smaller than $mr\log(mr)$. When m and r are of the same order – which is generally chosen in practice – this quantity is smaller than $D_{\text{alternant}}$ or D_{Goppa} which are of the form $\Omega(mr^2)$. The main ingredient for proving Theorem 4 consists in analyzing a certain (partial) Gaussian elimination process on the matrix $\boldsymbol{M} \stackrel{\text{def}}{=} (p_{ij}p_{ij'})_{\substack{1\leqslant i\leqslant k \\ k+1\leqslant j<j'\leqslant n}}$. Basically it amounts to view the matrix $\boldsymbol{M}$ in block form, each block consisting in the matrix $\boldsymbol{B}_j = (p_{ij}p_{ij'})_{\substack{1\leqslant i\leqslant k \\ j<j'\leqslant n}}$ with $k+1 \leqslant j < n$. Each $\boldsymbol{B}_j$ is of size $k \times (rm - j)$. Notice that in $\boldsymbol{B}_j$, the rows for which $p_{i,j} = 0$ consist only of zeros.

To start the Gaussian elimination process with $\boldsymbol{B}_1$, we will therefore pick up $rm-1$ rows for which $p_{i,k+1} \neq 0$. This gives a square matrix $\boldsymbol{M}_1$. We perform Gaussian elimination on $\boldsymbol{M}$ by adding rows involved in $\boldsymbol{M}_1$ to put the first block $\boldsymbol{B}_1$ in standard form. We carry on this process with $\boldsymbol{B}_2$ by picking now $rm-2$ rows which have not been chosen before and which correspond to $p_{i,k+2} \neq 0$. This yields a square submatrix $\boldsymbol{M}_2$ of size $rm-2$ and we continue this process until reaching the last block. The key observation is that:

$$\operatorname{rank}(\boldsymbol{M}) \geqslant \operatorname{rank}(\boldsymbol{M}_1) + \operatorname{rank}(\boldsymbol{M}_2) + \cdots + \operatorname{rank}(\boldsymbol{M}_{rm-1}).$$

A rough analysis of this process yields the theorem above. The important point is what happens for different blocks are independent processes, it corresponds to looking at different rows of the matrix $\boldsymbol{P}$. A more detailed analysis would probably yield a stronger result that $\mathbf{prob}(D_{\text{random}} \geqslant \omega(mr)) = o(1)$, for any function ω going to infinity with mr or allowing to treat the case $N \geqslant k$ where we would like to show that $\mathbf{prob}(D_{\text{random}} \geqslant N - k + \omega(mr)) = o(1)$.

10.5 Interpretation of the Normalized Dimension

10.5.1 Alternant Case

We consider alternant codes over $\mathbb{F}_q$ of degree r. The goal is to identify a set of vectors of $(\mathbb{F}_{q^m})^n$ which, after decomposing each entry according to a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$, provides a basis of the solution space of $\mathcal{L}_{\boldsymbol{P}}$. Let us observe that to set up the linear system $\mathcal{L}_{\boldsymbol{P}}$ as defined in (10.4), we have used the trivial identity $Y_iY_iX_i^2 = (Y_iX_i)^2$. Actually, we can use any identity $Y_iX_i^aY_iX_i^b = Y_iX_i^cY_iX_i^d$ with $a, b, c, d \in \{0, 1, \ldots, r-1\}$ such that $a + b = c + d$. It is straightforward to check that we obtain the same algebraic system $\mathcal{L}_{\boldsymbol{P}}$ with:

$$\sum_{j=k+1}^{n} \sum_{j'>j} p_{i,j}p_{i,j'}(Y_jX_j^aY_{j'}X_{j'}^b + Y_{j'}X_{j'}^aY_jX_j^b + Y_jX_j^cY_{j'}X_{j'}^d + Y_{j'}X_{j'}^cY_jX_j^d) = 0. \tag{10.7}$$

So, the fact that *there are many different ways of combining the equations of the algebraic system together yielding the same linearized system $\mathcal{L}_{\boldsymbol{P}}$* explains why the dimension of the vector space solution is large. For larger values of r, the automorphisms of $\mathbb{F}_{q^m}$ of the kind $x \mapsto x^{q^\ell}$ for some $\ell \in \{0, \ldots, m-1\}$ can be used to obtain the identity for any integers a, b, c, d, ℓ, ℓ' such that $aq^{\ell'} + bq^\ell = cq^{\ell'} + dq^\ell$. We get again the linear system $\mathcal{L}_{\boldsymbol{P}}$ but the decomposition over $\mathbb{F}_q$ of the entries of vectors obtained from such equations give vectors that are dependent of those coming from the identity $Y_iX_i^aY_i^{q^{\ell-\ell'}}X_i^{bq^{\ell-\ell'}} = Y_iX_i^cY_i^{q^{\ell-\ell'}}X_i^{dq^{\ell-\ell'}}$ if we assume $\ell' \leqslant \ell$. Therefore, we are only interested in vectors that satisfy equations obtained with $0 \leqslant a, b, c, d < r$, $0 \leqslant \ell < m$ and $a + q^\ell b = c + q^\ell d$.

Definition 30. *Let a, b, c and d be integers in $\{0, \ldots, r-1\}$ and ℓ in $\{0, \ldots, \lfloor \log_q(r-1) \rfloor\}$ such that $a + q^\ell b = c + q^\ell d$. We define $\boldsymbol{Z}_{a,b,c,d,\ell} \stackrel{\text{def}}{=} \Big(\boldsymbol{Z}_{a,b,c,d,\ell}[j,j']\Big)_{k+1\leqslant j<j'\leqslant n}$ where $\boldsymbol{Z}_{a,b,c,d,\ell}[j,j'] \stackrel{\text{def}}{=} Y_jX_j^aY_{j'}^{q^\ell}X_{j'}^{q^\ell b} + Y_{j'}X_{j'}^aY_j^{q^\ell}X_j^{q^\ell b} + Y_jX_j^cY_{j'}^{q^\ell}X_{j'}^{q^\ell d} + Y_{j'}X_{j'}^cY_j^{q^\ell}X_j^{q^\ell d}$, for any j and j' satisfying $k+1 \leqslant j < j' \leqslant n$.*

Without loss of generality, we can assume that $d > b$ and set $\delta \stackrel{\text{def}}{=} d - b$. The next proposition shows that some vectors $\boldsymbol{Z}_{c+q^\ell\delta,b,c,b+\delta,\ell}$ can be expressed as a linear combination of vectors defined with $\delta = 1$.

Proposition 24. *Let ℓ, δ, b and c be integers such that $\ell \geqslant 0$, $\delta \geqslant 1$, $1 \leqslant b + \delta \leqslant r-1$ and $1 \leqslant c + q^\ell\delta \leqslant r-1$. Let us assume that $\delta \geqslant 2$ and let $b_i \stackrel{\text{def}}{=} b + i - 1$ and $c_i \stackrel{\text{def}}{=} c + q^\ell(\delta - i)$. We have*

$$\boldsymbol{Z}_{c+q^\ell\delta,b,c,b+\delta,\ell} = \sum_{i=1}^{\delta} \boldsymbol{Z}_{c_i+q^\ell,b_i,c_i,b_i+1,\ell}. \tag{10.8}$$

Definition 31. *Let $\mathcal{B}_r$ be the set of* nonzero *vectors $\boldsymbol{Z}_{c+q^\ell\delta,b,c,b+\delta,\ell}$ obtained with tuples (δ,b,c,ℓ) such that $\delta = 1$ while satisfying $0 \leqslant b < c \leqslant r-2$ if $\ell = 0$, and if $1 \leqslant \ell \leqslant \lfloor \log_q(r-1) \rfloor$:*

$$\begin{cases} 0 \leqslant b \leqslant r-2, \\ 0 \leqslant c \leqslant r-1-q^\ell. \end{cases}$$

Proposition 25. *For any integer $r \geqslant 3$ the cardinality of $\mathcal{B}_r$ is equal to $T_{\text{alternant}}$.*

Proposition 25 gives an explanation of the value of $D_{\text{alternant}}$ and gives the following heuristic.

Heuristic 1. *Consider a certain decomposition of the elements of $\mathbb{F}_{q^m}$ in a $\mathbb{F}_q$ basis. Let $\pi_i : \mathbb{F}_{q^m} \to \mathbb{F}_q$ be the function giving the i-th coordinate in this decomposition where $1 \leqslant i \leqslant m$. By extension we denote for $\boldsymbol{z} = (z_j)_{1\leqslant j\leqslant n} \in (\mathbb{F}_{q^m})^n$ by $\pi_i(\boldsymbol{z})$ the vector $(\pi_i(z_j))_{1\leqslant j\leqslant n} \in \mathbb{F}_q^n$. Then, for any j such that $1 \leqslant j \leqslant n$ and for random choices of x_j's and y_j's, the set $\{\pi_i(\boldsymbol{Z}) \mid 1 \leqslant i \leqslant m, \boldsymbol{Z} \in \mathcal{B}_r\}$ forms a basis of the vector space of solutions of $\mathcal{L}_{\boldsymbol{P}}$.*

10.5.2 Binary Goppa Case

In this section we will explain Experimental Fact 2 observed for binary Goppa codes. We denote by r the degree of the Goppa polynomial. The theoretical expression T_{Goppa} has a simpler expression in that special case.

Proposition 26. *Let $e \stackrel{def}{=} \lceil \log_2 r \rceil + 1$ and $N \stackrel{def}{=} \binom{mr}{2}$. When $q = 2$, Formula* (10.6) *can be simplified to:*

$$T_{Goppa} = \frac{1}{2} r \Big((2e+1)r - 2^e - 1 \Big). \tag{10.9}$$

Theorem 3 shows that a binary Goppa code of degree r can be regarded as a binary alternant code of degree $2r$. This seems to indicate that we should have $D_{\text{Goppa}}(r) = mT_{\text{alternant}}(2r)$. This is not the case however. It turns out that $D_{\text{Goppa}}(r)$ is significantly smaller than this. In our experiments, we have found out that the vectors of $\mathcal{B}_{2r}$ still form a generating set for the solution space of $\mathcal{L}_{\boldsymbol{P}}$, but they are not independent anymore. Our goal is therefore to identify the dependencies between $\pi_i(\boldsymbol{Z})$'s with $\boldsymbol{Z}$ in $\mathcal{B}_{2r}$. Although we are firstly interested in linear relations between the $\pi_i(\boldsymbol{Z})$'s, we shall see that many of them come from $\mathbb{F}_{2^m}$-relations that link directly the $\boldsymbol{Z}$'s as shown by the following proposition which exploits the fact that the Y_i's are derived from the Goppa polynomial $\gamma(z)$ by $Y_i = \gamma(X_i)^{-1}$.

Proposition 27. *Let t, ℓ and c be integers such that $0 \leqslant t \leqslant r-2$, $1 \leqslant \ell \leqslant \lfloor \log_2(2r-1) \rfloor$ and $0 \leqslant c \leqslant 2r - 2^\ell - 1$. We set $c^* \stackrel{def}{=} c + 2^{\ell-1}$. It holds that:*

$$\sum_{b=0}^{r} \gamma_b^{2^\ell} \boldsymbol{Z}_{c+2^\ell,t+b,c,t+b+1,\ell} = \boldsymbol{Z}_{c^*+2^{\ell-1},2t,c^*,2t+1,\ell-1} + \boldsymbol{Z}_{c+2^{\ell-1},2t+1,c,2t+2,\ell-1}. \tag{10.10}$$

As a consequence, the set $\{\pi_i(\boldsymbol{Z}) \mid 1 \leqslant i \leqslant m, \boldsymbol{Z} \in \mathcal{B}_{2r}\}$ can not be a basis of the linearized system in the Goppa case.

Proposition 28. *The number N_L of equations of the form* (10.10) *is $2(r-1)(ru+1-2^u)$ where $u \stackrel{def}{=} \lfloor \log_2(2r-1) \rfloor$.*

Notice that each equation of the form (10.10) involves one vector of $\mathcal{B}_{2r}$ that does not satisfy the other equations. These equations are therefore independent and if we denote by $< \mathcal{B}_{2r} >_{\mathbb{F}_{2^m}}$ the vector space over $\mathbb{F}_{2^m}$ generated by the vectors of $\mathcal{B}_{2r}$ we should have:

$$\dim < \mathcal{B}_{2r} >_{\mathbb{F}_{2^m}} \leqslant |\mathcal{B}_{2r}| - N_L.$$

The experimentations we have made indicate that actually equality holds here. However, this does not mean that the dimension of the vector space over $\mathbb{F}_2$ generated by the set $\{\pi_i(\boldsymbol{Z}) \mid \boldsymbol{Z} \in \mathcal{B}_{2r}, 1 \leqslant i \leqslant m, \boldsymbol{Z} \in \mathcal{B}_{2r}\}$ is equal to $m \dim < \mathcal{B}_{2r} >_{\mathbb{F}_{2^m}}$. It turns out that there are still other dependencies among the $\pi_i(\boldsymbol{Z})$'s. To see this, let us define the vector $\boldsymbol{Q}_{a,b,c,d,\ell} \stackrel{\text{def}}{=} \left(\boldsymbol{Q}_{a,b,c,d,\ell}[j,j']\right)_{k+1\leqslant j<j'\leqslant n}$ with:

$$\boldsymbol{Q}_{a,b,c,d,\ell}[j,j'] = \left(\boldsymbol{Z}_{a,b,c,d,\ell}[j,j']\right)^2.$$

Observe also that for any $1 \leqslant i \leqslant m$ we always have:

$$\pi_i(\boldsymbol{Q}_{a,b,c,d,\ell}) \in \{\pi_i(\boldsymbol{Z}) \mid 1 \leqslant i \leqslant m, \boldsymbol{Z} \in \mathcal{B}_{2r}\}.$$

Proposition 29. *For any integers $b \geqslant 0$, $t \geqslant 0$, $\delta \geqslant 1$ and ℓ such that $0 \leqslant \ell \leqslant \lfloor \log_2(2r-1) \rfloor - 1$, $b + \delta \leq 2r-1$ and $t + 2^\ell\delta \leqslant r-1$, we have*

$$\boldsymbol{Z}_{2t+2^{\ell+1}\delta,b,2t,b+\delta,\ell+1} = \sum_{c=0}^{r} \gamma_c^2 \boldsymbol{Q}_{c+2^\ell\delta,b,t+c,b+\delta,\ell}. \tag{10.11}$$

Proposition 30. *Let N_Q he number of vectors of $\mathcal{B}_{2r}$ satisfying Equation* (10.11) *and $u \stackrel{def}{=} \lfloor \log_2(2r-1) \rfloor$, we have that*

$$N_Q = (2r-1)(ru - 2^u + 1).$$

Table 10.1: A binary Goppa code of length $n = 2^m$ and degree $r < r_{\max}$ is distinguishable from a random code.

m	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
$r_{\max}$	5	8	8	11	16	20	26	34	47	62	85	114	157	213	290	400

Each of such equation gives rise to m linear equations over $\mathbb{F}_2$ involving the $\pi_i(\boldsymbol{Z})$ for $\boldsymbol{Z}$ in $\mathcal{B}_{2r}$. Therefore, it could be expected that $\Delta_{\text{Goppa}} = |\mathcal{B}_{2r}| - N_L - N_Q$. But, some of the N_Q vectors of $\mathcal{B}_{2r}$ are counted twice because they appear both in linear relations of the form (10.10) and "quadratic" equations of the form (10.11). Let $N_{L\cap Q}$ be the number of such vectors. By counting them carefully we can prove that:

Proposition 31. $N_{L\cap Q} = (r-1)\Big((u-\frac{1}{2})r - 2^u + 2\Big)$ *where* $u \stackrel{def}{=} \lfloor \log_2(2r-1) \rfloor$.

Proposition 32. *For any integer* $r \geqslant 2$*, we have*

$$T_{Goppa}(r) = |\mathcal{B}_{2r}| - N_L - N_Q + N_{L\cap Q}.$$

10.6 Conclusion and Cryptographic Implications

The existence of a distinguisher for the specific case of binary Goppa codes is not valid for any value of r and m but tends to be true for codes that have a rate $\frac{n-mr}{n}$ close to one. This kind of codes are mainly encountered with the signature scheme [CFS01]. If we assume that the length n is equal to 2^m and we denote by $r_{\max}$ the smallest integer r such that $N - mT_{\text{Goppa}} \geqslant 2^m - mr$ then any binary Goppa code defined of degree $r < r_{\max}$ can be distinguished (Table 10.1). For example, the binary Goppa code obtained with $m = 13$ and $r = 19$ corresponding to a McEliece public key of 90 bits of security, is distinguishable. More interestingly, all the keys proposed in [FS09] for the signature scheme can be distinguished.

We recall that the existence of such a distinguisher does not undermine the security of [McE78] and [CFS01]. It only shows that their security should not be reduced to the difficulty of decoding a random linear code by means of the GCD assumption. Therefore this would suggest to directly assume that the McEliece trapdoor function is one-way without any other assumptions.

Chapter 11

Cryptanalysis of KKS Signature Scheme[1]

11.1 Introduction

Digital signature schemes are probably among the most useful cryptographic algorithms. If quantum computers were to become reality, it would be useful to devise such schemes which would resist to it. A possible approach to meet this goal could be to build such schemes whose security relies on the difficulty of decoding linear codes. Two code based schemes of this kind have been proposed, namely the Courtois-Finiasz-Sendrier signature scheme [CFS01] and the Kabatianskii, Krouk and Smeets (KKS) scheme [KKS97, KKS05].

The Courtois-Finiasz-Sendrier (CFS) scheme presents the advantage of having an extremely short signature and its security has been proven to rely on the well-known syndrome decoding problem and the distinguishability of binary Goppa codes from a random code. However, it has been proved in [FGUO$^+$11] that the latter problem can be solved in the range of parameters used in the CFS signature algorithm. This does not prove that their proposal is insecure. However, it invalidates the hypotheses of their security proof. The main difficulty in suggesting a CFS type scheme is to come up with a family of very high rate codes with an efficient decoding algorithm and whose structure can be hidden in the same way as in the McEliece scheme. This narrows down quite a bit the families of codes which can be used in this setting and up to now only Goppa codes are known to meet this goal. It should be emphasized that it is precisely their rich algebraic structure which makes it possible to distinguish them from random codes.

On the other hand, the KKS proposal does not rely on Goppa codes and can be instantiated with random codes. Moreover, unlike in the CFS signature scheme, it does not compute a signature by using a decoding algorithm for the code and thus completely avoids the necessity of having to use restricted families of codes with a "hidden" trapdoor. Moreover, a variation of it has been proposed in [BMJ11] and has been proved to be EUF-1CMA secure in the random oracle model. The security of the KKS scheme has been investigated in [COV07]. It was shown that a passive attacker who may intercept just a few signatures can recover the private key. All the schemes proposed in [KKS97] can be broken in this way with the help of at most 20 signatures. Basically it uses the fact that a valid message-signature pair reveals on average half of the secret support J (see Section 11.3 where this set is defined precisely). Therefore with $O(\log |J|)$ message-signature pairs it is expected to recover the whole set J. The security of the scheme is not compromised by this attack however if only one signature is computed, and this especially in the variant proposed in [BMJ11] where some random noise is added on top of the signature.

The purpose of this article is to present a completely new security analysis of the KKS scheme and its variant proposed in [BMJ11]. Our approach for breaking the scheme is to define a certain error correcting code from the couple of public matrices used in the scheme and to notice that certain rather low weight codewords give actually valid signatures. It is therefore natural to use standard algorithms for finding low-weight codewords in this setting, such as Stern's algorithm [Ste88] or its Dumer variant [Dum96, FS09] (see also [BLP11]). It turns out that such algorithms are unusually successful in this setting due to the conjunction of three factors: (i) there are many low-weight codewords, (ii) they are localized on a rather small support, (iii) some part of this support is known to the attacker. It appears that all parameters suggested in [KKS97, KKS05, BMJ11] are easily broken by this approach and this without even knowing a single signature pair. Moreover, this approach can exploit the knowledge of a message-signature pair which speeds up the attack.

We provide an analysis of this attack which explains what makes it feasible for the parameters proposed in [KKS97, KKS05, BMJ11]. The KKS scheme relies on a couple of matrices which can be viewed as parity-check matrices of two linear codes. We show that when the first code has a rate which is smaller than the rate of the second one (or has approximately the same rate), then our attack is quite successful. This was exactly the case for all the parameters suggested in the past. In other words, our attack does not compromise the security of the whole KKS scheme. It just points out that the region of weak parameters is really much larger than previously thought.

11.2 Terminology and Notation

In the whole paper q denotes some prime power and we denote by $\mathbb{F}_q$ the finite field with q elements. Let n be a non-negative integer. The set of integers i such that $1 \leqslant i \leqslant n$ is denoted by $[1 \cdots n]$. The cardinality of a set A is denoted by $|A|$. The

[1] This chapter is a complete reproduction of the article [OT11] that will be presented at PQCrypto 2011.

concatenation of the vectors $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_m)$ is denoted by $(\boldsymbol{x}||\boldsymbol{y}) \stackrel{\text{def}}{=} (x_1, \ldots, x_n, y_1, \ldots, y_m)$. The support $\mathsf{supp}(\boldsymbol{x})$ of $\boldsymbol{x} \in \mathbb{F}_q^n$ is the set of i's such that $x_i \neq 0$. The *(Hamming) weight* $\mathsf{wt}(\boldsymbol{x})$ is the cardinality of $\mathsf{supp}(\boldsymbol{x})$. For a vector $\boldsymbol{x} = (x_i)$ and a subset I of indices of $\boldsymbol{x}$, we denote by $\boldsymbol{x}_I$ its restriction to the indices of I, that is:

$$\boldsymbol{x}_I \stackrel{\text{def}}{=} (x_i)_{i \in I}.$$

We will also use this notation for matrices, in this case it stands for the submatrix formed by the columns in the index set, i.e. for any $k \times n$ matrix $\boldsymbol{H}$

$$\boldsymbol{H}_J \stackrel{\text{def}}{=} (h_{ij})_{\substack{1 \leqslant i \leqslant k \\ j \in J}}.$$

A linear code $\mathscr{C}$ of type $[n, k, d]$ over $\mathbb{F}_q$ is a linear subspace of $\mathbb{F}_q^n$ of dimension k and minimum distance d where by definition $d \stackrel{\text{def}}{=} \min\{\mathsf{wt}(\boldsymbol{x}) \ : \ \boldsymbol{x} \in \mathscr{C} \text{ and } \boldsymbol{x} \neq \boldsymbol{0}\}$. The elements of $\mathscr{C}$ are *codewords*. A linear code can be defined either by a parity check matrix or a generator matrix. A *parity check matrix* $\boldsymbol{H}$ for $\mathscr{C}$ is an $(n-k) \times n$ matrix such that $\mathscr{C}$ is the right kernel of $\boldsymbol{H}$:

$$\mathscr{C} = \{\boldsymbol{c} \in \mathbb{F}_q^n \ : \ \boldsymbol{H}c^T = 0\}$$

where $\boldsymbol{x}^T$ denotes the *transpose* of $\boldsymbol{x}$. A generator matrix $\boldsymbol{G}$ is a $k \times n$ matrix formed by a basis of $\mathscr{C}$. We say that $\boldsymbol{G}$ is in *systematic* form if there exists a set J such that $\boldsymbol{G}_J = \boldsymbol{I}_k$. The *syndrome* $\boldsymbol{s}$ by $\boldsymbol{H}$ of $\boldsymbol{x} \in \mathbb{F}_q^n$ is defined as $\boldsymbol{s}^T \stackrel{\text{def}}{=} \boldsymbol{H}\boldsymbol{x}^T$. A *decoding* algorithm for $\boldsymbol{H}$ is an algorithm such that, given $\boldsymbol{s}$ in $\mathbb{F}_q^r$, finds a vector $\boldsymbol{e}$ of *minimum* weight whose syndrome is $\boldsymbol{s}$.

11.3 The Kabatianskii-Krouk-Smeets Signature Scheme and its Variant

This section is devoted to the description of two code-based signature schemes proposed in [KKS97] and more recently in [BMJ11], where the latter can be viewed as a "noisy" version of the former [KKS97]. Our presentation presents the main ideas without giving all the details which can be found in the original papers. We first focus on the scheme of [KKS97] whose construction relies on the following ingredients:

1. a full rank binary matrix $\boldsymbol{H}$ of size $(N-K) \times N$ with entries in a finite field $\mathbb{F}_q$.
2. a subset J of $\{1, \ldots, N\}$ of cardinality n,
3. a linear code $\mathscr{C}_{\text{hidden}}$ over $\mathbb{F}_q$ of length $n \leqslant N$ and dimension k defined by a generator matrix $\boldsymbol{G}$ of size $k \times n$. Let t_1 and t_2 be two integers such that with very high probability, we have that $t_1 \leqslant \mathsf{wt}(\boldsymbol{u}) \leqslant t_2$ for any non-zero codeword $\boldsymbol{u} \in \mathscr{C}_{\text{hidden}}$.

The matrix $\boldsymbol{H}$ is chosen such that the best decoding algorithms cannot solve the following search problem.

Problem 1. *Given the knowledge of $\boldsymbol{s} \in \mathbb{F}_q^{N-K}$ which is the syndrome by $\boldsymbol{H}$ of some $\boldsymbol{e} \in \mathbb{F}_q^N$ whose weight lies in $[t_1 \cdots t_2]$, find explicitly $\boldsymbol{e}$, or eventually $\boldsymbol{x}$ in $\mathbb{F}_q^N$ different from $\boldsymbol{e}$ sharing the same properties as $\boldsymbol{e}$.*

Finally let $\boldsymbol{F}$ be the $(N-K) \times k$ matrix defined by $\boldsymbol{F} \stackrel{\text{def}}{=} \boldsymbol{H}_J\boldsymbol{G}^T$. The Kabatianskii-Krouk-Smeets (KKS) signature scheme is then described in Figure 11.1.

Figure 11.1: Description of the KKS scheme given in [KKS97].

- Setup.
  1. The signer S chooses N, K n, k, t_1 and t_2 according to the required security level.
  2. S draws a random $(N-K) \times N$ matrix $\boldsymbol{H}$.
  3. S randomly picks a subset J of $\{1, \ldots, N\}$ of cardinality n.
  4. S randomly picks a random $k \times n$ generator matrix $\boldsymbol{G}$ that defines a code $\mathscr{C}_{\text{hidden}}$ such that with high probability $t_1 \leqslant \mathsf{wt}(\boldsymbol{u}) \leqslant t_2$ for any non-zero codeword $\boldsymbol{u} \in \mathscr{C}_{\text{hidden}}$.
  5. $\boldsymbol{F} \stackrel{\text{def}}{=} \boldsymbol{H}_J\boldsymbol{G}^T$ where $\boldsymbol{H}_J$ is the restriction of $\boldsymbol{H}$ to the columns in J.
- Keys.
  - Private key. J and $\boldsymbol{G}$
  - Public key. $\boldsymbol{F}$ and $\boldsymbol{H}$
- Signature. The signature σ of a message $\boldsymbol{x} \in \mathbb{F}_q^k$ is defined as the unique vector σ of $\mathbb{F}_q^N$ such that $\sigma_i = 0$ for any $i \notin J$ and $\sigma_J = \boldsymbol{x}\boldsymbol{G}$.
- Verification. Given $(\boldsymbol{x}, \sigma) \in \mathbb{F}_q^k \times \mathbb{F}_q^N$, the verifier checks that $t_1 \leqslant \mathsf{wt}(\sigma) \leqslant t_2$ and $\boldsymbol{H}\sigma^T = \boldsymbol{F}\boldsymbol{x}^T$.

The scheme was modified in [BMJ11] to propose a one-time signature scheme by introducing two new ingredients, namely a hash function f and adding an error vector $\boldsymbol{e}$ to the signature. It was proved that such a scheme is EUF-1CMA secure in the random oracle model. The description is given in Figure 11.2.

Figure 11.2: Description of the scheme of [BMJ11].

- Setup.
  1. The signer S chooses N, K n, k, t_1 and t_2 according to the required security level.
  2. S chooses a hash function $f : \{0,1\}^* \times \mathbb{F}_2^{N-K} \longrightarrow \mathbb{F}_2^k$.
  3. S draws a random binary $(N-K) \times N$ matrix $\boldsymbol{H}$.
  4. S randomly picks a subset J of $\{1, \ldots, N\}$ of cardinality n.
  5. S randomly picks a $k \times n$ generator matrix $\boldsymbol{G}$ that defines a binary code $\mathscr{C}_{\text{hidden}}$ such that with high probability $t_1 \leqslant \mathsf{wt}(\boldsymbol{u}) \leqslant t_2$ for any non-zero codeword $\boldsymbol{u} \in \mathscr{C}_{\text{hidden}}$.
  6. $\boldsymbol{F} \stackrel{\text{def}}{=} \boldsymbol{H}_J \boldsymbol{G}^T$ where $\boldsymbol{H}_J$ is the restriction of $\boldsymbol{H}$ to the columns in J.
- Keys.
  - Private key. J and $\boldsymbol{G}$
  - Public key. $\boldsymbol{F}$ and $\boldsymbol{H}$
- Signature. The signature of a message $\boldsymbol{x} \in \{0,1\}^*$ is (h, σ) defined as follows:
  - S picks a random $\boldsymbol{e} \in \mathbb{F}_2^N$ such that $\mathsf{wt}(\boldsymbol{e}) = n$.
  - Let $\boldsymbol{h} \stackrel{\text{def}}{=} f(\boldsymbol{x}, \boldsymbol{H}\boldsymbol{e}^T)$ and $\boldsymbol{y}$ be the unique vector of $\mathbb{F}_2^N$ such that (i) $\mathsf{supp}(\boldsymbol{y}) \subset J$, (ii) $\boldsymbol{y}_J = \boldsymbol{h}\boldsymbol{G}$. The second part of the signature σ is then given by $\sigma \stackrel{\text{def}}{=} \boldsymbol{y} + \boldsymbol{e}$.
- Verification. Given a signature $(\boldsymbol{h}, \sigma) \in \mathbb{F}_2^k \times \mathbb{F}_2^N$ for $\boldsymbol{x} \in \{0,1\}^*$, the verifier checks that $\mathsf{wt}(\sigma) \leqslant 2n$ and $\boldsymbol{h} = f(\boldsymbol{x}, \boldsymbol{H}\sigma^T + \boldsymbol{F}\boldsymbol{h}^T)$.

11.4 Description of the Attack

The purpose of this section is to explain the idea underlying our attack which aims at recovering the private key. The attack is divided in two main steps. First, we produce a valid signature for some message using only the public key. To do so, we define a certain code from matrices $\boldsymbol{H}$ and $\boldsymbol{F}$. It turns out that low weight codewords of this code give valid message-signature pairs. Then we just apply Dumer's algorithm [Dum91] in order to find these low weight codewords. This attack can even be refined in the following way. Whenever we are able to produce one valid message-signature pair, and since each signature reveals partial information about the private key (especially about J as explained further in this section), we can use it to get another valid message-signature pair revealing more information about J. We repeat this process a few times until we totally recover the whole private key. More details will be given in the following sections.

In what follows, we make the assumption that all the codes are binary because all the concrete proposals are of this kind. The non-binary case will be discussed in the conclusion.

11.4.1 An auxiliary code

We give here the first ingredient we use to forge a valid message/signature pair for the KKS scheme just from the knowledge of the public pair $\boldsymbol{H}, \boldsymbol{F}$. This attack can also be used for the second scheme given by Figure 11.2. In the last case, it is not a valid message/signature pair anymore but an auxiliary quantity which helps in revealing J. This ingredient consists in a linear code $\mathscr{C}_{\text{pub}}$ of length $N+k$ defined as the kernel of $\hat{\boldsymbol{H}}$ which is obtained by the juxtaposition of the two public matrices $\boldsymbol{H}$ and $\boldsymbol{F}$ as given in Figure 11.3. The reason behind this definition lies in the following Fact 4.

Fact 4. *Let $\boldsymbol{x}'$ be in $\mathbb{F}_2^{N+k}$ and set $(\sigma||\boldsymbol{x}) \stackrel{def}{=} \boldsymbol{x}'$ with σ in $\mathbb{F}_2^N$ and $\boldsymbol{x}$ in $\mathbb{F}_2^k$. Then σ is a signature of $\boldsymbol{x}$ if and only if:*

1. $\hat{\boldsymbol{H}}\boldsymbol{x}'^T = 0$
2. $t_1 \leqslant \mathsf{wt}(\sigma) \leqslant t_2$.

The code $\mathscr{C}_{\text{pub}}$ is of dimension $k+K$, and of particular interest is the linear space $\mathscr{C}_{\text{sec}} \subset \mathscr{C}_{\text{pub}}$ that consists in words that satisfy both conditions of Fact 4 and that are obtained by all pairs $(\sigma, \boldsymbol{x})$ of valid message-signature pairs which are obtained by

Figure 11.3: Parity-check matrix $\hat{\boldsymbol{H}}$ of the code $\mathscr{C}_{\text{pub}}$

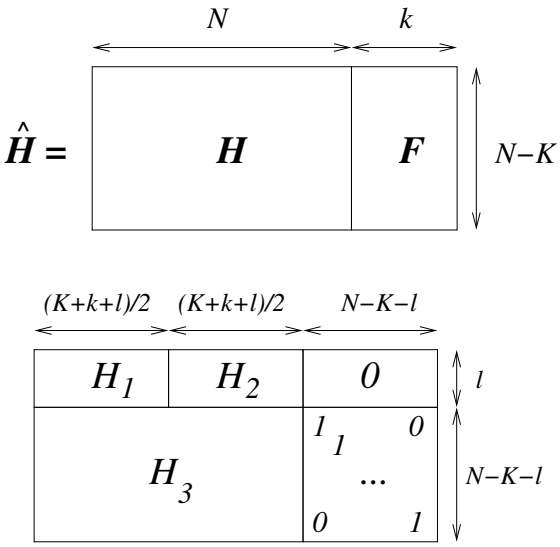

Figure 11.4: A parity-check matrix for $\mathscr{C}_{\text{pub}}$ in quasi-systematic form.

the secret signature algorithm, that is to say:

$$\mathscr{C}_{\text{sec}} \stackrel{\text{def}}{=} \Big\{(\sigma||\boldsymbol{x}) \in \mathbb{F}_2^{N+k} \ : \ \boldsymbol{x} \in \mathbb{F}_2^k,\ \sigma \in \mathbb{F}_2^N,\ \sigma_J = \boldsymbol{x}\boldsymbol{G},\ \sigma_{[1\cdots N]\setminus J} = 0\Big\}. \tag{11.1}$$

Clearly, the dimension of $\mathscr{C}_{\text{sec}}$ is k. Additionally, we expect that the weight of σ is of order $n/2$ for any $(\sigma, \boldsymbol{x})$ in $\mathscr{C}_{\text{sec}}$, which is much smaller than the total length N. This strongly suggests to use well-known algorithms for finding low weight codewords to reveal codewords in $\mathscr{C}_{\text{sec}}$ and therefore message-signature pairs. The algorithm we used for that purpose is specified in the following subsection.

11.4.2 Finding low-weight codewords

We propose to use the following variation on Stern's algorithm due to [Dum91] (See also [FS09]). The description of the algorithm is given in Algorithm 1. It consists in searching for low-weight codewords among the candidates that are of very low-weight $2p$ (where p is typically in the range $1 \leqslant p \leqslant 4$) when restricted to a set I of size slightly larger than the dimension $k + K$ of the code $\mathscr{C}_{\text{pub}}$, say $|I| = k + K + l$ for some small integer l. The key point in this approach is to choose I among a set S of test positions. The set S will be appropriately chosen according to the considered context. If no signature pair is known, then a good choice for S is to take:

$$S = [1 \cdots N]. \tag{11.2}$$

This means that we always choose the test positions among the N first positions of the code $\mathscr{C}_{\text{pub}}$ and never among the k last positions. The reason for this choice will be explained in the following subsection.

11.4.3 Explaining the success of the attack

It turns out that this attack works extremely well on all the parameter choices made in the literature, and this even without knowing a single message-signature pair which would make life much easier for the attacker as demonstrated in [COV07]. In a first pass, the attack recovers easily message-signature pairs for all the parameters suggested in [BMJ11, KKS97, KKS05]. Once a signature-message pair is obtained, it can be exploited to bootstrap an attack that recovers the private key as we will explain later.

The reason why the attack works much better here than for general linear codes comes from the fact that $\hat{\boldsymbol{H}}$ *does not behave like a random matrix at all even if the two chosen matrices for the scheme, namely* $\boldsymbol{H}$ *and* $\boldsymbol{G}$ *are chosen at random*. The left part and the right part $\boldsymbol{H}$ and $\boldsymbol{F}$ are namely related by the equation:

$$\boldsymbol{F} = \boldsymbol{H}_J \boldsymbol{G}^T.$$

Indeed, the parity-check matrix $\hat{\boldsymbol{H}}$ displays peculiar properties: $\mathscr{C}_{\text{pub}}$ contains $\mathscr{C}_{\text{sec}}$ as a subcode and its codewords represent valid message-signature pairs. This subcode has actually a very specific structure that helps greatly the attacker:

1. There are many codewords in $\mathscr{C}_{\text{sec}}$, namely 2^k.
2. The support of these codewords is included in a fixed (and rather small) set of size $k + n$.

Algorithm 1 KKSforge: algorithm that forges a valid KKS signature.

PARAMETERS:

r : number of iterations,

l : small integer ($l \leqslant 40$),

p : very small integer ($1 \leqslant p \leqslant 4$).

S : a subset of $[1 \cdots N]$ from which in each iteration a subset of cardinality $K+k+l$ will be randomly chosen.

INPUT: $\hat{\boldsymbol{H}}$

OUTPUT: a list $\mathcal{L}$ containing valid signature/message pairs $(\sigma, \boldsymbol{x}) \in \mathbb{F}_2^N \times \mathbb{F}_2^k$.

$\mathcal{L} \leftarrow \emptyset$.
for $1 \leqslant t \leqslant r$ **do**
  Step 1: Randomly pick $K+k+l$ positions among S to form the set I. This set is partitioned into $I = I_1 \cup I_2$ such that $||I_1| - |I_2|| \leqslant 1$.
  Step 2: Perform Gaussian elimination over the complementary set $\{1, 2, \ldots, N+k\} \setminus I$ to put $\hat{\boldsymbol{H}}$ in quasi-systematic form (as shown in Figure 11.4).
  Step 3:
  Generate all binary vectors $\boldsymbol{x}_1$ of length $\lfloor (K+k+l)/2 \rfloor$ and weight p and store them in a table at the address $H_1\ \boldsymbol{x}_1^T$
  for all binary vectors $\boldsymbol{x}_2$ of length $\lceil (K+k+l)/2 \rceil$ and weight p **do**
    for all $\boldsymbol{x}_1$ stored at the address $H_2\ \boldsymbol{x}_2^T$ **do**
      Compute $\boldsymbol{x}_3 \stackrel{\text{def}}{=} (\boldsymbol{x}_1 || \boldsymbol{x}_2) \boldsymbol{H}_3^T$ and form the codeword $\boldsymbol{x} \stackrel{\text{def}}{=} (\boldsymbol{x}_1 || \boldsymbol{x}_2 || \boldsymbol{x}_3)$ of $\mathscr{C}_{\text{pub}}$
      if $t_1 \leqslant \mathsf{wt}(\boldsymbol{x}_{[1 \cdots N]}) \leqslant t_2$ **then**
        $\mathcal{L} \leftarrow \mathcal{L} \cup \{\boldsymbol{x}\}$
      end if
    end for
  end for
end for
return $\mathcal{L}$

3. k positions of this set are known to the attacker.

4. These codewords form a linear code (of dimension k).

Because of all these properties, the aforementioned attack will work much better than should be expected from a random code. More precisely, let us bring in:

$$I' \stackrel{\text{def}}{=} I \cap J.$$

Notice that the expectation $\mathbb{E}\{|I'|\}$ of the cardinality of the set I' is equal to:

$$\mathbb{E}\{|I'|\} = \frac{n}{N}(k+K+l) = (R + \alpha\rho + \lambda)n \tag{11.3}$$

where we introduced the following notation:

$$R \stackrel{\text{def}}{=} \frac{K}{N}, \quad \rho \stackrel{\text{def}}{=} \frac{k}{n}, \quad \alpha \stackrel{\text{def}}{=} \frac{n}{N} \quad \text{and} \quad \lambda \stackrel{\text{def}}{=} \frac{l}{N}.$$

The point is that whenever there is a codeword $\boldsymbol{c}$ in $\mathscr{C}_{\text{sec}}$ which is such that $\mathsf{wt}(\boldsymbol{c}_{I'}) = 2p$ we have a non-negligible chance to find it with Algorithm 1. This does not hold with certainty because the algorithm does not examine all codewords $\boldsymbol{x}$ such that $\mathsf{wt}(\boldsymbol{x}_I) = 2p$, but rather it consists in splitting I in I_1 and I_2 of the same size and looking for codewords $\boldsymbol{x}$ such that $\mathsf{wt}(\boldsymbol{x}_{I_1}) = \mathsf{wt}(\boldsymbol{x}_{I_2}) = p$. In other words, we consider only a fraction δ of such codewords where:

$$\delta = \frac{\binom{(K+k+l)/2}{p}\binom{(K+k+l)/2}{p}}{\binom{K+k+l}{2p}} \approx \sqrt{\frac{(K+k+l)}{\pi p(K+k+l-2p)}}.$$

We will therefore obtain all codewords $\boldsymbol{c}$ in $\mathscr{C}_{\text{sec}}$ which are such that $\mathsf{wt}(\boldsymbol{c}_{I_1} = \mathsf{wt}(\boldsymbol{c}_{I_2}) = p$. Consider now the restriction $\mathscr{C}'_{\text{sec}}$ of $\mathscr{C}_{\text{sec}}$ to the positions belonging to I', that is:

$$\mathscr{C}'_{\text{sec}} = \Big\{ (\boldsymbol{x}_i)_{i \in I'} \ : \ \boldsymbol{x} = (\boldsymbol{x}_i)_{i \in [1 \cdots N+k]} \in \mathscr{C}_{\text{sec}} \Big\}. \tag{11.4}$$

The crucial issue is now the following question:

Does there exist in $\mathscr{C}'_{sec}$ a codeword of weight $2p$?

The reason for this is explained by the following proposition.

Proposition 33. *Let $I'_s \stackrel{def}{=} I_s \cap J$ for $s \in \{1,2\}$. If there exists a codeword $\boldsymbol{x}'$ in $\mathscr{C}'_{sec}$ such that $\mathsf{wt}(\boldsymbol{x}'_{I'_1}) = \mathsf{wt}(\boldsymbol{x}'_{I'_2}) = p$, then it will be the restriction of a codeword $\boldsymbol{x}$ in $\mathscr{C}_{sec}$ which will belong to the list $\mathcal{L}$ output by Algorithm 1.*

Proof. Consider a codeword $\boldsymbol{x}'$ in $\mathscr{C}'_{\text{sec}}$ such that $\mathsf{wt}(\boldsymbol{x}'_{I'_1}) = \mathsf{wt}(\boldsymbol{x}'_{I'_2}) = p$. For $s \in \{1,2\}$, extend $\boldsymbol{x}_{I'_s}$ with zeros on the other positions of I_s and let $\boldsymbol{x}_s$ be the corresponding word. Notice that $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ will be considered by Algorithm 1 and $\boldsymbol{x}_1$ will be stored at the address $\boldsymbol{H}_1\boldsymbol{x}_1^T$. By definition of $\boldsymbol{x}'$, $(\boldsymbol{x}_1||\boldsymbol{x}_2)$ is the restriction of a codeword $\boldsymbol{x}$ of $\mathscr{C}_{\text{sec}}$ to I, say $\boldsymbol{x} = (\boldsymbol{x}_1||\boldsymbol{x}_2||\boldsymbol{y})$ with $\boldsymbol{y} \in \mathbb{F}_2^{N-K-l}$. Since $\mathscr{C}_{\text{sec}} \subset \mathscr{C}_{\text{pub}}$ we have $\hat{\boldsymbol{H}}\boldsymbol{x}^T = 0$. Let $\hat{\boldsymbol{H}}'$ be the matrix obtained from $\hat{\boldsymbol{H}}$ put in quasi-systematic form through a Gaussian elimination as given in Figure 11.4. We also have $\hat{\boldsymbol{H}}'\boldsymbol{x}^T = 0$ and hence:

$$\boldsymbol{H}_1\boldsymbol{x}_1^T + \boldsymbol{H}_2\boldsymbol{x}_2^T = 0 \tag{11.5}$$

and

$$\boldsymbol{H}_3(\boldsymbol{x}_1||\boldsymbol{x}_2)^T + \boldsymbol{y}^T = 0. \tag{11.6}$$

Equation (11.5) shows that $\boldsymbol{x}_1$ is stored at address $\boldsymbol{H}_2\boldsymbol{x}_2^T$ and will be considered at Step 8 of the algorithm. In this case, $\boldsymbol{x}$ will be stored in $\mathcal{L}$. □

We expect that the dimension of $\mathscr{C}'_{\text{sec}}$ is still k and that this code behaves like a random code of the same length and dimension. Ignoring the unessential issue whether or not $\boldsymbol{x}'$ satisfies $\mathsf{wt}(\boldsymbol{x}'_{I'_1}) = \mathsf{wt}(\boldsymbol{x}'_{I'_2}) = p$, let us just assume that there exists $\boldsymbol{x}'$ in $\mathscr{C}'_{\text{sec}}$ such that $|\boldsymbol{x}'| = 2p$. There is a non negligible chance that we have $\mathsf{wt}(\boldsymbol{x}'_{I'_1}) = \mathsf{wt}(\boldsymbol{x}'_{I'_2}) = p$ and that this codeword will be found by our algorithm. The issue is therefore whether or not there is a codeword of weight $2p$ in a random code of dimension k and length $|I'|$. This holds with a good chance (see [BF02] for instance) as soon as:

$$2p \geqslant d_{\text{GV}}(|I'|, k) \tag{11.7}$$

where $d_{\text{GV}}(|I'|, k)$ denotes the Gilbert-Varshamov distance of a code of length $|I'|$ and dimension k. Recall that [MS86]:

$$d_{\text{GV}}(|I'|, k) \approx h^{-1}\left(1 - k/|I'|\right)|I'|$$

where $h^{-1}(x)$ is the inverse function defined over $[0, \frac{1}{2}]$ of the binary entropy function $h(x) \stackrel{\text{def}}{=} -x\log_2 x - (1-x)\log_2(1-x)$. Recall that we expect to have:

$$|I'| \approx (R + \alpha\rho + \lambda)n,$$

which implies

$$\frac{k}{|I'|} \approx \frac{\rho}{R + \alpha\rho + \lambda} \approx \frac{\rho}{R}$$

when α and λ are small. Roughly speaking, to avoid such an attack, several conditions have to be met:

1. ρ has to be significantly smaller than R,
2. n has to be large enough.

This phenomenon was clearly not taken into account in the parameters suggested in [KKS97, KKS05, BMJ11] as shown in Table 11.1. The values of $d_{\text{GV}}(|I'|, k)$ are extremely low (in the range $1-6$). In other words, taking $p = 1$ is already quite threatening for all these schemes. For the first parameter set, namely $(k, n, K, N) = (60, 1023, 192, 3000)$, this suggests to take $p = 3$. Actually taking $p = 1$ is already enough to break the scheme. The problem with these low values of p comes from the dependency of the complexity in p as detailed in the following section. For instance as long as p is smaller than 3 the complexity of one iteration is dominated by the Gaussian elimination Step 2.

Finally, let us observe that when this attack gives a message/signature pair, it can be used as a bootstrap for an attack that recovers the whole private key as will be explained in the following subsection.

Table 11.1: KKS Parameters with the corresponding value of $d_{\text{GV}}(n', k)$.

Article	ρ	n	l	$n' \stackrel{\text{def}}{=} \mathbb{E}\{\lvert I'\rvert\}$	R	N	$d_{\text{GV}}(n', k)$
[KKS97]	$\frac{60}{1023} \approx 0.059$	1,023	8	89	$\frac{192}{3000} \approx 0.064$	3,000	6
[KKS05]	$\frac{48}{255} \approx 0.188$	255	8	65	$\frac{273}{1200} \approx 0.228$	1,200	5
[KKS97]	$\frac{48}{180} \approx 0.267$	180	8	64	$\frac{335}{1100} \approx 0.305$	1,100	4
[BMJ11]	$1/2$	320	12	165	$1/2$	11,626	1
[BMJ11]	$1/2$	448	13	230	$1/2$	16,294	1
[BMJ11]	$1/2$	512	13	264	$1/2$	18,586	1
[BMJ11]	$1/2$	768	13	395	$1/2$	27,994	2
[BMJ11]	$1/2$	1,024	14	527	$1/2$	37,274	2

11.4.4 Exploiting a signature for extracting the private key

If a signature σ of a message $\boldsymbol{x}$ is known, then $\boldsymbol{y} \stackrel{\text{def}}{=} (\sigma, \boldsymbol{x})$ is a codeword of $\mathscr{C}_{\text{sec}}$ which has weight about $n/2$ when restricted to its N first positions. This yields almost half of the positions of J. This can be exploited as follows. We perform the same attack as in the previous subsection, but we avoid choosing positions i for which $\sigma_i = 1$. More precisely, if we let $J_\sigma \stackrel{\text{def}}{=} \mathsf{supp}(\sigma) = \{i : \sigma_i = 1\}$, then we choose $K + k + l$ positions among $[1 \cdots N] \setminus J_\sigma$ to form I. The point of this choice is that we have more chances to have a smaller size for $I' = I \cap J$. Let $n' \stackrel{\text{def}}{=} |I'|$, we have now:

$$\mathbb{E}\left\{n' \,|J_\sigma\right\} = \frac{n - |J_\sigma|}{N - |J_\sigma|}(k + K + l) \tag{11.8}$$

$$\mathbb{E}\left\{|I'|\right\} = \mathbb{E}\left\{\mathbb{E}\left\{n' \,|J_\sigma\right\}\right\} \approx \frac{n/2}{(N - n/2)}(k + K + l). \tag{11.9}$$

The last approximation follows from the fact that the weight $\mathsf{wt}(\sigma)$ is quite concentrated around $n/2$. The same reasoning can be made as before, but the odds that the algorithm finds other valid signatures are much higher. This comes from the fact that the expectation $|I'|$ is half the expected size of I' in the previous case as given in Equation (11.3). Previously we had $\mathbb{E}\left\{\frac{|I'|}{k}\right\} \approx \frac{R}{\rho}$, whereas now we have:

$$\mathbb{E}\left\{\frac{|I'|}{k}\right\} \approx \frac{R}{2\rho}.$$

In other words, in order to avoid the previous attack we had to take ρ significantly smaller than R and now, we have to take ρ significantly smaller than $R/2$. For all the parameters proposed in the past, it turns out that $d_{\text{GV}}(|I'|, k)$ is almost always equal to 1, which makes the attack generally successful in just one iteration by choosing $p = 1$.

Moreover, if another valid signature σ' is obtained and by taking the union $J_\sigma \cup J_{\sigma'}$ of the supports, then about $3/4$ of the positions of J will be revealed. We can start again the process of finding other message/signature pairs by choosing $K + k + l$ positions among $\{1, 2, \ldots, N\} \setminus (J_\sigma \cup J_{\sigma'})$ to form the sets I. This approach can be iterated as explained in Algorithm 2. This process will quickly reveal the whole set J and from this, the private key is easily extracted as detailed in [COV07].

Algorithm 2 Recovering the private key from $t \geqslant 1$ signatures.

PARAMETERS:

r : number of iterations

l : small integer ($l \leqslant 40$)

p : very small integer ($1 \leqslant p \leqslant 4$).

INPUT:

$\hat{\boldsymbol{H}}$: public matrix as defined in Figure 11.3

$\{\sigma_1, \ldots, \sigma_t\}$: list of $t \geqslant 1$ valid signatures

OUTPUT: $J \subset [1 \cdots N]$ of cardinality n

$J \leftarrow \cup_{i=1}^{t} \mathsf{supp}(\sigma_i)$
repeat
  $S \leftarrow [1 \cdots N] \setminus J$
  $\mathcal{L} \leftarrow \mathsf{KKSforge}(r,l,p,S,\hat{\boldsymbol{H}})$
  for all $\sigma \in \mathcal{L}$ **do**
    $J \leftarrow J \cup \mathsf{supp}(\sigma)$
  end for
until $|J| = n$
return J

Finally, let us focus on the variant proposed in [BMJ11]. In this case, we have slightly less information than in the original KKS scheme. This can be explained by the following reasoning. In this case too, we choose S again as $[1 \cdots N] \setminus J_\sigma$, where as before J_σ is defined as $J_\sigma \stackrel{\text{def}}{=} \{i : \sigma_i = 1\}$. However this time, by defining n' again as $n' \stackrel{\text{def}}{=} |I'|$, we have

$$\mathbb{E}\left\{n' \,|J_\sigma\right\} = \frac{|J'_\sigma|}{N - |J_\sigma|}(k + K + l)$$

where

$$J'_\sigma = J \setminus J_\sigma.$$

However, this time due to the noise which is added, $|J_\sigma|$ is expected to be larger than before (namely of order $\frac{n}{2} + \frac{(N-n)n}{N}$).

11.5 Analysis of the Attack

The purpose of this section is to provide a very crude upper-bound on the complexity of the attack. We assume here that the code $\mathscr{C}_{\text{rand}}$ of length n which is equal to the restriction on J of $\mathscr{C}_{\text{sec}}$:

$$\mathscr{C}_{\text{rand}} \stackrel{\text{def}}{=} \left\{ (x_j)_{j\in J} \ : \ \boldsymbol{x} = (x_1, \ldots, x_{N+k}) \in \mathscr{C}_{\text{sec}} \right\}$$

behaves like a random code. More precisely we assume that it has been chosen by picking a random parity-check matrix $\boldsymbol{H}_{\text{rand}}$ of size $(n-k) \times n$ (by choosing its entries uniformly at random among $\mathbb{F}_2$). This specifies a code $\mathscr{C}_{\text{rand}}$ of length n as $\mathscr{C}_{\text{rand}} = \{\boldsymbol{x} \in \mathbb{F}_2^n : \boldsymbol{H}_{\text{rand}}\boldsymbol{x}^T = 0\}$. We first give in the following section some quite helpful lemmas about codes of this kind.

11.5.1 Preliminaries about random codes

We are interested in this section in obtaining a lower bound on the probability that a certain subset X of $\mathbb{F}_2^n$ has a non empty intersection with $\mathscr{C}_{\text{rand}}$. For this purpose, we first calculate the two following probabilities. The probabilities are taken here over the random choices of $\boldsymbol{H}_{\text{rand}}$.

Lemma 2. *Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be two different and nonzero elements of $\mathbb{F}_2^n$. Then*

$$\mathbf{prob}(\boldsymbol{x} \in \mathscr{C}_{rand}) = 2^{k-n} \tag{11.10}$$

$$\mathbf{prob}(\boldsymbol{x} \in \mathscr{C}_{rand}, \boldsymbol{y} \in \mathscr{C}_{rand}) = 2^{2(k-n)} \tag{11.11}$$

To prove this lemma, we will introduce the following notation and lemma. For $\boldsymbol{x} = (x_i)_{1\leqslant i\leqslant s}$ and $\boldsymbol{y} = (y_i)_{1\leqslant i\leqslant s}$ being two elements of $\mathbb{F}_2^s$ for some arbitrary s, we define $\boldsymbol{x}\cdot\boldsymbol{y}$ as

$$\boldsymbol{x}\cdot\boldsymbol{y} = \sum_{1\leqslant i\leqslant s} x_i y_i,$$

the addition being performed over $\mathbb{F}_2$.

Lemma 3. *Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be two different and nonzero elements of $\mathbb{F}_2^n$ and choose $\boldsymbol{h}$ uniformly at random in $\mathbb{F}_2^n$, then*

$$\mathbf{prob}(\boldsymbol{x}\cdot\boldsymbol{h} = 0) = \frac{1}{2} \tag{11.12}$$

$$\mathbf{prob}(\boldsymbol{x}\cdot\boldsymbol{h} = 0, \boldsymbol{y}\cdot\boldsymbol{h} = 0) = \frac{1}{4} \tag{11.13}$$

Proof. To prove Equation (11.12) we just notice that the subspace $\{\boldsymbol{h} \in \mathbb{F}_2^n : \boldsymbol{x}\cdot\boldsymbol{h} = 0\}$ is of dimension $n-1$. There are therefore 2^{n-1} solutions to this equation and

$$\mathbf{prob}(\boldsymbol{x}\cdot\boldsymbol{h} = 0) = \frac{2^{n-1}}{2^n} = \frac{1}{2}.$$

On the other hand, the hypothesis made on $\boldsymbol{x}$ and $\boldsymbol{y}$ implies that $\boldsymbol{x}$ and $\boldsymbol{y}$ generate a subspace of dimension 2 in $\mathbb{F}_2^n$ and that the dual space, that is $\{\boldsymbol{h} \in \mathbb{F}_2^n : \boldsymbol{x}\cdot\boldsymbol{h} = 0, \boldsymbol{y}\cdot\boldsymbol{h} = 0\}$ is of dimension $n-2$. Therefore

$$\mathbf{prob}(\boldsymbol{x}\cdot\boldsymbol{h} = 0, \boldsymbol{y}\cdot\boldsymbol{h} = 0) = \frac{2^{n-2}}{2^n} = \frac{1}{4}$$

□

Proof of Lemma 2. Let $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{n-k}$ be the $n-k$ rows of $\boldsymbol{H}_{\text{rand}}$. Then

$$\begin{aligned}
\mathbf{prob}(\boldsymbol{x} \in \mathscr{C}_{\text{rand}}) &= \mathbf{prob}(\boldsymbol{H}_{\text{rand}}\boldsymbol{x}^T = 0) \\
&= \mathbf{prob}(\boldsymbol{h}_1\cdot\boldsymbol{x} = 0, \ldots, \boldsymbol{h}_{n-k}\cdot\boldsymbol{x} = 0) \\
&= \mathbf{prob}(\boldsymbol{h}_1\cdot\boldsymbol{x} = 0)\ldots\mathbf{prob}(\boldsymbol{h}_{n-k}\cdot\boldsymbol{x} = 0) \qquad (11.14) \\
&= 2^{k-n} \qquad (11.15)
\end{aligned}$$

where Equation (11.14) follows by the independence of the events and Equation (11.15) uses Lemma 3. Equation (11.11) is obtained in a similar fashion. □

Lemma 4. *Let X be some subset of $\mathbb{F}_2^n$ of size m and let f be the function defined by $f(x) \stackrel{def}{=} \max\left(x(1-x/2), 1-\frac{1}{x}\right)$. We denote by x the quantity $\frac{m}{2^{n-k}}$, then*

$$\mathbf{prob}(X \cap \mathscr{C}_{rand} \neq \emptyset) \geqslant f(x).$$

Proof. For $\boldsymbol{x}$ in X we define $E_{\boldsymbol{x}}$ as the event "$\boldsymbol{x}$ belongs to $\mathscr{C}_{\text{rand}}$" and we let $q \stackrel{\text{def}}{=} 2^{k-n}$. We first notice that

$$\mathbf{prob}(X \cap \mathscr{C}_{\text{rand}} \neq \emptyset) = \mathbf{prob}\left(\bigcup_{\boldsymbol{x}\in X} E_{\boldsymbol{x}}\right).$$

By using the Bonferroni inequality [Com74, p. 193] on the probability of the union of events we obtain

$$\begin{aligned}
\mathbf{prob}\left(\bigcup_{\boldsymbol{x}\in X} E_{\boldsymbol{x}}\right) &\geqslant \sum_{\boldsymbol{x}\in X} \mathbf{prob}(E_{\boldsymbol{x}}) - \sum_{\{\boldsymbol{x},\boldsymbol{y}\}\subset X} \mathbf{prob}(E_{\boldsymbol{x}} \cap E_{\boldsymbol{y}}) && (11.16)\\
&\geqslant mq - \frac{m(m-1)}{2} q^2 && (11.17)\\
&\geqslant mq - \frac{m^2 q^2}{2}\\
&\geqslant mq(1 - mq/2),
\end{aligned}$$

where (11.17) follows from Lemma 2. This bound is rather sharp for small values of mq. On the other hand for larger values of mq, another lower bound on $\mathbf{prob}(X \cap \mathscr{C}_{\text{rand}} \neq \emptyset)$ is more suitable [dC97]. It gives

$$\begin{aligned}
\mathbf{prob}\left(\bigcup_{\boldsymbol{x}\in X} E_{\boldsymbol{x}}\right) &\geqslant \sum_{\boldsymbol{x}\in X} \frac{\mathbf{prob}(E_{\boldsymbol{x}})^2}{\sum_{\boldsymbol{y}\in X} \mathbf{prob}(E_{\boldsymbol{x}} \cap E_{\boldsymbol{y}})} && (11.18)\\
&\geqslant \frac{mq^2}{q + (m-1)q^2} && (11.19)\\
&\geqslant 1 - \frac{1}{mq},
\end{aligned}$$

By taking the maximum of both lower bounds, we obtain our lemma. □

11.5.2 Estimating the complexity of Algorithm 1

Here we estimate how many iterations have to be performed in order to break the scheme when no signature is known and when $S = [1 \cdots N]$. For this purpose, we start by lower-bounding the probability that an iteration is successful. Let us bring the following random variables for $i \in \{1, 2\}$:

$$I'_i \stackrel{\text{def}}{=} I_i \cap J \quad \text{and} \quad W_i \stackrel{\text{def}}{=} |I'_i|\,.$$

By using Lemma 33, we know that an iteration finds a valid signature when there is an $\boldsymbol{x}$ in $\mathscr{C}_{\text{sec}}$ such that

$$|\boldsymbol{x}_{I'_1}| = |\boldsymbol{x}_{I'_2}| = p.$$

Therefore the probability of success P_{succ} is lower bounded by

$$\sum_{w_1+w_2\leqslant n} \mathbf{prob}\left\{\exists \boldsymbol{x} \in \mathscr{C}_{\text{sec}} : |\boldsymbol{x}_{I'_1}| = \left|\boldsymbol{x}_{I'_2}\right| = p | W_1 = w_1, W_2 = w_2\right\} \mathbf{prob}(W_1 = w_1, W_2 = w_2) \tag{11.20}$$

On the other hand, by using Lemma 4 with the set

$$X \stackrel{\text{def}}{=} \left\{\boldsymbol{x} = (x_j)_{j\in J} : |\boldsymbol{x}_{I'_1}| = \left|\boldsymbol{x}_{I'_2}\right| = p\right\}$$

which is of size $\binom{w_1}{p}\binom{w_2}{p}2^{n-w_1-w_2}$, we obtain

$$\mathbf{prob}\left\{\exists \boldsymbol{x} \in \mathscr{C}_{\text{sec}} : |\boldsymbol{x}_{I'_1}| = \left|\boldsymbol{x}_{I'_2}\right| = p | W_1 = w_1, W_2 = w_2\right\} \geqslant f(x). \tag{11.21}$$

with

$$x \stackrel{\text{def}}{=} \frac{\binom{w_1}{p}\binom{w_2}{p}2^{n-w_1-w_2}}{2^{n-k}} = \binom{w_1}{p}\binom{w_2}{p}2^{k-w_1-w_2}$$

The first quantity is clearly equal to

$$\mathbf{prob}(W_1 = w_1, W_2 = w_2) = \frac{\binom{n}{w_1}\binom{n-w_1}{w_2}\binom{N-n}{(K+k+l)/2-w_1}\binom{N-n-(K+k+l)/2+w_1}{(K+k+l)/2-w_2}}{\binom{N}{(K+k+l)/2}\binom{N-(K+k+l)/2}{(K+k+l)/2}}. \tag{11.22}$$

Plugging in the expressions obtained in (11.21) and (11.22) in (11.20) we have an explicit expression of a lower bound on P_{succ}. The number of iterations for our attack to be successful is estimated to be of order $\frac{1}{P_{\text{succ}}}$. We obtain therefore an upper-bound on the expected number of iterations, what we denote by `UpperBound`. Table 11.2 shows for various KKS parameters, p and l the expected number of iterations.

Table 11.2: KKS Parameters with the corresponding value of $\frac{1}{P_{\text{succ}}}$.

Article	ρ	n	l	p	$n' \stackrel{\text{def}}{=} \mathbb{E}\{\lvert I' \rvert\}$	R	N	UpperBound
[KKS97]	$\frac{60}{1023} \approx 0.059$	1,023	8	1	91	$\frac{192}{3000} \approx 0.064$	3,000	111.26
	$\frac{60}{1023} \approx 0.059$	1,023	14	2	91	$\frac{192}{3000} \approx 0.064$	3,000	14.17
[KKS05]	$\frac{48}{255} \approx 0.188$	255	8	1	66	$\frac{273}{1200} \approx 0.228$	1,200	26.41
	$\frac{48}{255} \approx 0.188$	255	14	2	66	$\frac{273}{1200} \approx 0.228$	1,200	4.37
[KKS97]	$\frac{48}{180} \approx 0.267$	180	8	1	65	$\frac{335}{1100} \approx 0.305$	1,100	6.07
	$\frac{48}{180} \approx 0.267$	180	15	2	65	$\frac{335}{1100} \approx 0.305$	1,100	1.82
[BMJ11]	$1/2$	320	12	1	165	$1/2$	11,626	1.24
[BMJ11]	$1/2$	448	13	1	230	$1/2$	16,294	1.34
[BMJ11]	$1/2$	512	13	1	264	$1/2$	18,586	1.39
[BMJ11]	$1/2$	768	13	1	395	$1/2$	27,994	1.61
[BMJ11]	$1/2$	1,024	14	1	527	$1/2$	37,274	1.85

11.5.3 Number of operations of one iteration

The complexity of one iteration of Algorithm 1 is $C(p,l) = C_{\text{Gauss}} + C_{\text{hash}} + C_{\text{check}}$ where C_{Gauss} is the complexity of a Gaussian elimination, C_{hash} is the complexity of hashing all the $\boldsymbol{x}_1$'s and C_{check} is the complexity of checking all the $\boldsymbol{x}_2$'s with the following expressions:

$$C_{\text{Gauss}} = O\Big((N+k)(N-k)(N-k-l)\Big) = O(N^3) \tag{11.23}$$

$$C_{\text{hash}} = O\left(\binom{(K+k+l)/2}{p}\right) \tag{11.24}$$

$$C_{\text{check}} = O\left(\frac{1}{2^l}(N-K-l)^2\binom{(K+k+l)/2}{p}^2\right) \tag{11.25}$$

The last expression giving C_{check} comes from the fact that the algorithm considers $\binom{(K+k+l)/2}{p}$ elements $\boldsymbol{x}_2$, and for each of these candidates, we check about $O\left(\frac{1}{2^l}\binom{(K+k+l)/2}{p}\right)$ elements $\boldsymbol{x}_1$'s, which involves a matrix multiplication in Step 9. Let us note that l will be chosen such that C_{hash} and C_{check} are roughly of the same order, say $2^l \approx \binom{(K+k+l)/2}{p}$.

11.6 Experimental Results

The attack described in Section 11.4 was implemented in C and was run on a laptop MacBook Pro with an Intel Core i7 of 2.66 GHz to validate the analysis developed in Section 11.5. Table 11.3 presents the average number of iterations that were necessary to obtain a codeword of weight in the range $[t_1 \cdots t_2]$. The average is computed with 4000 tests most of the time, with the exceptions of the penultimate entry (only 1000 tests) and the last entry (only 500 tests). The values of t_1 and t_2 are taken from [KKS97] and [BMJ11]. The algorithm halts whenever it finds a word in the prescribed set. Note that for [BMJ11], we have taken $t_1 = n/2 - \frac{3}{2}\sqrt{n}$ and $t_2 = n/2 + \frac{3}{2}\sqrt{n}$ as advocated by the authors. All the codes that we considered during our simulations were randomly chosen. This setting does not completely comply with the recommendations made by the authors for the schemes given in [KKS97]. In one case, it is suggested to use binary BCH codes of length $n = 255$ and dimension $k = 48$, and in another case a binary code of length $n = 180$ and dimension $k = 48$ that was constructed by means of 12 random binary equidistant codes of length 15, dimension 4 and minimum distance 8. However, we emphasize that these specific constraints are irrelevant because the attack is generic and only requires public data ($\boldsymbol{F}$ and $\boldsymbol{H}$) and aims at forging a valid signature. We can see in Table 11.3 that the number of iterations are in accordance with the theoretical upper-bound UpperBound on the value of $\frac{1}{P_{\text{succ}}}$ obtained in the previous section, which is an upper bound on the average number of iterations.

11.7 Concluding Remarks

Design principles. As explained in Section 11.3, the parameters of the KKS scheme were chosen in order to make decoding of $\mathscr{C}_{\text{known}}$ intractable when the weight of errors is in the range $[t_1 \cdots t_2]$, where $\mathscr{C}_{\text{known}}$ denotes the code defined by the parity-check matrix $\boldsymbol{H}$. In [BMJ11], it is further required that $\mathscr{C}_{\text{known}}$ is of minimum distance greater than $4n$. Both requirements are clearly insufficient to ensure that the scheme is secure as demonstrated by this paper. We suggest here to replace all these requirements by choosing the parameters such as to make our attack impracticable. This algorithm is exponential in nature when the parameters are well chosen. If we want to avoid that the knowledge of a message-signature pair allows to recover the secret key, this implies for instance that the rate R of $\mathscr{C}_{\text{known}}$ should be significantly larger than 2ρ, that is twice the rate of the secret code $\mathscr{C}_{\text{hidden}}$. This would change the parameters of the scheme significantly and give much larger key sizes than has been proposed in [KKS97, KKS05, BMJ11]. Choosing these parameters requires however to analyze properly the complexity of the attack when one message-signature is known (here we just analyzed the complexity of the attack which does not make use of any

message-signature pair). The analysis we performed in our case can be carried over to the case when a message-signature pair is known but this is beyond the scope of this paper and will appear in a full version of this paper.

Relating the security to the problem of decoding a linear code. The attack which has been suggested here is nothing but a well known algorithm for finding low weight codewords or for decoding a generic linear code. It just happens that this algorithm is much more powerful here than for a random linear code due to the peculiar nature of the code it is applied to. However as mentioned above, this attack is exponential in nature and can easily be defeated by choosing the parameters appropriately. It would be interesting to analyze the relationship of the problem of breaking the KKS scheme with decoding problems in more depth, or to prove that the problem which has to be solved is indeed NP hard.

Non-binary codes. Obviously there is a non binary version of the KKS scheme which would deal with codes defined over larger alphabets. The benefits of the generalized scheme are questionable. The attack presented here generalizes easily to higher order fields. What is more, moving to non-binary fields seems to be a poor idea in terms of security. For instance, whereas a message/signature pair reveals only half the positions of J in the binary case, in the q-ary case we expect to obtain roughly a fraction $\frac{q-1}{q}$ of positions of J, which is significantly larger.

Decoding one out of many. Another approach could have been used for attacking the scheme. Let us denote by $\boldsymbol{s}_1, \cdots, \boldsymbol{s}_k$ the columns of $\boldsymbol{F}$. These vectors can be considered as k syndromes of codewords of $\mathscr{C}_{\text{hidden}}$ with respect to the parity-check matrix $\boldsymbol{H}$. If we want to obtain one message/pair we can try to find an error $\boldsymbol{e}_i$ of weight in the range $[t_1 \cdots t_2]$ such that $\boldsymbol{H}\boldsymbol{e}_i^T = \boldsymbol{s}_i$. This suggests to use "the decoding one out of many" approach [Sen11], that is we have k words to decode and we want to decode at least one of them. This problem can be solved more efficiently than just decoding one instance. We can even refine this approach by considering all possible syndromes obtained by all possible (non-zero) combinations $\sum_i \alpha_i \boldsymbol{s}_i$. In this case, we would have to solve "a decoding one out of many" problem with $2^k - 1$ instances. However a naive use of the results of [Sen11] would be far from indicating that all the parameters of [KKS97, KKS05, BMJ11] are easily broken by this approach.

Table 11.3: Average umber of iterations of Algorithm 1.

Article	ρ	n	l	p	R	N	UpperBound	t_1	t_2	Iter.	Time (s)
[KKS97]	$\frac{60}{1023} \approx 0.059$	1,023	8	1	$\frac{192}{3000} \approx 0.064$	3,000	111.26	352	672	102.34	19.50
	$\frac{60}{1023} \approx 0.059$	1,023	14	2	$\frac{192}{3000} \approx 0.064$	3,000	14.17	352	672	9.22	1.754
[KKS05]	$\frac{48}{255} \approx 0.188$	255	8	1	$\frac{273}{1200} \approx 0.228$	1,200	26.41	48	208	24.32	0.384
	$\frac{48}{255} \approx 0.188$	255	14	2	$\frac{273}{1200} \approx 0.228$	1,200	4.37	48	208	3.13	0.051
[KKS97]	$\frac{48}{180} \approx 0.267$	180	8	1	$\frac{335}{1100} \approx 0.305$	1,100	6.07	96	96	5.58	0.061
	$\frac{48}{180} \approx 0.267$	180	15	2	$\frac{335}{1100} \approx 0.305$	1,100	1.82	96	96	1.23	0.017
[BMJ11]	$1/2$	320	12	1	$1/2$	11,626	1.24	133	187	1.13	6.425
[BMJ11]	$1/2$	448	13	1	$1/2$	16,294	1.34	192	256	1.18	18.90
[BMJ11]	$1/2$	512	13	1	$1/2$	18,586	1.39	222	290	1.26	32.23
[BMJ11]	$1/2$	768	13	1	$1/2$	27,994	1.61	342	426	1.51	119.5
[BMJ11]	$1/2$	1,024	14	1	$1/2$	37,274	1.85	464	560	1.73	350.8

Chapter 12

Conclusion and Perspectives

12.1 Algebraic Crytanalysis

Code-based cryptography is still an area in development in spite of its age. The field poses many questions and challenges that have to be studied. The first issue consists in better assessing key-recovery attacks of the McEliece cryptosystem [McE78] because unlike decoding attacks which has benefited from several results [McE78, LB88, Leo88, Ste88, vT90, CC94, CC95, Dum96, CC98, CS98, BLP08], the only known key-recovery attack amounts to performing an exhaustive search. As a result, the time complexity of any decoding attack is currently always lower than the best key-recovery attack.

The algebraic cryptanalysis that we introduced in [FOPT10a] is a first step towards a solving of this important question. In this approach, we define a bihomogeneous algebraic system the private key has to satisfy. For a binary Goppa code of length $n \leqslant 2^m$ and dimension k and code-rate $R = k/n$, the polynomial system has $2rk \geqslant 2(1-R)R\frac{n^2}{\log(n)}$ equations and $2n$ unknowns. Even if the system is heavily over-constrained (there are more equations than unknowns) and is highly structured (of Vandermonde form and the monomials are $Y_j X_j^a$ avec $0 \leqslant a \leqslant 2r-1$ et $1 \leqslant j \leqslant n$), the solving in the field extension $\mathbb{F}_{2^m}/\mathbb{F}_2$ of degree m is currently an open problem. The classical algorithms based on Gröbner bases [Buc65, CLO01, Fau99, Fau02] are not efficient with the current parameters. We have seen that the time complexity of this technique is $O\left(\binom{N+d}{d}^\omega\right)$ where ω is the "linear-algebra constant" *i.e.* $2 < \omega \leqslant 3$, N is the number of unknowns and d is the degree of regularity.

This observation prompts us to ask whether the algebraic cryptanalysis represents the appropriate framework for mounting efficient attacks. The recent results of [FSS11] show that for a square bilinear system (the number of equations is equal to the number of unknowns), the complexity of F_5 algorithm is upper-bounded by $O\left(\binom{N+d}{d}^\omega\right)$ with $N \stackrel{\text{def}}{=} n_Y + n_X$ where n_Y (*resp.* n_X) is the number of variables Y_j (*resp.* X_j) and $d \stackrel{\text{def}}{=} \min(n_Y+1, n_X+1)$.

This result motivates us to restrict ourselves to the subsystem only built with equations that contain monomials of the form $Y_j X_j^{2^b}$ with $0 \leqslant b \leqslant \log_2(2r-1)$ where r is the degree of the Goppa polynomial. This leads to a quadratic system by replacing each variable X_j by m binary variables $X_{j,1}, \ldots, X_{j,m}$ so that the number of variables $X_{j,l}$ becomes mn. The system still has n variables Y_j. In that particular context, the degree of regularity d is then $n-k$ for a binary Goppa code (or alternant code) of dimension k and length n, and $N = 2n-k = (2-R)n$. However, the system is not square because it has at least $Rn\log_2(2r-1)$ equations. Usually in cryptography, the parameters are chosen such that $R \geqslant 1/2$ and $r \geqslant 9$, which implies $N \leqslant \frac{3}{2}n$ whereas $Rn\log_2(2r-1) > 2n$. Therefore, this situation requires to assess exactly what is the impact on the complexity of F_5 when the number of equations increases. This study would permit in particular to propose an tighter analysis of the attacks proposed in [FOPT10a] against variants of [McE78]. Let us recall that we have shown in [FOPT10a] that it is possible to recover efficiently the private key of [Gab05, BCGO09, MB09]. The main reasons of this outcome are firstly the number of variables is very low, and the number of equations is very high.

The existence of algebraic methods raises the fundamental question of whether it will bring a real breakthrough approach in cryptanalyzing McEliece-like schemes. A first and natural step for answering it is to compare with existing attacks that recover the private keys. The most interesting attack to compare with is the Sidelnikov and Shestakov attack [SS92] which recovers very efficiently the secret codes when it is a Generalized Reed-Solomon code. The complexity of the attack is $\mathcal{O}(n^3)$ where n is the length of the code. If the algebraic cryptanalysis is applied to that precise case and if we assume that the dimension of Generalized Reed-Solomon code is $k = n-r$ where r is a nonzero integer $< n$, the polynomial system would have n unknowns X_j, r free variables Y_j and $k(r-1)$ equations. The key argument to assess of the efficiency of this attack is to determine th value of the degree of regularity d of the system. In particular, if d is very small or more generally upper-bounded by a constant, it would give a new polynomial attack.

Another possible strategy that would enable to measure the input of the algebraic approach, and more importantly, to determine the area inside which it becomes efficient, is to consider simplified system by for instance decreasing the number of unknowns. One could think for example to alternant codes obtained with $y_j = 1$. In that context, the number of variables then becomes $n-k$ which corresponds to only variables X_j.

12.2 Code Equivalence Problem

The algebraic approach defined in [FOPT10a] can be exploited to study an important problem in coding theory, namely the *code equivalence* problem (Definition 9). We rephrase it in the following manner:

Given two codes $\mathscr{A}$ and $\mathscr{B}$ having the same length n and the same dimension k, is there a permutation π in the symmetric group of order n denoted by $\mathfrak{S}_n$ such that for any $\boldsymbol{a} = (a_1, \ldots, a_n)$ in $\mathscr{A}$, we have always have that $(a_{\sigma^{-1}(1)}, \ldots, a_{\sigma^{-1}(n)})$ belongs to $\mathscr{B}$?

This issue is essential in the classification of codes having the same parameters (length, dimension, minimum distance are all the same). The article [PR97] proves that *Graph Isomorphism* problem reduces to it. On the other hand, Sendrier's algorithm [Sen00] solves it with a time complexity exponential in the dimension of $\mathscr{A} \cap \mathscr{A}^{\perp}$ and polynomial in n.

We propose another way of studying the code equivalence problem based on the solving of a quadratic polynomial system. Let us assume that $\mathscr{A}$ is defined by a generator matrix $\boldsymbol{A}$ and $\mathscr{B}$ is defined by a parity-check matrix $\boldsymbol{B}$. Let us denote by $\boldsymbol{X} = (x_{i,j})$ a $n \times n$ square matrix which is expected to represent a permutation matrix. We then have the following equality:

$$\boldsymbol{X}\boldsymbol{X}^T = \boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{I}_n. \tag{12.1}$$

If there exists a permutation (matrix) that sends $\mathscr{A}$ onto $\mathscr{B}$, it should satisfy the linear system:

$$\boldsymbol{A}\boldsymbol{X}\boldsymbol{B}^T = \boldsymbol{0}$$

where $\boldsymbol{B}^T$ is the transpose of $\boldsymbol{B}$. Moreover, we can use the fact that $\boldsymbol{X}$ is necessarily a binary matrix, which is equivalent to say that $x_{i,j}^2 = x_{i,j}$ for any i and j, and it should also satisfy the following polynomial equations obtained for any couples (i, i') et (j, j') such that $i \neq i'$ and $j \neq j'$:

$$x_{i,j}x_{i,j'} = 0 \quad \text{and} \quad x_{i,j}x_{i',j} = 0$$

We can observe that if we also add the linear equations $\sum_{j=1}^{n} x_{i,j} = 1$ and $\sum_{i=1}^{n} x_{i,j} = 1$ with $1 \leqslant i \leqslant n$ then (12.1) can be safely removed from the polynomial system. Consequently, the number of linear equations is $k(n-k) + 2n = n^2\left(R(1-R) + \frac{2}{n}\right)$ with $R \stackrel{\text{def}}{=} k/n$. By assuming[1] that these equations are independent, it enables to reduce the number of unknowns to $N \stackrel{\text{def}}{=} n^2 - n^2\left(R(1-R) + \dfrac{2}{n}\right) = n^2\left(R^2 - R + 1 - \dfrac{2}{n}\right)$. On the other hand the number of quadratic equations denoted by M is given by $n^2 + 2n\binom{n}{2} = n^3$. So by writing that $n^3 = \left(\dfrac{N}{R^2 - R + 1 - \frac{2}{n}}\right)^{\frac{3}{2}}$ we see that $\dfrac{M}{N} \geqslant \dfrac{1}{2\sqrt{2}}\sqrt{N}$. The key parameter for measuring the efficiency is the degree of regularity. In the article [BFSY05], an asymptotic analysis is proposed when the ration M/N is constant. It is therefore interesting to generalise this result to cases where the ratio is larger.

12.3 Reductionist Security

The design of cryptographic primitives has to imperatively rely on convincing arguments accrediting that the proposed schemes are sure. Since, the apparition of the McEliece scheme, code-based primitive designers' practise is to suggest schemes and prove the security by showing that all the existing attacks are inefficient. Our works in this thesis show that this way of thinking is not *safe* and does not give a formal proof that the scheme is secure. Indeed, no guarantee is given that there won't be a better attack in the future.

Reductionist security addresses this matter by showing that if an attacker succeeds in impairing a wished security notion there would exist a (polynomial) transformation resulting in solving a hard problem. This methodology has the merit of identifying algorithmic problems whose difficulty can be evaluated independently.

Currently, the McEliece cryptosystem has not undergone any attack that would show that encryption function is not one-way. But, the unsuccessful attempts that propose encryption schemes copying McEliece's general idea prompts researchers to provide reductionist security proofs. Recently, [DMQN09] shows that it is possible to convert the McEliece scheme into a semantically secure one in the standard model against an adaptative Chosen Ciphertext Attack (IND-CCA2) under the assumption that the problem of decoding random linear codes and Goppa code distinguishing problem are difficult. The problem of decoding random linear codes is an old problem that received a lot of attention these last years and hence is widely accepted as a hard problem. But the Goppa code distinguishing problem appeared quite recently in [CFS01] and consequently few things are known about its difficulty. However, it becomes the centerpiece for elaborating a reductionist security proof for any trapdoor function whose trapdoor is a binary Goppa code.

Under certain conditions, we have shown that is possible to solve efficiently the Goppa code distinguishing problem. The idea is to compute the dimension D of solution space of a linear system deduced from the algebraic system that any alternant code should verify. It turns out that the value of D is huge and varies depending on the type of code, namely a generic alternant

[1] It is a realistic assumption for random linear codes.

code, Goppa code or random code. This surprising outcome has however a limitation. Indeed, the code rate has to be very close to 1. This kind of code are well-suited for the CFS signature scheme [CFS01] whereas for encryption scheme the preferred codes have code rates that are less than 1, and hence do not fit in the area of validity of our distinguisher.

But the existence of such a distinguisher has at least one consequence. As already observed, even if it does not question neither the security of the scheme [McE78] nor the security reduction given in [DMQN09], it does invalidate the security reduction of the CFS signature scheme. The situation was already critical for this scheme because the recommended parameters [FS09] lead to extremely large keys. *In other words, the area of code-based cryptography returns to a situation where there is no efficient and provably secure signature scheme.*

Furthermore, it is natural to ask whether we can extend the range of validity of the distinguisher. Ideally, one would like to find a method that would distinguish a Goppa code of any rate. One first idea is to generalize the linearization technique. Unfortunately, there is little chance that this succeeds because we can show that any linear system obtained by this way is essentially equivalent. Therefore, it seems that a totally new approach has to be designed in order to extend the distinguisher.

Another fundamental question is to know if one can devise a key-recovery attack derived from the distinguisher, in particular against the CFS scheme. This phenomenon has already happened in the past for the SFLASH signature scheme which was broken thanks to the existence of a distinguisher. This point needs to be investigated and it undoubtedly requires new ideas to address it.

Eventually, even if there is no attack based on the distinguisher and even if the distinguisher does not cover the parameters one encounters in the McEliece scheme, its existence might throw doubt on the interest of using the Goppa code distinguishing assumption. But this problem is deeply linked to the problem of decoding random linear codes. It seems that it is impossible to get rid of it whenever one seeks to a security reduction from the problem of decoding random linear codes. The Goppa code distinguishing problem actually represents the link between an ideal object, which is the random code, and a real object, namely the Goppa code. One easily understands that getting rid of the Goppa code distinguishing problem means to define a new problems [Dal10]. One solution is to directly claim that the McEliece function is one-way like it is done in the case of RSA cryptosystem. This area of research deserves to be investigated in order to prove for instance that the McEliece scheme is IND-CCA2 in the standard model under the assumption that the McEliece problem (Definition 14) is difficult. This also implies to study it further and in particular to pursue the study of general decoding algorithms. This field of research has currently received a renewed interest [MMT11]. We know that practically these algorithms do not represent a real threat once parameters are generated to resist them. However, the attack we developed against the KKS signature scheme shows that this field should be active, and one very important question is to know whether these algorithms can be improved by exploiting the property that the underlying code has an algebraic structure.

Bibliography

[Bar98] A. Barg. Complexity issues in coding theory. In V. Pless and W.C. Huffman, editors, *Handbook of Coding Theory*, volume 1, pages 649–754. Elsevier Science, 1998.

[Bar04] M. Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie*. PhD thesis, Université de Paris VI, 2004.

[BBC08] M. Baldi, M. Bodrato, and G.F. Chiaraluce. A new analysis of the McEliece cryptosystem based on QC-LDPC codes. In *Security and Cryptography for Networks (SCN)*, pages 246–262, 2008.

[BC07] M. Baldi and G. F. Chiaraluce. Cryptanalysis of a new instance of McEliece cryptosystem based on QC-LDPC codes. In *IEEE International Symposium on Information Theory*, pages 2591–2595, Nice, France, March 2007.

[BCGO09] T. P. Berger, P.L. Cayrel, P. Gaborit, and A. Otmani. Reducing key length of the McEliece cryptosystem. In Bart Preneel, editor, *Progress in Cryptology - Second International Conference on Cryptology in Africa (AFRICACRYPT 2009)*, volume 5580 of *Lecture Notes in Computer Science*, pages 77–97, Gammarth, Tunisia, June 21-25 2009.

[BCP97] W. Bosma, J. J. Cannon, and Catherine Playoust. The Magma algebra system I: The user language. *J. Symb. Comput.*, 24(3/4):235–265, 1997.

[Ber97] T. Berson. Failure of the mceliece public-key cryptosystem under message-resend and related-message attack. In Burton Kaliski, editor, *Advances in Cryptology – CRYPTO 97*, volume 1294 of *Lecture Notes in Computer Science*, pages 213–220. Springer Berlin / Heidelberg, 1997.

[BF02] A. Barg and G. D. Forney. Random codes: Minimum distances and error exponents. *IEEE Transactions on Information Theory*, 48(9):2568–2573, September 2002.

[BFS02] M. Bardet, J.C. Faugère, and B. Salvy. Complexity study of Gröbner basis computation. Technical report, INRIA, 2002. `http://www.inria.fr/rrrt/rr-5049.html`.

[BFS04] M. Bardet, J.C. Faugère, and B. Salvy. On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In *Proc. International Conference on Polynomial System Solving (ICPSS)*, pages 71–75, 2004.

[BFSY05] M. Bardet, J.C. Faugère, B. S., and B.Y. Yang. Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems. In *Proc. of MEGA 2005, Eighth International Symposium on Effective Methods in Algebraic Geometry*, 2005.

[BL04] T. P. Berger and P. Loidreau. Designing an efficient and secure public-key cryptosystem based on reducible rank codes. In *INDOCRYPT*, volume 3348 of *LNCS*, pages 218–229, 2004.

[BL05] T. P. Berger and P. Loidreau. How to mask the structure of codes for a cryptographic use. *Designs Codes and Cryptography*, 35(1):63–79, 2005.

[BLM11] P. S. L. M. Barreto, R. Lindner, and R. Misoczki. Monoidic codes in cryptography. Cryptology ePrint Archive, Report 2011/371, 2011.

[BLP08] D. J. Bernstein, T. Lange, and C. Peters. Attacking and defending the McEliece cryptosystem. In *PQCrypto*, volume 5299 of *LNCS*, pages 31–46, 2008.

[BLP11] D. J. Bernstein, T. Lange, and C. Peters. Smaller decoding exponents: ball-collision decoding. In *Proceedings of Crypto 2011*, 2011.

[BMJ11] P. S.L.M. Barreto, R. Misoczki, and M. A. Simplicio Jr. One-time signature scheme from syndrome decoding over generic error-correcting codes. *Journal of Systems and Software*, 84(2):198 – 204, 2011.

[BMvT78] E. Berlekamp, R. McEliece, and H. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384–386, May 1978.

[BN90] J. Bruck and M. Naor. The hardness of decoding linear codes with preprocessing. *IEEE Transactions on Information Theory*, 36(2):381–385, 1990.

[BR95] M. Bellare and P. Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer Berlin / Heidelberg, 1995.

[Buc65] Buchberger, B. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, Innsbruck, 1965.

[CC94] A. Canteaut and H. Chabanne. A further improvement of the work factor in an attempt at breaking McEliece's cryptosystem. In *EUROCODE 94*, pages 169–173. INRIA, 1994.

[CC95] A. Canteaut and F. Chabaud. Improvements of the attacks on cryptosystems based on error-correcting codes. Technical Report 95–21, INRIA, 1995.

[CC98] A. Canteaut and F. Chabaud. A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, 1998.

[CFS01] N. T. Courtois, M. Finiasz, and N. Sendrier. How to achieve a McEliece-based digital signature scheme. *Lecture Notes in Computer Science*, 2248:157–174, 2001.

[CLO01] D. A. Cox, J. B. Little, and D. O'Shea. *Ideals, Varieties, and algorithms: an Introduction to Computational Algebraic Geometry and Commutative Algebra.* Undergraduate Texts in Mathematics, Springer-Verlag, New York., 2001.

[Com74] L. Comtet. *Advanced Combinatorics*. Reidel, Dordrecht, 1974.

[COV07] P.L. Cayrel, A. Otmani, and D. Vergnaud. On Kabatianskii-Krouk-Smeets Signatures. In *Proceedings of the first International Workshop on the Arithmetic of Finite Fields (WAIFI 2007)*, Springer Verlag Lecture Notes, pages 237–251, Madrid, Spain, June 21–22 2007.

[CS98] A. Canteaut and N. Sendrier. Cryptanalysis of the original McEliece cryptosystem. In *Advances in Cryptology - ASIACRYPT 98*, number 1514 in Lecture Notes in Computer Sscience, pages 187–199. Springer-Verlag, 1998.

[Dal07] L. Dallot. Towards a concrete security proof of Courtois, Finiasz and Sendrier signature scheme. In *WEWoRC*, pages 65–77, 2007.

[Dal10] L. Dallot. *Analyse de protocoles cryptographiques fondés sur les codes correcteurs d'erreurs*. PhD thesis, University of Caen, 2010.

[dC97] D. de Caen. A lower bound on the probability of a union. *Discrete Mathematics*, 169:217–220, 1997.

[DMQN09] R. Dowsley, J. Müller-Quade, and A. C. A. Nascimento. A CCA2 secure public key encryption scheme based on the McEliece assumptions in the standard model. In *CT-RSA*, pages 240–251, 2009.

[Dum91] I. Dumer. On minimum distance decoding of linear codes. In *Proc. 5th Joint Soviet-Swedish Int. Workshop Inform. Theory*, pages 50–52, Moscow, 1991.

[Dum96] I. Dumer. Suboptimal decoding of linear codes : partition techniques. *IEEE Transactions on Information Theory*, 42(6):1971–1986, 1996.

[Dür87] Arne Dür. The automorphism groups of Reed-Solomon codes. *Journal of Combinatorial Theory, Series A*, 44:69–82, 1987.

[Fau99] J.-C. Faugère. A new efficient algorithm for computing gröbner bases (f4). *Journal of Pure and Applied Algebra*, 139(1-3):61–88, 1999.

[Fau02] J.-C. Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero : F5. In *ISSAC 02*, pages 75–83. ACM press, 2002.

[FGLM93] J.C. Faugère, P. M. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional gröbner bases by change of ordering. *J. Symb. Comput.*, 16(4):329–344, 1993.

[FGUO+11] J.C. Faugère, V. Gauthier-Umana, A. Otmani, L. Perret, and J.P. Tillich. Distinguisher for high rate McEliece cryptosystems. In *Proceedings of the 2011 IEEE Information Theory Workshop (ITW 2011)*, Paraty, Brazil, October 16-20 2011.

[FM08] C. Faure and L. Minder. Cryptanalysis of the McEliece cryptosystem over hyperelliptic curves. In *Proceedings of the eleventh International Workshop on Algebraic and Combinatorial Coding Theory*, pages 99–107, Pamporovo, Bulgaria, June 2008.

[FOPT10a] J.C. Faugère, A. Otmani, L. Perret, and J.P. Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 279–298, May 30 –June 3 2010.

[FOPT10b] J.C. Faugère, A. Otmani, L. Perret, and J.P. Tillich. Algebraic cryptanalysis of McEliece variants with compact keys - towards a complexity analysis. In *Yet Another Conference on Cryptography (YACC 2010)*, Porquerolles Island, France, October 4-8 2010.

[FS09] M. Finiasz and N. Sendrier. Security bounds for the design of code-based cryptosystems. In M. Matsui, editor, *Asiacrypt 2009*, volume 5912 of *LNCS*, pages 88–105. Springer, 2009.

[FSS11] J.-C. Faugère, M. Safey El Din, and P.-J. Spaenlehauer. Grbner Bases of Bihomogeneous Ideals Generated by Polynomials of Bidegree (1,1): Algorithms and Complexity. *Journal of Symbolic Computation*, 46(4):406–437, 2011.

[Gab05] P. Gaborit. Shorter keys for code based cryptography. In *Proceedings of the 2005 International Workshop on Coding and Cryptography (WCC 2005)*, pages 81–91, Bergen, Norway, March 2005.

[Gib91] J. Gibson. Equivalent goppa codes and trapdoors to McEliece's public key cryptosystem. In Donald Davies, editor, *Advances in Cryptology – EUROCRYPT 91*, volume 547 of *Lecture Notes in Computer Science*, pages 517–521. Springer Berlin / Heidelberg, 1991.

[Gib95] J. K. Gibson. Severely denting the Gabidulin version of the McEliece public key cryptosystem. *Design Codes and Cryptography*, 6(1):37–45, 1995.

[Gop70] V. D. Goppa. A new class of linear correcting codes. *Probl. Peredachi Inf.*, 6(3), 1970.

[GPT91] E. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. Ideals over a non-commutative ring and their applications to cryptography. In *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques*, number 547 in LNCS, pages 482–489, Brighton, april 1991.

[HGS99] C. Hall, I. Goldberg, and B. Schneier. Reaction attacks against several public-key cryptosystem. In Vijay Varadharajan and Yi Mu, editors, *Information and Communication Security*, volume 1726 of *Lecture Notes in Computer Science*, pages 2–12. Springer Berlin / Heidelberg, 1999.

[JM96] H. Janwa and O. Moreno. McEliece public key cryptosystems using algebraic-geometric codes. *Designs Codes and Cryptography*, 8(3):293–307, 1996.

[KI01] K. Kobara and H. Imai. Semantically secure McEliece public-key cryptosystems-conversions for McEliece PKC. In Kwangjo Kim, editor, *Public Key Cryptography, 4th International Workshop on Practice and Theory in Public Key Cryptography, PKC 2001, Cheju Island, Korea, February 13-15, 2001, Proceedings*, volume 1992 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2001.

[KKS97] G. Kabatianskii, E. Krouk, and B. J. M. Smeets. A digital signature scheme based on random error-correcting codes. In *IMA Int. Conf.*, volume 1355 of *Lecture Notes in Computer Science*, pages 161–167. Springer, 1997.

[KKS05] G. Kabatiansky, E. Krouk, and S. Semenov. *Error Correcting Coding and Security for Data Networks: Analysis of the Superchannel Concept*. John Wiley & Sons, 2005.

[LB88] P. J. Lee and E. F. Brickell. An observation on the security of McEliece's public-key cryptosystem. In *Advances in Cryptology - EUROCRYPT 88*, volume 330/1988 of *Lecture Notes in Computer Science*, pages 275–280. Springer, 1988.

[Leo88] J. S. Leon. A probabilistic algorithm for computing minimum weights of large error-correcting codes. *IEEE Transactions on Information Theory*, 34(5):1354–1359, 1988.

[Lev86] L. A. Levin. Average case complete problems. *SIAM J. Comput.*, 15(1):285–286, 1986.

[LS01] P. Loidreau and N. Sendrier. Weak keys in the mceliece public-key cryptosystem. *IEEE Transactions on Information Theory*, 47(3):1207–1211, 2001.

[MB09] R. Misoczki and P. S. L. M. Barreto. Compact McEliece keys from Goppa codes. In *Selected Areas in Cryptography (SAC 2009)*, Calgary, Canada, August 13-14 2009.

[McE78] R. J. McEliece. *A Public-Key System Based on Algebraic Coding Theory*, pages 114–116. Jet Propulsion Lab, 1978. DSN Progress Report 44.

[McL84] A. M. McLoughlin. The complexity of computing the covering radius of a code. *IEEE Transactions on Information Theory*, 30(6):800–804, 1984.

[MCMMP11] I. Marquez-Corbella, E. Martinez-Moro, and R. Pellikaan. Evaluation of public-key cryptosystems based on algebraic geometry codes. In J. Borges and M. Villanueva, editors, *Proceedings of the Third International Castle Meeting on Coding Theory and Applications*, pages 199–204, Barcelona, Spain, September 11-15 2011.

[MMT11] A. May, A. Meurer, and E. Thomae. Decoding random linear codes in $o(2^{0.054n})$, 2011. In Advances in Cryptology (Asiacrypt 2011).

[MRS00] C. Monico, J. Rosenthal, and A. Shokrollahi. Using low density parity check codes in the McEliece cryptosystem. In *IEEE International Symposium on Information Theory (ISIT 2000)*, page 215, Sorrento, Italy, 2000.

[MS86] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North–Holland, Amsterdam, fifth edition, 1986.

[MS07] L. Minder and A. Shokrollahi. Cryptanalysis of the Sidelnikov cryptosystem. In *Eurocrypt 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 347–360, Barcelona, Spain, 2007.

[Nie86] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems Control Inform. Theory*, 15(2):159–166, 1986.

[NIKM08] R. Nojima, H. Imai, K. Kobara, and K. Morozov. Semantic security for the McEliece cryptosystem without random oracles. *Des. Codes Cryptography*, 49(1-3):289–305, 2008.

[OT11] A. Otmani and J.P. Tillich. An efficient attack on all concrete KKS proposals. Cryptology ePrint Archive, Report 2011/356, 2011. `http://eprint.iacr.org/`.

[OTD08] A. Otmani, J.P. Tillich, and L. Dallot. Cryptanalysis of McEliece cryptosystem based on quasi-cyclic ldpc codes. In *Proceedings of First International Conference on Symbolic Computation and Cryptography*, pages 69–81, 2008.

[OTD10] A. Otmani, J.P. Tillich, and L. Dallot. Cryptanalysis of two mceliece cryptosystems based on quasi-cyclic codes. *Mathematics in Computer Science*, 3:129–140, 2010.

[Ove08] R. Overbeck. Structural attacks for public key cryptosystems based on Gabidulin codes. *J. Cryptology*, 21(2):280–301, 2008.

[Pat75] N. Patterson. The algebraic decoding of Goppa codes. *IEEE Transactions on Information Theory*, 21(2):203–207, 1975.

[Per11] E. Persichetti. Compact mceliece keys based on quasi-dyadic srivastava codes. Cryptology ePrint Archive, Report 2011/179, 2011.

[PR97] E. Petrank and R. M. Roth. Is code equivalence easy to decide? *IEEE Transactions on Information Theory*, 43(5):1602–1604, 1997.

[Sen00] N. Sendrier. Finding the permutation between equivalent linear codes: The support splitting algorithm. *IEEE Transactions on Information Theory*, 46(4):1193–1203, 2000.

[Sen11] N. Sendrier. Decoding one out of many, 2011. preprint.

[Sha48] C. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, pages 379–423, 623–656, October 1948.

[Sid94] V.M. Sidelnikov. A public-key cryptosytem based on Reed-Muller codes. *Discrete Mathematics and Applications*, 4(3):191–207, 1994.

[SS92] V.M. Sidelnikov and S.O. Shestakov. On the insecurity of cryptosystems based on generalized Reed-Solomon codes. *Discrete Mathematics and Applications*, 1(4):439–444, 1992.

[Ste88] J. Stern. A method for finding codewords of small weight. In G. D. Cohen and J. Wolfmann, editors, *Coding Theory and Applications*, volume 388 of *Lecture Notes in Computer Science*, pages 106–113. Springer, 1988.

[UL09] V. Gauthier Umana and G. Leander. Practical key recovery attacks on two McEliece variants. `http://eprint.iacr.org/2009/509`, 2009.

[Var97] A. Vardy. Algorithmic complexity in coding theory and the minimum distance problem. In *STOC*, pages 92–109, 1997.

[VDv02] E. Verheul, J.M. Doumen, and H.C.A. van Tilborg. Sloppy alice attacks! adaptive chosen ciphertext attacks on the mceliece cryptosystem. In M. Blaum, P.G. Farrell, and H.C.A. van Tilborg, editors, *Information, Coding and Mathematics*, pages 99–119. Kluwer Academic Publishers, Boston, Massachusetts, May 2002.

[vT90] J. van Tilburg. On the McEliece public-key cryptosystem. In *CRYPTO 88: Proceedings of the 8th Annual International Cryptology Conference on Advances in Cryptology*, pages 119–131, London, UK, 1990. Springer-Verlag.

[Wie10] C. Wieschebrink. Cryptanalysis of the niederreiter public key scheme based on grs subcodes. In N. Sendrier, editor, *Post-Quantum Cryptography, Third International Workshop, PQCrypto 2010*, volume 6061 of *Lecture Notes in Computer Science*, pages 61–72, Darmstadt, Germany, May 2010. Springer.

Abstract

A large part in the design of secure cryptographic primitives consists in identifying hard algorithmic problems. Despite the fact that several problems have been proposed as a foundation for public-key primitives, those effectively used are essentially classical problems coming from integer factorization and discrete logarithm. On the other hand, coding theory appeared with the goal to solve the challenging problem of decoding a random linear code. It is widely admitted as a hard problem that has led McEliece in 1978 to propose the first code-based public-key encryption scheme. The key concept is to focus on codes that come up with an efficient decoding algorithm. He also advocated the use of binary Goppa codes. Since then, it belongs to the very few cryptosystems which remain unbroken.

This thesis is primarily interested in studying the security of code-based primitives. The first category we analyzed consists of variants of the McEliece cryptosystem. Our works expose practical key-recovery attacks either by mounting dedicated techniques, or by devising algebraic attacks. This latter result also provides a new framework to assess the security of the McEliece cryptosystem and a first step towards the design of attacks based on the solving of algebraic systems. Furthermore, we show that this approach can be used to study the Goppa Code Distinguishing problem, which asks whether there is an efficient way to distinguish a Goppa code from a randomly drawn linear code. It represents an important assumption which supports the use of Goppa codes in cryptography. We show that it is possible to efficiently solve it as long as the code rate is sufficiently high. Finally, we investigate the security of a signature scheme based on two random linear codes. Our analysis shows that the attack is sensitive to their rates and can be practical when the rates are close.

Keywords. Code-based cryptography, key-recovery attacks, algebraic cryptanalysis, Goppa code distinguishing problem.