Cryptanalysis of the McEliece Public Key Cryptosystem based on Polar Codes

Magali Bardet, Julia Chaulet, Vlad Dragoi, Ayoub Otmani, Jean-Pierre Tillich

Cryptanalysis of the McEliece Public Key Cryptosystem Based on Polar Codes

Magali Bardet[1], Julia Chaulet[2], Vlad Dragoi[1], Ayoub Otmani[1], and Jean-Pierre Tillich[2]

[1] Normandie Univ, France; UR, LITIS, F-76821 Mont-Saint-Aignan, France.
{magali.bardet,vlad.dragoi1,ayoub.otmani}@univ-rouen.fr
[2] Inria, SECRET Project, 78153 Le Chesnay Cedex, France.
{julia.chaulet,jean-pierre.tillich}@inria.fr

Abstract. Polar codes discovered by Arikan form a very powerful family of codes attaining many information theoretic limits in the fields of error correction and source coding. They have in particular much better decoding capabilities than Goppa codes which places them as a serious alternative in the design of both a public-key encryption scheme *à la* McEliece and a very efficient signature scheme. Shrestha and Kim proposed in 2014 to use them in order to come up with a new code-based public key cryptosystem. We present a key-recovery attack that makes it possible to recover a description of the permuted polar code providing all the information required for decrypting any message.

1 Introduction

The concept of *post-quantum* cryptography appeared after Peter Shor showed in [Sho97]that all cryptosystems which base their security on the hardness of the factoring problem or the discrete logarithm problem can be attacked in polynomial time with a quantum computer (see [BBD09] for an extensive report). This threatens most if not all public-key cryptosystems deployed in practice, such as RSA [RSA78] or DSA [Kra91]. Cryptography based on the difficulty of decoding a linear code, on the other hand, is believed to resist quantum attacks and is therefore considered as a viable replacement for those schemes in future applications. Yet, independently of their so-called "post-quantum" nature, code-based cryptosystems offer other benefits even for present-day applications due to their excellent algorithmic efficiency.

The first code-based cryptosystem is the McEliece cryptosystem [McE78], originally proposed using binary Goppa codes. Afterwards, several code families have been suggested to replace them: generalized Reed–Solomon codes (GRS) [Nie86] or subcodes of them [BL05], Reed–Muller codes [Sid94], algebraic geometry codes [JM96], LDPC codes [BC07,BBC08], a certain kind of non binary Goppa codes (called wild Goppa codes or wild Goppa codes incognito) [BLP10,BLP11], MDPC codes [MTSB12], convolutional codes [LJ12], and more recently polar codes [SK14] or subcodes of them [HSEA14]. Some of these schemes allow to reduce the public key size compared to the original McEliece

cryptosystem while presumably keeping the same level of security against generic decoding algorithms.

However, for many of the aforementioned schemes it has been shown that a description of the underlying code suitable for decoding can be obtained which breaks the corresponding scheme. This has been achieved for GRS codes in [SS92], subcodes of GRS codes in [Wie10], Reed-Muller codes in [MS07]. Algebraic geometry codes based on (very) low genus hyperelliptic curves were broken in [FM08], whereas the general case was broken in [CMCP14]. A first version of the scheme based on LDPC codes proposed in [BC07] has been successfully attacked in [OTD08] (but the new scheme proposed in [BBC08] seems to be immune to this kind of attack). Some of the parameters that can be found in [BLP10,BLP11] have been successfully cryptanalyzed with a polynomial time attack in [COT14] or with an exponential time attack in [FPdP14], and finally the convolutional scheme of [LJ12] was successfully cryptanalyzed in [LT13].

All of these attacks (with the exception of [LT13]) pinpoint algebraic properties of the codes which raises the issue of looking for alternative code families with little or no algebraic structure. In this respect the proposals of [SK14,HSEA14] might be very attractive. Moreover, polar codes enjoy another feature that only few other codes have: they enjoy a decoding algorithm that can also be used to produce for any binary word a codeword that is essentially as close as possible as announced by the information theoretic upper bounds [CT91, Th.13.2.1,Th.13.3.1] (see [KU10] for a proof of this result). This would make such codes perfect candidates in a signature scheme [OT12] based on the Niederreiter scheme [Nie86]. In particular, this would give a much more efficient signature than the CFS scheme [CFS01].

A McEliece scheme based on (binary) polar codes also raises some other interesting issues. There is basically no large choice for such codes and there is essentially only one polar code (up to permutation of the coordinates) of a given rate. Generally it is advocated that one should take a large code family in the McEliece cryptosystem, because if there is only one code up to permutation of the coordinates, then attacking the scheme amounts to solve the code equivalence problem [PR97]. For most codes, this is generally easy to do by using the support splitting algorithm [Sen00]. However, this algorithm requires in a crucial way that the code has a small hull (which is the intersection of the code with its dual) and a small permutation group, both of them being precisely the opposite for polar codes. Interestingly enough, polar codes are known to be related to the Reed-Muller code family [Arı09]. These two code families behave exactly in the same way with respect to these properties: they have very large hull and permutation group and there is also only one (or zero) Reed-Muller code for a given rate. Note that when a McEliece scheme based on Reed-Muller codes was proposed in [Sid94], its security relied precisely on the assumption that it could be possible in theory to use a single code (up to permutation of the coordinates) by using codes with a large hull and a large permutation group that would defeat attacks based on the support splitting algorithm. It took thirteen years [MS07]

to break this McEliece scheme and the attack used many algebraic properties of the Reed-Muller codes, that are presumably absent for polar codes.

However, we will show that despite the fact that polar codes seem to be immune against a plain use of the support splitting algorithm, it can nevertheless be cryptanalyzed successfully. We will show here how to recover from the public generator matrix a description of the code that is suitable for decoding. Our attack uses several ingredients:

(i) polar codes have rather low weight codewords which can be found by standard low weight codeword searching algorithms [Ste88,Dum91];
(ii) shortening the code with respect to these low weight codewords and taking the dual also gives a code with low weight codewords which can be recovered with the aforementioned algorithms;
(iii) by characterizing the permutation group of polar codes together with the low-weight codewords found in Step (ii), it is possible to find among the codewords found in Step (i) a subset of codewords which up to equivalence by the permutation group can be considered as codewords whose support are very specific affine spaces;
(iv) Puncturing the code with respect to the support of an element of minimum weight in this last subset of codewords gives a code of small length (typically 16 or 32) whose structure is known up to code equivalence. The code equivalence problem is then solved in this case and is used to recover step by step the underlying polar codes.

Steps (i) and (ii) are directly inspired from the Minder-Shokrollahi attack [MS07] on the McEliece cryptosystem based on Reed-Muller codes, however Steps (iii) and (iv) are new and very specific to polar codes. Basically, the fact that the whole affine group is the permutation group of Reed-Muller code simplifies a great deal the attack of [MS07]. This is not the case anymore for polar codes and the crux for being able to mount this attack is to understand which subgroup of the affine group is part of the permutation group of a polar code and then to use this structure in a relevant way. Amazingly enough, it turns out that a rather large subgroup is the answer to this problem and that polar codes are much more symmetric than could be guessed from their definition. This result is of independent interest and might be used to improve the decoding algorithms of polar codes.

In a general way, in order to understand the structure of polar codes for breaking this cryptosystem we have introduced here new concepts. In particular we suggest here a new code construction, that we call decreasing monomial codes which contains both the Reed-Muller code family and the polar code family and which has a large subgroup of the affine group as permutation group. This construction explains why polar codes have such a large permutation group, but again this new code construction could be of independent interest in coding theory. We also introduce in Step (iv) a novel iterative way of solving the code equivalence problem that could be interesting for solving the code equivalence problem for codes obtained from the $(u|u+v)$ construction.

2 Basic Facts

In this section we recall a few facts about the McEliece cryptosystem, polar codes, the code equivalence problem, and code operations like shortening or puncturing.

Polar Codes. Polar codes were discovered by Arikan [Arı09] and form a very powerful family of codes that gave a nice constructive way of attaining many information theoretic limits in error correction and source coding. In particular, they allow to attain the capacity of any symmetric memoryless channel with a low complexity decoding algorithm (namely the successive cancellation decoder of Arikan). Since they have much better decoding capabilities than Goppa codes, it is reasonable to study whether they can be used in a McEliece scheme. Due to their better correction capacity, this allows for instance to decrease the key sizes of the scheme. Decoding such codes is also faster than decoding Goppa codes and this can also be used to speed up the decryption process.

They can be described as codes of length $n = 2^m$, where m is an arbitrary integer. They may take any dimension between 0 and 2^m. The polar code of length $n = 2^m$ and dimension k is obtained through a generator matrix which picks a specific subset of k rows of the $2^m \times 2^m$ matrix:

$$\boldsymbol{G}_m \stackrel{\text{def}}{=} \underbrace{\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}}_{m \text{ times}}.$$

Note that we depart here slightly from the usual convention for polar codes which is to use in the Kronecker product the matrix $\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$. The two definitions (ours and the standard one) are easily seen to be equivalent, they just amount to order the code positions differently. Our convention presents the advantage of simplifying the polynomial formalism that follows.

The specific choice of rows that are picked depends (a little bit) on the noisy channel for which the code is devised. For a given noise model, there is a way to compute the k rows which are used to define the generator matrix of the code. Roughly speaking these rows are chosen in such a way that it gives good performances for the successive cancellation decoder.

McEliece cryptosystem. The (binary) McEliece public-key scheme [McE78] can be described as follows. The key generation algorithm picks a random $k \times n$ generator matrix $\boldsymbol{G}$ of a binary linear code $\mathscr{C}$ which is itself randomly picked in a family of codes for which t errors can be efficiently corrected. The *secret* key is the decoding algorithm $\mathcal{D}$ associated to $\mathscr{C}$ and the *public* key is $\boldsymbol{G}$. To encrypt $\boldsymbol{u} \in \mathbb{F}_2^k$, the sender chooses a random vector $\boldsymbol{e}$ in $\mathbb{F}_2^n$ of Hamming weight less than or equal to t and computes the ciphertext $\boldsymbol{c} = \boldsymbol{u}\boldsymbol{G} + \boldsymbol{e}$. The receiver then recovers the plaintext by applying $\mathcal{D}$ on $\boldsymbol{c}$.

Code equivalence problem. In the McEliece scheme based on polar codes [SK14], since there is in essence a single (binary) polar code of a given dimension and length, breaking the scheme amounts to find for a permuted version of the

polar code a permutation that gives the original polar code. In other words, we face here as for the McEliece scheme based on Reed-Muller codes the code equivalence problem. To give a formal definition of this problem we will use the following notation and definition.

Notation 1 (Permutation of a word and a code). *The symmetric group of degree n is denoted by S_n. Let $\boldsymbol{x} = (x_i)_{0 \leqslant i < n} \in \mathbb{F}_2^n$ and π be a permutation of $\{0, 1, \ldots, n-1\}$. We denote by $\boldsymbol{x}^\pi = (x_{\pi(i)})_{0 \leqslant i < n}$ the vector $\boldsymbol{x}$ permuted by π and for a binary code $\mathscr{C}$ of length n, its permutation by π is defined by*

$$\mathscr{C}^\pi \stackrel{def}{=} \left\{ \boldsymbol{c}^\pi \mid \boldsymbol{c} \in \mathscr{C} \right\}.$$

Definition 1 (Permutation group of a code). *The permutation group of a code $\mathscr{C}$ is the set of permutations π such that $\mathscr{C}^\pi = \mathscr{C}$.*

The code equivalence problem can be stated as follows:

Problem 1 (Code equivalence search problem). Given $\mathscr{C}$ and $\mathscr{C}^\pi$ where $\mathscr{C}$ is a code of length n and π belongs to S_n, find $\hat{\pi}$ in S_n such that $\mathscr{C}^{\hat{\pi}} = \mathscr{C}^\pi$.

Note that we do not necessarily have $\hat{\pi} = \pi$ when the permutation group of the code is non trivial. It is namely immediate to prove that:

Proposition 1. *For any $\boldsymbol{x} = (x_i)_{0 \leqslant i < n}$ in $\mathbb{F}_2^n$ and all permutations π and π' in S_n we have*

$$(\boldsymbol{x}^\pi)^{\pi'} = \boldsymbol{x}^{\pi\pi'}.$$

Let $\mathscr{C}$ be a code of length n with permutation group $\mathcal{G}$ and π be a permutation of the same length as $\mathscr{C}$ (i.e. a permutation in S_n). We have

$$\left\{ \hat{\pi} \in S_n \mid \mathscr{C}^{\hat{\pi}} = \mathscr{C}^\pi \right\} = \mathcal{G}\pi$$

If $\mathscr{C}$ has permutation group $\mathcal{G}$, then $\mathscr{C}^\pi$ has permutation group $\pi^{-1}\mathcal{G}\pi$.

Proof. The first part of the proposition can be proved by bringing in $\boldsymbol{x}' \stackrel{\text{def}}{=} \boldsymbol{x}^\pi$ and observing that:

(i) for any i in $\{0, \ldots, n-1\}$ we have $x_i = x'_{\pi^{-1}(i)}$ since $x'_j = x_{\pi(j)}$ for $j = \pi^{-1}(i)$.
(ii) $(\boldsymbol{x}^\pi)^{\pi'} = \boldsymbol{x}'^{\pi'} = (x'_{\pi'(i)})_{0 \leqslant i < n} = (x'_{\pi^{-1}(\pi(\pi'(i)))})_{0 \leqslant i < n} = (x_{\pi(\pi'(i))})_{0 \leqslant i < n} = \boldsymbol{x}^{\pi\pi'}$.

From this we deduce that for any σ in $\mathcal{G}$, we have $\mathscr{C}^{\sigma\pi} = (\mathscr{C}^\sigma)^\pi = \mathscr{C}^\pi$. Conversely if $\mathscr{C}^{\hat{\pi}} = \mathscr{C}^\pi$, then $(\mathscr{C}^{\hat{\pi}})^{\pi^{-1}} = \mathscr{C}$ and therefore $\hat{\pi}\pi^{-1}$ is in $\mathcal{G}$, meaning that $\hat{\pi}$ is in $\mathcal{G}\pi$.

To prove the last part we observe that if γ is a permutation that leaves $\mathscr{C}$ invariant, then $\pi^{-1}\gamma\pi$ is a permutation that leaves $\mathscr{C}^\pi$ invariant, since $(\mathscr{C}^\pi)^{\pi^{-1}\gamma\pi} = \mathscr{C}^{\gamma\pi} = (\mathscr{C}^\gamma)^\pi = \mathscr{C}^\pi$. Conversely if γ' is a permutation of $\mathscr{C}^\pi$, then the same kind of computation shows that $\gamma \stackrel{\text{def}}{=} \pi\gamma'\pi^{-1}$ is a permutation of $\mathscr{C}$.

What makes the equivalence problem difficult for polar codes is that the standard algorithm for solving it, namely the support splitting algorithm of [Sen00] is too complex to be used in this context due to the very large size of the hull of the polar code. What makes the problem even more intricate is the fact that a polar code turns out to have a very large permutation group which complicates the task significantly.

Operations on codes. One of the basic operations used in the support splitting algorithm for solving the code equivalence problem is to consider shortened and punctured codes. For a given code $\mathscr{C}$ and a subset $\mathcal{J} \subseteq \{0, \ldots, n-1\}$ the *punctured* code $\mathcal{P}_{\mathcal{J}}(\mathscr{C})$ and *shortened* code $\mathcal{S}_{\mathcal{J}}(\mathscr{C})$ are defined as:

$$\mathcal{P}_{\mathcal{J}}(\mathscr{C}) \stackrel{\text{def}}{=} \Big\{ (c_i)_{i \notin \mathcal{J}} \mid \boldsymbol{c} \in \mathscr{C} \Big\};$$
$$\mathcal{S}_{\mathcal{J}}(\mathscr{C}) \stackrel{\text{def}}{=} \Big\{ (c_i)_{i \notin \mathcal{J}} \mid \exists \boldsymbol{c} = (c_i)_i \in \mathscr{C} \text{ such that } \forall i \in \mathcal{J},\ c_i = 0 \Big\}.$$

Instead of writing $\mathcal{P}_{\{j\}}(\mathscr{C})$ and $\mathcal{S}_{\{j\}}(\mathscr{C})$ when $\mathcal{J} = \{j\}$ we rather use the notation $\mathcal{P}_j(\mathscr{C})$ and $\mathcal{S}_j(\mathscr{C})$. These codes are used in the following way to solve the code equivalence problem: $\mathscr{C}$ is punctured in a position i whereas $\mathscr{C}^\pi$ is punctured in some position j. If we have a quick way to check that two codes are not equivalent, then we can use this tool to check whether the two punctured codes may be equivalent or not (in the support splitting algorithm this is done by computing the weight enumerator of the hull which is obviously invariant by permutation). If the two punctured codes are not equivalent, then we know for sure that i and j can not correspond to each other via the permutation of position π. The same idea works also for the shortened code.

3 Decreasing Monomial Codes

The purpose of this section is to introduce a novel algebraic framework that sheds some light about the structure of polar codes. We will in particular give a new class of codes, that we call *decreasing monomial codes* that contains as a particular case, polar codes and Reed-Muller codes. The dual of a decreasing monomial code is a decreasing monomial code and under a very mild condition, such codes turn out to be weakly self-dual (i.e. the hull of the code is the code itself). We will then prove that this general construction has a very large permutation group and both facts put together will explain why polar codes have such a large permutation group and hull. We will use here the polynomial formalism that is generally used to describe Reed-Muller codes. It turns out that this polynomial formalism is also very handy for describing polar codes.

Reed-Muller codes. It is well known that Reed-Muller codes of length 2^m can be obtained as evaluation codes of polynomials in $\mathbb{F}_2[x_0, \ldots, x_{m-1}]$. Polar codes can also be described through this formalism. Since we are interested in evaluations of such polynomials over entries in $\mathbb{F}_2^m$ we will identify x_i with x_i^2 and work in the ring $\mathbb{R}_2[x_0, \ldots, x_{m-1}] = \mathbb{F}_2[x_0, \ldots, x_{m-1}]/(x_0^2 - x_0, \ldots, x_{m-1}^2 - x_{m-1})$. It will be convenient with this formalism to associate to a polynomial $g \in$

$\mathbb{R}_2[x_0, \ldots, x_{m-1}]$ the binary vector denoted by $\mathsf{ev}(g)$ in $\mathbb{F}_2^n$ with $n = 2^m$ which is the evaluation of the polynomial in all the binary entries $(u_0, \ldots, u_{m-1}) \in \mathbb{F}_2^m$. In other words

$$\mathsf{ev}(g) = \big(g(u_0, \ldots, u_{m-1})\big)_{(u_0,\ldots,u_{m-1})\in\mathbb{F}_2^m}$$

With this notation, we will view the indices of a vector as elements of $\mathbb{F}_2^m$. This notation does not specify the order we use for the elements of $\mathbb{F}_2^m$. We actually use the natural order by viewing $(u_0, \ldots, u_m)$ as the integer $\sum_{i=0}^{m-1} u_i 2^i$. With this notation at hand, the Reed-Muller code $\mathscr{R}(r, m)$ is defined as

$$\mathscr{R}(r, m) \stackrel{\text{def}}{=} \big\{\mathsf{ev}(P) \mid \deg P \leqslant r\big\}$$

Obviously this code is generated by the codewords $\mathsf{ev}(g)$ where g is a monomial of degree less than or equal to r. Recall that a *monomial* is any product of variables of the form $x_0^{g_0} \cdots x_{m-1}^{g_{m-1}}$ where $g_0, \ldots, g_{m-1}$ are binary. The set of all monomials is denoted by:

$$\mathcal{M} \stackrel{\text{def}}{=} \{1, x_0, \ldots, x_{m-1}, x_0x_1, \ldots, x_0 \cdots x_{m-1}\}.$$

Reed-Muller codes have a very large permutation group which is isomorphic to the affine group over $\mathbb{F}_2^m$. Indeed, it can be checked immediately that:

(i) any bijective affine transformation A over $\mathbb{F}_2^m$ can be viewed as a permutation of the code positions by mapping $(u_0, \ldots, u_{m-1})$ to $A(u_0, \ldots, u_{m-1})$;

(ii) this permutation leaves the code invariant since $P(A(x_0, \ldots, x_{m-1}))$ is a polynomial of degree at most the degree of P and therefore if $\mathsf{ev}(P) \in \mathscr{R}(r, m)$ then $\mathsf{ev}(P \circ A) \in \mathscr{R}(r, m)$.

Monomial codes. It is straightforward to check that the rows of $\boldsymbol{G}$ are all possible evaluations of monomials. This fact is easily proved by induction on m by observing that $(1, 1)$ is the evaluation of the constant monomial 1 and that $(0, 1)$ is the evaluation of the monomial x_0. From this, we easily see that a polar code is a *monomial code*, meaning codes generated by evaluations of monomials (see the formal definition below).

It will also be very convenient to introduce the following partial order $\preceq$ on monomials

Definition 2 (Monomial order). *The monomials of the same degree are ordered as*

$$x_{i_1} \ldots x_{i_s} \preceq x_{j_1} \ldots x_{j_s} \text{ if and only if for any } \ell \in \{1, \ldots, s\},\ \ i_\ell \leqslant j_\ell$$

where we assume that $i_1 < \cdots < i_s$ and $j_1 < \cdots < j_s$.

This order is extended to other monomials through divisibility, namely: $f \preceq g$ if and only if there is a divisor g^* of g such that $f \preceq g^*$.

Obviously for any monomial f of $\mathcal{M}$ the constant polynomial 1 satisfies the inequality $1 \preceq f$. The *interval* $[f; h]$ where f and h are in $\mathcal{M}$ with $f \preceq h$ is the set of monomials $g \in \mathcal{M}$ such that $f \preceq g \preceq h$. We will also need the following definition

Definition 3 (Decreasing set). *A set $I \subseteq \mathcal{M}$ is* decreasing *if and only if ($f \in I$ and $g \preceq f$) implies $g \in I$.*

With these definitions, we define monomial and decreasing monomial codes as follows.

Definition 4 (Monomial and decreasing monomial codes). *Let I be a finite set of multivariate polynomials in m variables and set $n \stackrel{def}{=} 2^m$. The linear code defined by I is the vector subspace $\mathscr{C}(I) \subseteq \mathbb{F}_2^n$ generated by $\{\mathsf{ev}(f) \mid f \in I\}$. It is called the* polynomial code *associated to I.*

1. *When $I \subseteq \mathcal{M}$, $\mathscr{C}(I)$ is called a* monomial code.
2. *When $I \subseteq \mathcal{M}$ is a decreasing set, $\mathscr{C}(I)$ is called a* decreasing monomial code.

The dimension of monomial codes is easily derived.

Lemma 1. *For all $I \subseteq \mathcal{M}$ the dimension of the monomial code $\mathscr{C}(I)$ is equal to $|I|$.*

Proof. This comes from the linear independence of the monomials in $\mathbb{R}_2[x_0, \ldots, x_{m-1}]$.

Example 1. The r-th order Reed-Muller code is the decreasing monomial code defined by the interval $[1; x_{m-r} \ldots x_{m-1}]$since:

$$\mathscr{R}(r, m) = \mathscr{C}\left([1; x_{m-r} \ldots x_{m-1}]\right).$$

The dimension $1 + m + \cdots + \binom{m}{r}$ comes directly from Lemma 1.

It turns out that it can be proved, but this is beyond the scope of this article, that polar codes devised for the erasure channel are also decreasing monomial codes. The point is that if we take a row of $\boldsymbol{G}_m$ to be a row of the generator matrix (and view this row as a monomial - since as we have explained before - all these rows correspond to an evaluation of a particular monomial) all the rows that are "smaller" (in the sense of the monomial order defined before) will also be chosen to be part of the generator matrix of the polar code. This fact can be proved by studying the polarization process ([Arı09]) which is at the heart of choosing the relevant rows of $\boldsymbol{G}_m$. Simple heuristics can be invoked that this also holds for other channel models and we have experimental evidence showing that this seems to hold in particular for polar codes devised for the binary symmetric channel (which are the polar codes used here). This fact can be simply checked for the polar codes that we have attacked here.

Duality and permutation group of decreasing monomial codes. Duals of decreasing monomial codes have a very simple description and it will turn out that under certain very weak conditions, they are weakly self-dual. It is

readily seen that the dual of a monomial code is a polynomial code, but it is not necessarily a monomial code. However the dual of a decreasing monomial code turns out to be a decreasing monomial code. To describe this dual we will use the following notion of *(multiplicative) complement* of a monomial g and denote it by $\check{g}$.

Definition 5 (Complement). *For any $g \in \mathcal{M}$ we define the complement of g as*

$$\check{g} = \frac{x_0 \dots x_{m-1}}{g}.$$

With this notion, we have the following proposition whose proof is in Section A of the appendix.

Proposition 2. *Let $\mathscr{C}(I)$ be a decreasing monomial code, then its dual is a decreasing monomial code given by*

$$\mathscr{C}(I)^{\perp} = \mathscr{C}(\mathcal{M} \setminus \check{I}).$$

Notice that this proposition yields the well known result about the dual of a Reed-Muller code $RM(r,m) = \mathscr{C}\left([1; x_{m-r} \dots x_{m-1}]\right)$ where we have

$$\begin{aligned} RM(r,m)^{\perp} &= \mathscr{C}(\mathcal{M} \setminus [x_0 \dots x_{m-r-1}; x_0 \dots x_{m-1}]) \\ &= \mathscr{C}\left([1; x_{r+1} \dots x_{m-1}]\right) \\ &= \mathscr{R}(m-r-1, m). \end{aligned}$$

A straightforward consequence of this is that under some conditions, any decreasing monomial code is weakly self-dual.

Corollary 1. *Let $\mathscr{C}(I)$ be a decreasing monomial code with $|I| \leqslant \frac{1}{2}2^m$. Then*

$$\mathscr{C}(I) \subseteq \mathscr{C}(I)^{\perp} \text{ if and only if for any } f \in I,\ \check{f} \notin I.$$

Polar codes of rate (sufficiently) smaller than $1/2$ generally satisfy this assumption and in the case of rate greater than $\frac{1}{2}$ it is the dual of the polar code that satisfies this assumption. This can be explained by looking at the polarization process that is used to choose the monomials defining the polar code, but explaining this point is beyond the scope of this article. We just wish to add that this assumption is satisfied for the polar codes used in the McEliece cryptosystem that we have attacked in this article. This corollary explains why such codes are weakly self-dual and why the support splitting is of unreasonable complexity in such a case for recovering the unknown permutation between a known permuted polar code and a polar code.

Polynomial codes and monomial codes may have a trivial permutation group. Applying an affine permutation to a monomial code yields a polynomial code, but it is not necessarily a monomial code. To understand the action of a permutation π which is also an affine transformation on $\mathbb{F}_2^m$ we can notice that for any monomial f in $\mathbb{R}_2[x_0, \dots, x_{m-1}]$ we have

$$\mathsf{ev}(f)^{\pi} = \mathsf{ev}(f \circ \pi) \tag{1}$$

where on the lefthand side we view π as a permutation on the coordinates (viewed as elements of $\mathbb{F}_2^m$) whereas on the righthand side we view π as an affine permutation. This equation explains why a monomial code may not be a monomial code after applying an affine permutation and it is rather straightforward to come up with examples of monomial codes that have a trivial permutation group. However by considering the subclass of decreasing monomial codes we obtain codes with a very large permutation group which is the *lower triangular affine group*, that is:

Definition 6 (Lower triangular affine group). *The* lower triangular affine group $\mathbb{LTA}_m$ *on* $\mathbb{F}_2^m$ *is defined as the set of affine transformations over* $\mathbb{F}_2^m$ *of the form* $\boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$ *where* $\boldsymbol{A}$ *is a lower triangular binary matrix with "1"'s on the diagonal and* $\boldsymbol{b}$ *is arbitrary in* $\mathbb{F}_2^m$.

Theorem 1. *The permutation group of a decreasing monomial code in* m *variables contains* $\mathbb{LTA}_m$.

This theorem is proved in Section A of the appendix. This theorem explains why polar codes have a large subgroup of the permutation group of Reed-Muller codes as permutation group. This fact is one of the keys for the cryptanalysis which follows.

Minimum distance of decreasing codes. We first recall some well known facts about the minimum distance of Reed-Muller codes (see for instance [MS86, Ch. 13, §4]):

Theorem 2. *The minimum distance of the Reed-Muller code* $\mathscr{R}(r,m)$ *is* 2^{m-r}. *There is a one to one correspondance between the affine subspaces of* $\mathbb{F}_2^m$ *of dimension* $m-r$ *and the minimum codeword of* $\mathscr{R}(r,m)$*: all minimum codewords are obtained as* $\mathsf{ev}(x'_0 \dots x'_{r-1})$ *where* $x'_0, \dots, x'_{r-1}$ *are obtained from* $x_0, \dots, x_{r-1}$ *by a bijective affine change of coordinates.*

In other words, "up to action of the permutation group there is only one codeword of minimum weight". All these facts have simplified significantly the attack of the McEliece cryptosystem based on Reed-Muller codes in [MS07]. We will see in what follows that polar codes behave differently with this respect.

To understand the minimum distance of a decreasing monomial code, and of a polar code in particular, the following notion is very useful.

Definition 7. *Let* $\mathscr{C}(I)$ *be a decreasing monomial code over* m *variables. We let*

$$r_-(\mathscr{C}(I)) \stackrel{def}{=} \max\left\{r \mid \mathscr{R}(r,m) \subseteq \mathscr{C}(I)\right\}$$
$$r_+(\mathscr{C}(I)) \stackrel{def}{=} \min\left\{r \mid \mathscr{C}(I) \subseteq \mathscr{R}(r,m)\right\}$$

It is readily checked that another way of defining these quantities is that r_- is the largest r for which the monomial $x_{m-r} \dots x_{m-1}$ is in I. On the other hand r_+ is the largest integer r for which $x_0 \dots x_{r-1}$ is in I. These quantities are related to the minimum distance of a decreasing monomial code and its dual through the following result

Proposition 3. *Let $\mathscr{C}(I)$ be a decreasing monomial code over m variables. We have the following properties:*

(i) The minimum distance of $\mathscr{C}(I)$ is equal to $2^{m-r_+(\mathscr{C}(I))}$.
(ii) $r_-(\mathscr{C}(I)^\perp)$ and $r_+(\mathscr{C}(I)^\perp)$ satisfy the equalities:

$$r_-(\mathscr{C}(I)^\perp) = m - 1 - r_+(\mathscr{C}(I))$$
$$r_+(\mathscr{C}(I)^\perp) = m - 1 - r_-(\mathscr{C}(I))$$

(iii) The minimum distance of $\mathscr{C}(I)^\perp$ is equal to $2^{r_-(\mathscr{C}(I))+1}$

This proposition is proved in appendix Section A. A straightforward corollary of these propositions is that the minimum distance of a polar code is always smaller than or equal to the minimum distance of the Reed-Muller code of the same dimension (if it exists) and this is already a strong indication that this minimum distance is rather small (at most the square root of the length for codes of rate greater than $\frac{1}{2}$ for instance). For the polar codes we are interested in in this study, we will be able to find minimum weight codewords in the polar code and its dual with standard algorithms for finding low weight codewords [Ste88,Dum91], since both minimum distances turn out to be rather small.

For Reed-Muller codes there is only one orbit of the permutation group inside the set of minimum codewords. The case of decreasing monomial codes is more complicated. However, and this will be very helpful for classifying these codewords, we have:

Theorem 3. *Each orbit under the action of $\mathbb{LTA}_m$ contained in the set of minimum codewords of the decreasing monomial code $\mathscr{C}(I)$ contains a monomial of I.*

This theorem is proved in Section A of the appendix.

4 Cryptanalysis

We will explain here how we solve the code equivalence problem for a decreasing monomial code $\mathscr{C}(I)$. This can be applied to any polar code and yields an attack that breaks the McEliece scheme based on polar codes proposed in [SK14]. In this section, we use the simplified notation r_- for $r_-(\mathscr{C}(I))$ and r_+ for $r_+(\mathscr{C}(I))$. We also use the notion of signature formalized as follows.

Definition 8 (Signature). *Let $\mathscr{C}$ be a code of length n. Let $\mathcal{G}$ be a subgroup of permutations of $\mathscr{C}$ and W be a subset of $\mathscr{C}$ globally invariant under $\mathcal{G}$. We say that a function $\Sigma(\boldsymbol{c}, \mathscr{C})$ where $\boldsymbol{c}$ belongs to $\mathscr{C}$ is a signature for the action of $\mathcal{G}$ on W if and only if:*

(i) $\Sigma(\boldsymbol{c}, \mathscr{C}) = \Sigma(\boldsymbol{c}^\pi, \mathscr{C}^\pi)$ for π from S_n (i.e. Σ is invariant by permutation),
(ii) $\Sigma(\boldsymbol{c}, \mathscr{C}) \neq \Sigma(\boldsymbol{c}', \mathscr{C})$ if $\boldsymbol{c}$ and $\boldsymbol{c}'$ both belong to W but are not in the same orbit under $\mathcal{G}$ (i.e. Σ takes distinct values for each orbit).

Notice here that a signature always takes the same value on an orbit under $\mathcal{G}$ since if we take $\boldsymbol{c}$ in W and γ is an element of $\mathcal{G}$, then $\Sigma(\boldsymbol{c}, \mathscr{C}) = \Sigma(\boldsymbol{c}^\gamma, \mathscr{C}^\gamma) = \Sigma(\boldsymbol{c}^\gamma, \mathscr{C})$ since γ belongs to the permutation group of the code.

The algorithm for performing the attack can now be summarized as follows:

Step 1. (Minimum weight codewords searching) Search the non-zero minimum weight vectors of $\mathscr{C}(I)$ and $\mathscr{C}(I)^\pi$. We denote these two sets by $W_{\min}$ and $W^\pi_{\min}$ respectively. Note that $W_{\min} = \{\boldsymbol{c} \in \mathscr{C}(I) : |\boldsymbol{c}| = 2^{m-r_+}\}$, $W^\pi_{\min} = \{\boldsymbol{c} \in \mathscr{C}(I)^\pi : |\boldsymbol{c}| = 2^{m-r_+}\}$ and the codeword $\boldsymbol{c}_{\min} \stackrel{\text{def}}{=} \mathsf{ev}(x_0 \cdots x_{r_+-1})$ belongs to $W_{\min}$.

Step 2. (Signature of orbits in $W_{\min}$) Compute the orbits of $W_{\min}$ under the lower triangular subgroup $\mathbb{LTA}_m$ of the affine group and find a signature for these orbits. This signature is based on shortening the dual $\mathscr{C}(I)^\perp$ on the support of $\boldsymbol{c}$ (where $\boldsymbol{c}$ belongs to $W_{\min}$) and computing the dimension of this code and the number of codewords of minimum weight in it.

Step 3. (Computation of orbits in $W^\pi_{\min}$) Use this signature to decompose $W^\pi_{\min}$ into distinct orbits under the group $\pi^{-1}\mathbb{LTA}_m\pi$ and use it to find the orbit of $\boldsymbol{c}^\pi_{\min}$.

Step 4. (Identification of affine spaces) Without loss of generality, we may take any codeword in the orbit of $\boldsymbol{c}^\pi_{\min}$ and declare that it is equal to $\boldsymbol{c}^\pi_{\min}$. Let $\mathcal{I}$ be the support of $\boldsymbol{c}_{\min}$, and $\mathcal{J}$ be the complementary set (that is the set of position for which $\boldsymbol{c}_{\min}$ takes the value 0). Note that with the way we identify positions as elements of $\mathbb{F}_2^m$, $\mathcal{I}$ can be viewed as the affine space $x_0 = x_1 = \cdots = x_{r_+-1} = 1$. The structure of the orbit of $\boldsymbol{c}_{\min}$ is such that the supports of all the codewords in this orbit are affine spaces of the form $x_0 = \varepsilon_0,\ x_1 = \varepsilon_1, \ldots, x_{r_+-1} = \varepsilon_{r_+-1}$, where the ε_i's are arbitrary elements in $\mathbb{F}_2$. Denote this affine space by $A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})$ and let $\boldsymbol{c}_{\min}(\varepsilon_0, \ldots, \varepsilon_{r_+-1})$ be the corresponding codeword. Up to a permutation of $\mathscr{C}^\pi$, we identify all the elements $\boldsymbol{c}_{\min}(\varepsilon_0, \ldots, \varepsilon_{r_+-1})^\pi$. This gives all the affine spaces permuted by π, that is $A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})^\pi \stackrel{\text{def}}{=} \{\pi^{-1}(i) \mid i \in A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})\}$.

Step 5. (Equivalence problem for a short code) Let $\mathcal{J}$ be the set of positions where $\boldsymbol{c}_{\min}$ takes zero values. Notice that the set of positions for which $\boldsymbol{c}^\pi_{\min}$ takes zero values is $\mathcal{J}^\pi$. Then we compute the codes $\mathscr{D} \stackrel{\text{def}}{=} \mathcal{P}_{\mathcal{J}}(\mathscr{C})$ and $\mathscr{D}^\pi \stackrel{\text{def}}{=} \mathcal{P}_{\mathcal{J}^\pi}(\mathscr{C}^\pi)$. We solve the code equivalence problem for $\mathscr{D}$ and $\mathscr{D}^{\pi'}$ where π' is the restriction of the permutation π to the affine space $\mathcal{I}$. Notice that this problem is solved for much shorter codes than the original system.

Step 6. (Induction step) Let $\boldsymbol{c}^i = \mathsf{ev}(x_0 \ldots, x_{i-1})$ with $\boldsymbol{c}^0$ being $\mathsf{ev}(1)$, that is the all-one codeword. Notice that $\boldsymbol{c}_{\min} = \boldsymbol{c}^{r_+}$, and let $\mathcal{J}^i$ be the set of positions for which $\boldsymbol{c}^i$ takes the value 0. Denote by $\mathscr{D}^i = \mathcal{P}_{\mathcal{J}^i}(\mathscr{C})$. Solve for $i = r_+ - 1, \ldots, 0$ the code equivalence problem for the pair $(\mathscr{D}^i, (\mathscr{D}^i)^{\pi^i})$ by using the solution to the code equivalence problem $(\mathscr{D}^{i+1}, (\mathscr{D}^{i+1})^{\pi^{i+1}})$ where π^i is the restriction of π to the set of positions of $\mathscr{D}^i$.

The last code equivalence problem we solve here (namely for $i = 0$) is just a solution to the original code equivalence problem.

4.1 Step 1 – Minimum weight codewords searching.

Finding the codewords of $\mathscr{C}(I)^\pi$ can be performed by applying Dumer's algorithm [Dum91]. The complexity of this algorithm for finding a codeword of weight w in a code of rate R can be estimated as $O\left(e^{-w\ln(1-R)(1+o(1))}\right)$ when w is a sublinear function of the length (see [CTS15] for more details) and the length n of the code goes to infinity. For monomial codes it can be readily checked that codes with rate greater than some constant $\varepsilon > 0$ have minimum distance at most $O(\sqrt{n})$ (this comes from straightforward and well known results about the minimum distance of Reed-Muller codes and Proposition 3). This is clearly achievable for the polar codes we have considered in this article.

On the other hand, all the minimum codewords of $\mathscr{C}(I)$ are easily obtained by using Theorem 3: $W_{\min}$ decomposes into orbits under the action of $\mathbb{LTA}_m$ where each orbit contains one of the monomials of I of degree r_+.

4.2 Step 2 – Signature of orbits in $W_{\min}$

To distinguish between the codewords of $W_{\min}$ we have first chosen a monomial in each of the orbits under $\mathbb{LTA}_m$ that decompose $W_{\min}$. For each of such monomials g we have computed the dual of the shortened code $\mathscr{D} \stackrel{\text{def}}{=} \left(\mathcal{S}_{\mathcal{J}}\left(\mathscr{C}(I)\right)\right)^\perp$ with respect to the support $\mathcal{J}$ of $\mathsf{ev}(g)$. It has turned out that, for the polar codes we have considered, the pair (number of codewords of weight 2^{r_-} in $\mathscr{D}$, dimension of $\mathscr{D}$) was discriminant enough to yield a signature of the orbit. This critical quantity 2^{r_-} occurs because we have

Theorem 4. *Let $g = x_{i_1}\dots x_{i_{r_+}}$ be a monomial of degree r_+ in I. Denote by $\mathsf{supp}(g)$ the support of $\mathsf{ev}(g)$, then the minimum distance of $\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^\perp$ is equal to 2^{r_-} if and only if there exists a monomial h in $\mathcal{M}\setminus\check{I}$ such that:*

(i) the number of variables of h that are also variables of g is $r_+ - 1$,

(ii) the number of variables of h that are also variables of $\check{g}$ is $m - r_- - r_+$.

This theorem is proved in Section B of the appendix.

4.3 Step 3 – Computation of orbits in $W^{\pi}_{\min}$

The signature Σ that has been found in the previous step is now applied to $W^{\pi}_{\min}$. It gives the orbits of $W^{\pi}_{\min}$ with respect to the conjugate group $\pi^{-1}\mathcal{G}\pi$. Indeed, it can be verified that

Proposition 4. *W^{π}_{min} is invariant by the action of $\pi^{-1}\mathbb{LTA}_m\pi$ and if Σ is a signature for W_{min} under the action of $\mathbb{LTA}_m$, then it is also a signature for the action of $\pi^{-1}\mathbb{LTA}_m\pi$ on W^{π}_{min}.*

We use this signature for finding the orbit of $\boldsymbol{c}_{\min}$. This orbit has a particularly nice structure:

Proposition 5. *The orbit of $\boldsymbol{c}_{min}$ under $\mathbb{LTA}_m$ consists of 2^{r_+} codewords that are of the form $\boldsymbol{c}_{min}(\varepsilon_0, \ldots, \varepsilon_{r_+-1})$ where the ε_i's are arbitrary elements of $\mathbb{F}_2$. The orbit of $\boldsymbol{c}^{\pi}_{min}$ under $\pi^{-1}\mathbb{LTA}_m\pi$ is given by 2^{r_+} codewords of weight 2^{m-r_+} that have disjoint supports which are the permuted versions $A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})^{\pi}$ of the affine spaces $A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})$.*

In other words, finding this orbit in $W^{\pi}_{\min}$ and looking at the support of the codewords that we have found in this way allows us to find the support of the permuted versions $A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})^{\pi}$ of the affine spaces $A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})$.

4.4 Step 4 – Identification of affine spaces

There are several ways to identify the permuted versions of the affine spaces we are interested in. One of the simplest way, which worked for the $[2048, 614]$ polar code that we studied, is by computing the dimensions of certain spaces. First we take any codeword in the orbit of $\boldsymbol{c}_{\min}$. Such codeword is of the form $\boldsymbol{c}^{\gamma\pi}_{\min}$ where γ is a permutation leaving $\mathscr{C}(I)$ invariant. In other words, up to applying the permutation group, we can safely declare that this codeword is $\boldsymbol{c}^{\pi}_{\min}$. Let $\mathcal{I}_0$ be the support of $\boldsymbol{c}_{\min} = \boldsymbol{c}(1, \ldots, 1)$. We choose $\mathcal{I}'_0$ be the support of the codeword $\boldsymbol{c}(\underbrace{1, \ldots, 1}_{(r_+-1) \text{ times}}, 0)$. Notice that $\mathcal{I} \stackrel{\text{def}}{=} \mathcal{I}_0 \cup \mathcal{I}'_0$ is the support of the codeword $\mathsf{ev}(x_0 \ldots x_{r_+-2})$. We compute the dimension of the code $\mathcal{P}_{\mathcal{I}}(\mathscr{C}(I))$. Now, we let $\mathcal{J}_0, \ldots, \mathcal{J}_{2^{r_+}-1}$ be the supports of the codewords that are in the orbit of $\boldsymbol{c}^{\pi}_{\min}$, with $\mathcal{J}_0$ being the support of the codeword $\boldsymbol{c}^{\gamma\pi}_{\min}$ that has been chosen. We compute the dimensions of the codes $\mathcal{P}_{\mathcal{J}_0 \cup \mathcal{J}_i}(\mathscr{C}(I)^{\pi})$ for $i = 1, \ldots, 2^{r_+} - 1$. It turns out that there is generally a single space $\mathcal{J}_i$ such that $\dim(\mathcal{P}_{\mathcal{J}_0 \cup \mathcal{J}_i}(\mathscr{C}(I)^{\pi})) = \dim(\mathcal{P}_{\mathcal{I}}(\mathscr{C}(I)))$. We pair these two spaces $\mathcal{J}_0$ and $\mathcal{J}_i$ together. This process can be used to pair together all the spaces $A(\varepsilon_0, \ldots, \varepsilon_{r_+-2}, 0)^{\gamma\pi}$ and $A(\varepsilon_0, \ldots, \varepsilon_{r_+-2}, 1)^{\gamma\pi}$ by pairing together $\mathcal{J}_i$ and $\mathcal{J}_j$ when $\mathcal{J}_j$ is the only space for a given i such that

$$\dim\left(\mathcal{P}_{\mathcal{J}_i \cup \mathcal{J}_j}(\mathscr{C}(I)^{\pi})\right) = \dim\left(\mathcal{P}_{\mathcal{I}}(\mathscr{C}(I))\right).$$

In such a case, $\mathcal{J}_i$ and $\mathcal{J}_j$ necessarily correspond to $A(\varepsilon_0, \ldots, \varepsilon_{r_+-2}, 0)^{\gamma\pi}$ and $A(\varepsilon_0, \ldots, \varepsilon_{r_+-2}, 1)^{\gamma\pi}$ for a certain $(\varepsilon_0, \ldots, \varepsilon_{r_+-2}) \in \mathbb{F}_2^{r_+-1}$. In other words, we know after this process all the spaces $A(\varepsilon_0, \ldots, \varepsilon_{r_+-2})^{\gamma\pi} = A(\varepsilon_0, \ldots, \varepsilon_{r_+-2}, 0)^{\gamma\pi} \cup A(\varepsilon_0, \ldots, \varepsilon_{r_+-2}, 1)^{\gamma\pi}$. We can carry on this process with the codeword $\boldsymbol{c} = \mathsf{ev}(x_0 \ldots x_{r_+-1})$ instead of $\boldsymbol{c}_{\min}$ and recover all the permuted affines spaces $A(1)^{\gamma\pi}, A(1, 1)^{\gamma\pi}, \ldots, A(\underbrace{1, 1, \ldots, 1}_{r_+ \text{ times}})^{\gamma\pi}$ for some permutation γ leaving $\mathscr{C}(I)$ invariant.

4.5 Step 5 – Equivalence problem for a short decreasing monomial code

We now have to solve the code equivalence problem for $\mathscr{D}$ which is a code of length 2^{m-r_+} which is much shorter than the original code. It is also straightfor-

ward to check that it is a decreasing monomial code. We can for instance carry out the process again that we saw before. For the $[2048, 614]$ polar code that we studied, we can even compute the whole permutation group of the code which is much closer to the whole affine group. It is here a code of length 32 that contains $\mathscr{R}(2,5)$ and is contained in $\mathscr{R}(3,5)$. We do not detail this point here, since there are many ways to actually solve the problem.

4.6 Step 6 – Induction step

The idea here is to reconstruct the permutation $\hat{\pi}$ given that we already know its action on the support of $\boldsymbol{c}_{\min}$. More precisely, the code equivalence problem that we solve here is:

Problem 2 (Code equivalence search problem with side information). Given $(\mathscr{C}, \mathscr{C}^{\pi})$ and t pairs of code positions (i_0, j_0), $(i_1, j_1), \ldots, (i_{t-1}, j_{t-1})$, find $\hat{\pi}$ such that $\mathscr{C}^{\hat{\pi}} = \mathscr{C}^{\pi}$ and $\hat{\pi}(i_s) = j_s$ for all $s \in \{0, 1, \ldots, t-1\}$

We use the following algorithm for solving this problem (we let here $\mathcal{I} \stackrel{\text{def}}{=} \{i_0, \ldots, i_{t-1}\}$ and $\mathcal{J} \stackrel{\text{def}}{=} \{j_0, \ldots, j_{t-1}\}$)

1. we pick a certain number ℓ of codewords $\boldsymbol{c}(0), \ldots, \boldsymbol{c}(\ell-1)$ of $\mathscr{C}$.
2. Let $\mathscr{C}(j)$ the set of codewords of $\mathscr{C}$ which coincide with $\boldsymbol{c}(j)$ on the positions belonging to $\mathcal{J}$. We also define $\mathscr{C}(i)^{\pi}$ as the set of codewords of $\mathscr{C}^{\pi}$ that coincide with $\boldsymbol{c}(i)^{\pi}$ on $\mathcal{I}$.
3. We compute for all i in $0, 1, \ldots, \ell-1$ and all positions j which are not in $\mathcal{J}$, the number $\Sigma(i,j)$ which is the number of codewords of minimum weight in $\mathcal{P}_j(\mathscr{C}(i))$, and similarly for all all positions j that are not in $\mathcal{I}$, the number $\Sigma^{\pi}(i,j)$ which is the number of codewords of minimum weight in $\mathcal{P}_j(\mathscr{C}(i)^{\pi})$.
4. We declare for u which is not in $\mathcal{I}$ that $\hat{\pi}(u) = v$ if there exists a unique v which does not belong to $\mathcal{J}$ such that $\Sigma(i,v) = \Sigma^{\pi}(i,u)$ for all i in $\{0, 1, \ldots, \ell-1\}$.

It is straightforward to verify that this algorithm outputs the unique $\hat{\pi}$ solving the problem in this case. We have also encountered cases, where even with the knowledge we have on $\hat{\pi}$, we have different solutions. In such a case, we were able to compute how many solutions we had and add to the set of pairs (i_s, j_s) an additional pair (or additional pairs) which gives a unique solution.

5 Implementation of the Attack on a [2048, 614]-Polar Code

We implemented the $[2048, 614]$-polar code as follows. The Shannon limit for the noise on a binary symmetric channel of crossover probability p that a code of rate $\frac{614}{2048}$ is able to sustain is about $p = 0.19$. We devised the polar code for a slightly smaller error rate of $p = 0.17$ and chose the 614 best rows of $\boldsymbol{G}_{11}$ which give the best performances for the successive cancellation decoder. Such a code

is able to correct more than 200 errors with a small error probability- this should be compared to the 130 errors that a Goppa code of the same rate is able to tolerate. In the case of a Goppa code we have about 70 bits of security against message attacks based on generic linear codes decoding algorithms, whereas we have more than 105 bits of security for the polar code.

We first checked that this code $\mathscr{C}$ and its dual $\mathscr{C}^{\perp}$ are both decreasing monomial codes and computed all the minimum weight codewords by using Theorem 3. The conditions of Corollary 1 were met and the code was weakly self-dual $\mathscr{C} \subset \mathscr{C}^{\perp}$. The minimum distance of $\mathscr{C}$ turned out to be equal to 32 and there were 42176 codewords of this weight, whereas the minimum distance of $\mathscr{C}^{\perp}$ was 8 and there were 6912 codewords of this weight in the dual. The same number of codewords were found by Dumer's algorithm in $\mathscr{C}^{\pi}$ and in $(\mathscr{C}^{\pi})^{\perp}$. It tooks 27 seconds to find these codewords in $\mathscr{C}^{\pi}$ and 3 seconds to find these codewords in $(\mathscr{C}^{\pi})^{\perp}$ on a 8-core XEON E3-1240 running at 3.40 GHz.

But the most time consuming part was Step 6 of the attack when we have to compute the various $\Sigma(i,j)$'s that are needed. This is done again by using Dumer's algorithm. The difference with obtaining codewords of minimum weight of the polar code is that in the polar case we know beforehand the number of minimum weight codewords by using a counting procedure based on Theorem 3 and we can stop the search procedure once we have the right amount of different codewords. However when we compute $\Sigma(i,j)$ we do not know beforehand the number of minimum weight codewords in $\mathcal{P}_j(\mathscr{C}(i))$ and we use a probabilistic procedure based on the coupon collector problem : once we have found n different minimal codewords, where on average we have found each codeword $\alpha \ln n$ times we stop the procedure for a certain value of α greater than 1. Here we have taken α to be equal to 3. In this case, to speed up the computation we chose the $c(i)$'s to be minimum weight codewords of $\mathscr{C}$. More than 80% of the total computation is actually taken for the last step of induction where we recover a permutation for the whole $[2048, 614]$ code from the partial permutation acting on half its positions. This takes about 227 hours and the total computation time is about 280 hours. This part of the attack is very likely to be improved significantly if need be.

6 Conclusion

Despite the fact that the code equivalence problem for binary polar codes is a hard instance for the Support Splitting Algorithm, we have shown in this paper that it can nevertheless be solved rather efficiently by a more sophisticated algorithm consisting in (i) looking for minimum weight codewords, (ii) classifying them by using our knowledge of the automorphism group of the polar code to find a particular minimum weight codeword, (iii) use this particular codeword to partition the code positions into affine spaces, (iv) puncture the set of positions with respect to all these affine spaces but one, and solve the code equivalence problem on this reduced problem. We use this to solve the code equivalence problem by induction on increasing affine spaces.

This allows to break the McEliece cryptosystem for the parameters proposed in [SK14]. It is likely that the only way to avoid this kind of attack (or possible improvements on it) is to look for polar code parameters for which we are unable to find minimum weight codewords either in the code or in its dual. This would require to change significantly the parameters proposed in [SK14] that would make such polar codes much less attractive for a use in a McEliece cryptosystem.

To obtain this attack we have proposed a new code family, that we call decreasing monomial codes containing as a particular subcase Reed-Muller codes and binary polar codes. These decreasing monomial codes have a very large permutation group that gives some insight about the permutation group of polar codes. This knowledge on the permutation group of polar codes we obtained could also be used in other settings, for instance to improve the decoding performances of polar codes.

This attack can be considered as a first step towards studying the polar code based McEliece scheme proposed in [HSEA14]. Our attack does not apply directly to this scheme since it is based on taking a particular kind of random subcode of the polar code. In such a case, the system does not consist in solving the code equivalence problem (or we have to solve as many instances as the number of possible subcodes of this kind which becomes unfeasible in this case). However it seems that some of the tools provided here, and a particular property of polar codes, might also be used to attack such a scheme. Indeed, taking the square of the polar code or the square of its dual (with the definition of a square code given in [CGG$^+$14]) gives a code which is not the full space in many cases. If the subcode of a polar code was chosen uniformly at random among the spaces of some prescribed codimension inside the code, then the square of such codes would be almost always equal to the square of the polar code when the codimension is large enough. This would give an attack since the square of a polar code which is a decreasing monomial code is readily seen to be a decreasing monomial code itself. From there we can solve the code equivalence problem on the square of this code by using the tools given in this paper. This reveals the secret permutation and breaks the system. With the way the subcodes are chosen in [HSEA14] this does not happen, but still the square of the subcode is a very large subcode of the square of the polar code itself and this looks highly suspicious.

References

[Arı09] Erdal Arıkan. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. Inform. Theory*, 55(7):3051–3073, 2009.

[BBC08] Marco Baldi, Marco Bodrato, and Franco Chiaraluce. A new analysis of the TMcEliece cryptosystem based on QC-LDPC codes. In *Proceedings of the 6th international conference on Security and Cryptography for Networks*, SCN '08, pages 246–262, Berlin, Heidelberg, 2008. Springer-Verlag.

[BBD09] Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen, editors. *Post-Quantum Cryptography.* Springer-Verlag, 2009.

[BC07] Marco Baldi and Franco Chiaraluce. Cryptanalysis of a new instance of McEliece cryptosystem based on QC-LDPC codes. In *Proc. IEEE Int. Symposium Inf. Theory - ISIT*, pages 2591–2595, Nice, France, June 2007.

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

[BLP10] Daniel J. Bernstein, Tanja Lange, and Christiane Peters. Wild McEliece. In Alex Biryukov, Guang Gong, and DouglasR. Stinson, editors, *Selected Areas in Cryptography*, volume 6544 of *Lecture Notes in Comput. Sci.*, pages 143–158, 2010.

[BLP11] Daniel J. Bernstein, Tanja Lange, and Christiane Peters. Wild McEliece Incognito. In Bo-Yin Yang, editor, *Post-Quantum Cryptography 2011*, volume 7071 of *Lecture Notes in Comput. Sci.*, pages 244–254. Springer Berlin Heidelberg, 2011.

[CFS01] Nicolas Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Comput. Sci.*, pages 157–174, Gold Coast, Australia, 2001. Springer.

[CGG+14] Alain Couvreur, Philippe Gaborit, Valérie Gauthier-Umaña, Ayoub Otmani, and Jean-Pierre Tillich. Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes. *Des. Codes Cryptogr.*, 73(2):641–666, 2014.

[CMCP14] Alain Couvreur, Irene Márquez-Corbella, and Ruud Pellikaan. A polynomial time attack against algebraic geometry code based public key cryptosystems. In *Proc. IEEE Int. Symposium Inf. Theory - ISIT 2014*, pages 1446–1450, June 2014.

[COT14] Alain Couvreur, Ayoub Otmani, and Jean-Pierre Tillich. Polynomial time attack on wild McEliece over quadratic extensions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Comput. Sci.*, pages 17–39. Springer Berlin Heidelberg, 2014.

[CT91] Thomas M. Cover and Joy A. Thomas. *Information Theory*. Wiley Series in Telecommunications. Wiley, 1991.

[CTS15] Rodolfo Canto-Torres and Nicolas Sendrier. Analysis of information set decoding for a sub-linear error weight, 2015. preprint.

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

[FM08] Cédric Faure and Lorenz 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.

[FPdP14] Jean-Charles Faugère, Ludovic Perret, and Frédéric de Portzamparc. Algebraic attack against variants of McEliece with Goppa polynomial of a special form. In *Advances in Cryptology - ASIACRYPT 2014*, volume 8873 of *Lecture Notes in Comput. Sci.*, pages 21–41, Kaoshiung, Taiwan, R.O.C., December 2014. Springer.

[HSEA14] R Hooshmand, M Koochak Shooshtari, T Eghlidos, and MR Aref. Reducing the key length of McEliece cryptosystem using polar codes. In *2014 11th International ISC Conference on Information Security and Cryptology (ISCISC)*, pages 104–108. IEEE, 2014.

[JM96] Heeralal Janwa and Oscar Moreno. McEliece public key cryptosystems using algebraic-geometric codes. *Des. Codes Cryptogr.*, 8(3):293–307, 1996.

[Kra91] David Kravitz. Digital signature algorithm. US patent 5231668, July 1991.

[KU10] Satish B. Korada and Rüdiger Urbanke. Polar codes are optimal for lossy source coding. *IEEE Trans. Inform. Theory*, 56(4):1751–1768, 2010.

[LJ12] Carl Löndahl and Thomas Johansson. A new version of McEliece PKC based on convolutional codes. In *Information and Communications Security, ICICS*, volume 7168 of *Lecture Notes in Comput. Sci.*, pages 461–470. Springer, 2012.

[LT13] Grégory Landais and Jean-Pierre Tillich. An efficient attack of a McEliece cryptosystem variant based on convolutional codes. In P. Gaborit, editor, *Post-Quantum Cryptography'13*, volume 7932 of *Lecture Notes in Comput. Sci.*, pages 102–117. Springer, June 2013.

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

[Mn07] Lorenz Mi nder. *Cryptography based on error correcting codes.* PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2007.

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

[MS07] Lorenz Minder and Amin Shokrollahi. Cryptanalysis of the Sidelnikov cryptosystem. In *Advances in Cryptology - EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Comput. Sci.*, pages 347–360, Barcelona, Spain, 2007.

[MTSB12] Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo S. L. M. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. *IACR Cryptology ePrint Archive, Report2012/409*, 2012, 2012.

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

[OT12] Ayoub Otmani and Jean-Pierre Tillich. On the Design of Code-Based Signatures. In *Code-based Cryptography Workshop (CBC 2012)*, Lyngby, Denmark, May 2012.

[OTD08] Ayoub Otmani, Jean-Pierre Tillich, and Léonard 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, Beijing, China, April 28-30 2008. LMIB Beihang University.

[PR97] Erez Petrank and Ron. Roth. Is code equivalence easy to decide? *IEEE Trans. Inform. Theory*, 43(5):1602–1604, 1997.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

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

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[Sid94] Vladimir Michilovich Sidelnikov. A public-key cryptosytem based on Reed-Muller codes. *Discrete Math. Appl.*, 4(3):191–207, 1994.

[SK14] Sujan Raj Shrestha and Young-Sik Kim. New McEliece cryptosystem based on polar codes as a candidate for post-quantum cryptography. In *2014 14th*

International Symposium on Communications and Information Technologies (ISCIT), pages 368–372. IEEE, 2014.

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

[Ste88] Jacques 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 Comput. Sci.*, pages 106–113. Springer, 1988.

[Wie10] Christian Wieschebrink. Cryptanalysis of the Niederreiter public key scheme based on GRS subcodes. In *Post-Quantum Cryptography 2010*, volume 6061 of *Lecture Notes in Comput. Sci.*, pages 61–72. Springer, 2010.

A Proofs of the results of Section 3

A.1 Proof of Proposition 2

In order to prove this result, we first prove a few lemmas about the partial order we introduced.

Lemma 2. *For all f and g in $\mathcal{M}$, $f \preceq g$ if and only if $\check{f} \succeq \check{g}$.*

Proof. Let $f = x_{i_1} \dots x_{i_s}$ and $g = x_{j_1} \dots x_{j_t}$ with $s \leqslant t$ and $i_1 < \cdots < i_s$, $j_1 < \cdots < j_t$. Then we have two cases:

- if $\deg f = \deg g$ then by definition of the order we have $i_\ell \leqslant j_\ell$ for all $j = 1, \dots, s$. Consider the ℓ-th variable $x_{i'_\ell}$ in the monomial $\check{f}$ and the ℓ-th variable $x_{j'_\ell}$ in the monomial $\check{g}$. Let us define

$$\varphi(u) \stackrel{\text{def}}{=} \ell - 1 + \#\{i_a : i_a \leqslant u\}$$
$$\gamma(u) \stackrel{\text{def}}{=} \ell - 1 + \#\{j_a : j_a \leqslant u\}$$

  Observe now that
  (i) since $\varphi(u+1)$ is either equal to $\varphi(u)$ or to $\varphi(u)+1$ and since $\varphi(0) \geqslant 0$, $\varphi(m-1) \leqslant m-1$, there exists at least one u such that $\varphi(u) = u$,
  (ii) when $\varphi(u) = u$ this means that there exist exactly ℓ variables x_b for b in $\{0, 1, \dots, u\}$ that belong to the monomial $\check{f}$.
  All this implies that i'_ℓ is the smallest index u such that $\varphi(u) = u$ (or what amounts to the same it is the smallest index u such that $\varphi(u) \leqslant u$). A similar property holds for j'_ℓ. In other words

$$i'_\ell = \min\{u : \varphi(u) \leqslant u\} \tag{2}$$
$$j'_\ell = \min\{u : \gamma(u) \leqslant u\} \tag{3}$$

  From the fact that $j_a \geqslant i_a$ for all a in $\{1, \dots, s\}$ we have that for all indices u

$$\varphi(u) \geqslant \gamma(u) \tag{4}$$

  On the other hand, we know that $i'_\ell = \varphi(i'_\ell)$, where the righthand term is larger than or equal to $\gamma(i'_\ell)$ by using (4). Therefore $\gamma(i'_\ell) \leqslant i'_\ell$, and by using (3) we deduce that $j'_\ell \leqslant i'_\ell$.

– if $\deg f < \deg g$ then by definition of the order: $f \preceq g \Leftrightarrow \exists\ g_1 \in \mathcal{M}$ s.t. $g = g_1 g_2$ with $\deg g_1 = \deg f$ and $f \preceq g_1$. From the first case we deduce that $\check{f} \succeq \check{g_1}$. On the other hand one checks immediately that $\check{g_1} \succeq \check{g}$. From these two inequalities we deduce $\check{f} \succeq \check{g}$.

Corollary 2. *Let $I \subseteq \mathcal{M}$ be a decreasing set then $\mathcal{M} \setminus \check{I}$ is a decreasing set.*

Proof. Let h be a monomial that belongs to $\mathcal{M} \setminus \check{I}$, and let g be a monomial such that $g \preceq h$. If $g \notin \mathcal{M} \setminus \check{I}$ then it would mean that there exists $f \in I$ such that $g = \check{f}$. This means that $\check{f} \preceq h$ and by using Lemma 2 we would get $\check{h} \preceq \check{\check{f}} = f$. Since I is a decreasing set, $\check{h} \in I$, that is to say, $\check{\check{h}} = h \in \check{I}$ which contradicts the assumption. Therefore $\mathcal{M} \setminus \check{I}$ is a decreasing set.

These lemmas can now be used to prove Proposition 2 that we recall below.

Proposition. *Let $\mathscr{C}(I)$ be a decreasing monomial code, then its dual is a decreasing monomial code given by*

$$\mathscr{C}(I)^{\perp} = \mathscr{C}(\mathcal{M} \setminus \check{I}).$$

Proof. As $\left|\check{I}\right| = |I|$, we have $\dim \mathscr{C}(\mathcal{M} \setminus \check{I}) = |\mathcal{M}| - |\check{I}| = |\mathcal{M}| - |I| = 2^m - \dim \mathscr{C}(I) = \dim \mathscr{C}(I)^{\perp}$, so we need to prove only one inclusion.

Let $f \in \mathcal{M} \setminus \check{I}$ and consider $g \in I$. Notice that

$$< \mathsf{ev}(f), \mathsf{ev}(g) >=< \mathsf{ev}(fg), \mathsf{ev}(1) >$$

where $< .,. >$ stands for the standard inner product in $\{0,1\}^{2^m}$: $< \boldsymbol{x}, \boldsymbol{y} >= \sum_i x_i y_i$. Observe now that fg is a monomial and that the only monomial whose evaluation is not orthogonal (with respect to $<,>$) to the all 1 vector is the "full" monomial $x_1 \dots x_m$. Assume now that we are in such a case: $fg = x_1 \cdots x_m$. This means that $\check{g}$ is a divisor of f. A divisor of a monomial is always smaller than or equal to this monomial with our definition of order. Therefore $\check{g} \preceq f$. From Corollary 2 we know that $\mathcal{M} \setminus \check{I}$ is a decreasing set and that this would imply $\check{g} \in \mathcal{M} \setminus \check{I}$. This would imply that $\check{\check{g}} = g$ would belong to $\check{\mathcal{M}} \setminus \check{\check{I}} = \mathcal{M} \setminus I$. This would contradict the assumption that g belongs to I. Therefore we proved by contradiction that $\mathscr{C}(\mathcal{M} \setminus \check{I}) \subseteq \mathscr{C}(I)^{\perp}$.

A.2 Proof of Theorem 1

Let us recall this theorem:

Theorem. *The permutation group of a decreasing monomial code in m variables contains $\mathbb{LTA}_m$.*

Proof. Let $\mathscr{C}(I)$ be a decreasing monomial code and let π be in $\mathbb{LTA}_m$. Consider $\boldsymbol{x}$ in $\mathbb{F}_2^m$. Let $\boldsymbol{x}' \stackrel{\text{def}}{=} \pi(\boldsymbol{x})$. There exist binary numbers a_{ij} and ε_i such that for any i in $\{0, \dots, m-1\}$ we have

$$x'_i = x_i + \sum_{j<i} a_{ij} x_j + \varepsilon_i.$$

An affine permutation π acts also in a natural way on monomials, with its action being defined by

$$\pi(x_{i_1} \dots x_{i_s}) \stackrel{\text{def}}{=} x'_{i_1} \dots x'_{i_s}.$$

In other words the action of an affine permutation π on a monomial f is given by $f \circ \pi$. Observe that this action is such that

$$\mathsf{ev}(f)^\pi = \mathsf{ev}(f \circ \pi).$$

Choose now a monomial f in I and use the observation above. We can expand $f \circ \pi$ and verify that it is a sum of monomials that are smaller than f with respect to the order $\preceq$ that we introduced. Since I is a decreasing set, then all these monomials belong to I as well and therefore we obviously have that $\mathsf{ev}(f \circ \pi)$ is also in $\mathscr{C}(I)$. $\mathscr{C}(I)$ is therefore invariant by π.

A.3 Proof of Proposition 3

Let $\mathscr{C}(I)$ be a decreasing monomial code. Let us start by proving Point (i), namely that *the minimum distance of* $\mathscr{C}(I)$ *is equal to* $2^{m-r_+(\mathscr{C}(I))}$. This follows on the spot by noticing that r_+ is also the largest degree of a monomial in I. If we consider the evaluation of this monomial we obtain a codeword of weight $2^{m-r_+(\mathscr{C}(I))}$. This implies that the minimum distance of $\mathscr{C}(I)$ is smaller than or equal to this quantity. On the other hand, the minimum distance of $\mathscr{C}(I)$ is larger than or equal to the minimum distance of $\mathscr{R}(r_+, m)$ which is equal to $2^{m-r_+(\mathscr{C}(I))}$ by using Theorem 2. This implies our claim.

Consider now the second point that we recall below

$$r_-(\mathscr{C}(I)^\perp) = m - 1 - r_+(\mathscr{C}(I)) \tag{5}$$
$$r_+(\mathscr{C}(I)^\perp) = m - 1 - r_-(\mathscr{C}(I)) \tag{6}$$

This follows immediately from Proposition 2: $\mathscr{C}(I)^\perp = \mathscr{C}(\mathcal{M} \setminus \check{I})$ and the alternative definitions of $r_-(\mathscr{C}(I)^\perp)$ and of $r_+(\mathscr{C}(I)^\perp)$ which are respectively the largest degree r such that all monomials of degree r are monomials in $\mathcal{M} \setminus \check{I}$ and the largest degree of a monomial that belongs to $\mathcal{M} \setminus \check{I}$.

The third point, namely that *the minimum distance of* $\mathscr{C}(I)^\perp$ *is equal to* $2^{r_-(\mathscr{C}(I))+1}$ is a straightforward of Point(i) applied to the monomial code $\mathscr{C}(I)^\perp$ and by using (6).

A.4 Proof of Proposition 3

Here we want to prove that any minimum weight codeword $\boldsymbol{c}$ in a decreasing monomial code $\mathscr{C}(I)$ can be written as $\boldsymbol{c} = \mathsf{ev}(f)^\pi$ where f is a monomial in I and π an element of $\mathbb{LTA}_m$.

Note that from Proposition 3 we know that a minimum weight codeword of $\mathscr{C}(I)$ is also a minimum codeword of $\mathscr{R}(r_+(\mathscr{C}(I)), m)$. For simplicity we will simply write r_+ for $r_+(\mathscr{C}(I))$ from now on. By using Theorem 2, we know that

$\boldsymbol{c}$ can be written as the evaluation of the product of r_+ independent affine forms $x'_0 \stackrel{\text{def}}{=} \varepsilon_0 + \sum_j a_{0j}x_j, \cdots, x'_{r_+-1} \stackrel{\text{def}}{=} \varepsilon_{r_+-1} + \sum_j a_{r_+-1,j}x_j$ where the ε_i's are elements of the binary field $\mathbb{F}_2$. We claim now that there are r_+ independent affine forms $x"_0, \ldots, x"_{r_+-1}$ such that:

(i) $\mathsf{ev}(x'_0 \ldots x'_{r_+-1}) = \mathsf{ev}(x"_0 \ldots x"_{r_+-1})$,

(ii) for all $i \in \{0, \ldots, r_+ - 1\}$ we have that the $x"_i$'s can be written as $\varepsilon'_i + \sum_{j<\varphi(i)} a'_{\varphi(i),j}x_j$, where φ is some permutation of $\{0, 1, \ldots, m-1\}$ and the ε'_i's and $a'_{\varphi(i),j}$ are binary.

This is easy to check by considering the affine form x'_i that involves the "largest" variable x_j (the one consisting of the largest index j). Let x_{j_0} be this variable. We may assume without loss of generality that this is x'_0. We can check now that

$$\mathsf{ev}(x'_0x'_1 \ldots x'_{r_+-1}) = \mathsf{ev}(x'_0x'''_1 \ldots x'''_{r_+-1}),$$

where $x'''_i = x'_i - x'_0 - 1$ if x'_i involves the variable x_{j_0} and $x'''_i = x'_i$ otherwise. Observe now that the $r_+ - 1$ affine forms $x'''_1, \ldots, x'''_{r_+-1}$ involve only variables x_j which are such that $j < j_0$. We can carry on this process with these $r_+ - 1$ (independent) affine forms $x""_1, \ldots, x'''_{r_+-1}$ by considering the variable x_j which is the largest among the variables that are involved in these affine forms and so on and so forth. We end up with r_+ affine forms $x"_0, \ldots, x"_{r_+-1}$ which have exactly the aforementioned properties (i) and (ii). Consider the monomial $x_{j_0} \ldots x_{j_{r_+-1}}$ which is the product of the "largest" variable x_j in each of these $x"_i$'s. This monomial has to belong to I and we obviously have $\mathsf{ev}(x"_0 \ldots x"_{r_+-1}) = \mathsf{ev}(\pi(x_{j_0} \ldots x_{j_{r_+-1}}))$ for some π in $\mathbb{LTA}_m$. This proves our theorem.

B Proof of the results of Section 4

B.1 Proof of Theorem 4

We will first begin this proof by proving a general result about the dual of shortened monomial codes.

Lemma 3. *Let $\mathscr{C}(I)$ be a decreasing monomial code and $g \in I$. Let $\mathsf{supp}(g)$ be the support of $\mathsf{ev}(g)$. We denote by $E\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^\perp$ the dual of the shortened code in $\mathsf{supp}(g)$ that we have extended by zeros in the positions in which we have shortened the code. Then*

$$E\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^\perp = \{\mathsf{ev}((1+g)f) : f \in \mathcal{M} \setminus \check{I}\}$$

Proof. Recall that we have

$$\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^\perp = \mathcal{P}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)^\perp\right)$$

We know that $\mathscr{C}(I)^\perp = \mathscr{C}(\mathcal{M} \setminus \check{I})$. The lemma follows from this and the fact the $\mathsf{ev}(1+g)$ takes value 1 on the complementary of $\mathsf{supp}(g)$ and 0 on $\mathsf{supp}(g)$.

The following notation turns out to be convenient.

Notation 2. *For a monomial $g = x_{i_1} \dots x_{i_s}$, its set of indices $Ind(g)$ is given by $\{i_1, \dots, i_s\}$ and its intersection $g \wedge h$ with a monomial h is given by*

$$g \wedge h \stackrel{def}{=} \Pi_{i \in Ind(g) \cap Ind(h)} x_i.$$

We will also need the following result that is only a slight generalization of [Mn07, Prop. 6, p.69] (and our proof will follow closely the proof of this proposition).

Lemma 4. *Let g be some monomial of degree $s \geqslant 1$. Denote by $\mathsf{supp}(g)$ the support of $\mathsf{ev}(g)$, then the minimum distance of $\left(\mathcal{S}_{\mathsf{supp}(g)}(\mathscr{C}(I))\right)^{\perp}$ is greater than or equal to 2^{r_-}. If the minimum distance is equal to 2^{r_-} then there exists a monomial h in $\mathcal{M} \setminus \check{I}$ such that*

(i) the number of variables of $h \wedge g$ is $s - 1$,
(ii) the number of variables of $h \wedge \check{g}$ is $m - r_- - s$.

Proof. Let us take a nonzero codeword of $\mathscr{C}(I)^{\perp}$, say that is the evaluation of some polynomial f, which is in this case of degree at most $m - 1 - r_-$. Write $f = \sum_j m_j$ as a sum of monomials. Then $\tilde{f} \stackrel{\text{def}}{=} \sum_{j : g \nmid m_j} m_j$ is defined as the polynomial where we have removed from the monomial expression of f all monomials that are divisible by g. Since $(\mathcal{S}_{\mathsf{supp}(g)}(\mathscr{C}(I)))^{\perp} = \mathcal{P}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)^{\perp}\right)$, we want to prove that the evaluation of f on $\{0,1\}^m \setminus \mathsf{supp}(g)$ is either zero or of weight $\geqslant 2^{r_-}$. Notice that the evaluation on $\{0,1\}^m \setminus \mathsf{supp}(g)$ coincides with the evaluation of $\tilde{f}$.

Let us assume that $g = x_0 \dots x_{s-1}$. With this choice, let us pick a monomial of $\tilde{f}$ that has maximum degree in $x_s, \dots, x_{m-1}$. Let d be this degree (in $x_s, \dots, x_{m-1}$). $\tilde{f}$ can be written as

$$\tilde{f} = mu(x_0, \dots, x_{s-1}) + v(x_0, \dots, x_{m-1}),$$

where m is a monomial of degree d in $x_s, \dots, x_{m-1}$. We take here in the monomials whose sum is equal to $\tilde{f}$ all monomials that are divisible by m and u is just the sum of these monomials divided by m. Let d' be the degree of u which is necessarily smaller than s since $\tilde{f}$ does not contain any monomial divisible by g.

Notice that $u(x_0 \dots x_{s-1})$ is non zero in at least $2^{s-d'} - 1$ entries if we do not count the $(1, \dots, 1)$ entry, since its evaluation is a codeword of $\mathscr{R}(d', s)$.

Call a "block" the set of points $(x_0, \dots, x_{m-1})$ which take a prescribed value on $x_0, \dots, x_{s-1}$. The support $\mathsf{supp}(g)$ of g corresponds to the block $x_0 = 1, \dots, x_{s-1} = 1$. Notice that the weight of $\mathsf{ev}(f)$ restricted to a block (with the exception of the block $x_0 = 1, \dots, x_{s-1} = 1$) is at least 2^{m-s-d}, since this restriction is a codeword of $\mathscr{R}(d, m-s)$. In other words the weight of $\mathsf{ev}(\tilde{f}(1+g))$ is lower-bounded by

$$|\mathsf{ev}(\tilde{f})(1+g)| \geqslant 2^{m-s-d}(2^{s-d'} - 1) \geqslant 2^{m-s-d} 2^{s-d'} \frac{1}{2} = 2^{m-d-d'-1}.$$

Notice that we have $d + d' \leqslant m - r_- - 1$ and therefore we finally obtain

$$|\mathsf{ev}(\tilde{f})| \geqslant 2^{m-(m-r_- -1)-1} = 2^{r_-}.$$

This proves the statement about the minimum distance in this case. A quick inspection of this proof shows that the only fact we used on g was that is is different from 1 (the particular form of g was only here to simplify notation), and therefore it also holds for all monomials g different from 1.

Assume now that the minimum distance of $\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^{\perp}$ is equal to 2^{r_-}. By a quick inspection of this proof this means that $\deg u = s - 1$ and $\deg m = m - r_- - 1 - (s-1) = m - r_- - s$. Write u as a set of monomials $u = \sum_j m'_j$ and choose m' as any monomial in this sum that is of degree $s - 1$. Obviously $h \stackrel{\text{def}}{=} mm'$ is a monomial of degree $s - 1 + m - r_- - s = m - r_- - 1$ that appears as a monomial in the sum $f = \sum_j m_j$. Therefore h is in $\mathcal{M} \setminus \check{I}$. Such an h has the aforementioned form.

We will now use this to prove Theorem 4. We recall its statement below.

Theorem. *Let $g = x_{i_1} \dots x_{i_{r_+}}$ be a monomial of degree r_+ in I. Denote by $\mathsf{supp}(g)$ the support of $\mathsf{ev}(g)$, then the minimum distance of $\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^{\perp}$ is equal to 2^{r_-} if and only if there exists a monomial h in $\mathcal{M} \setminus \check{I}$ such that:*

(i) the number of variables of h that are also variables of g is $r_+ - 1$,
(ii) the number of variables of h that are also variables of $\check{g}$ is $m - r_- - r_+$.

Proof. First of all let us notice that the minimum distance of $E\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^{\perp}$ is the same as the minimum distance of $\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^{\perp}$. From Lemma 3 we know that any codeword in the first code can be written as $\mathsf{ev}((1+g)f))$ where f is polynomial which is a linear combination of monomials in $\mathcal{M} \setminus \check{I}$. Consider now that there is a monomial h satisfying the conditions above. Let us prove that the weight of $\mathsf{ev}((1+g)h)$ is equal to 2^{r_-}. Let i_0 be the only index that is in $\mathrm{Ind}(g)$ but not in $\mathrm{Ind}(g \wedge h)$. Observe now that

$$\begin{aligned}(1+g)h &= (1 + x_{i_1} \dots x_{i_{r_+}}) \prod_{i \in \mathrm{Ind} g \wedge h} x_i \prod_{i \in \mathrm{Ind}(\check{g} \wedge h)} x_i \\ &= (1 + x_{i_0}) \prod_{i \in \mathrm{Ind} g \wedge h} x_i \prod_{i \in \mathrm{Ind}(\check{g} \wedge h)} x_i \\ &= (1 + x_{i_0})h.\end{aligned}$$

Thus

$$|\mathsf{ev}((1+g)h))| = |(\mathsf{ev}((1+x_{j_0})h)| = 2^{m-(m-r_- -1+1)} = 2^{r_-}.$$

By using the lower-bound on the minimum distance coming from Lemma 4 we obtain that the minimum distance of $\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^{\perp}$ is equal to 2^{r_-}.

Assume now that the minimum distance of $\left(\mathcal{S}_{\mathsf{supp}(g)}\left(\mathscr{C}(I)\right)\right)^{\perp}$ is equal to 2^{r_-}, then we can use Lemma 4 and obtain the aforementioned claim.

B.2 Proof of Proposition 4

Proposition. *W^{π}_{min} is invariant by the action of $\pi^{-1}\mathbb{LTA}_m\pi$ and if Σ is a signature for W_{min} under the action of $\mathbb{LTA}_m$, then it is also a signature for the action of $\pi^{-1}\mathbb{LTA}_m\pi$ on W^{π}_{min}.*

Proof. The invariance of $W^{\pi}_{\min}$ follows from the fact that (i) $\mathbb{LTA}_m$ is a subgroup of the permutation group of $\mathscr{C}(I)$ by Theorem 1 and (ii) this implies that $\pi^{-1}\mathbb{LTA}_m\pi$ is a subgroup of the permutation group of $\mathscr{C}(I)^{\pi}$ by Proposition 1. For the second part, it suffices to prove that Σ takes different values on the orbits of $W^{\pi}_{\min}$ under the action of $\pi^{-1}\mathbb{LTA}_m\pi$. Consider two elements $\boldsymbol{x}^{\pi}$ and $\boldsymbol{y}^{\pi}$ that belong to two different orbits. They are the permuted versions of $\boldsymbol{x}$ and $\boldsymbol{y}$ which belong to different orbits of $W_{\min}$. If this were not the case we would have $\boldsymbol{x} = \boldsymbol{y}^{\gamma}$ for γ in $\mathbb{LTA}_m$. However this would imply that $\boldsymbol{x}^{\pi} = \boldsymbol{y}^{\gamma\pi} = \boldsymbol{y}^{\pi\pi^{-1}\gamma\pi} = (\boldsymbol{y}^{\pi})^{\pi^{-1}\gamma\pi}$ and this would imply that $\boldsymbol{x}^{\pi}$ and $\boldsymbol{y}^{\pi}$ would be in the same orbit under the action of $\pi^{-1}\mathbb{LTA}_m\pi$. We finish the proof by observing that

$$\Sigma(\boldsymbol{x}^{\pi}, \mathscr{C}(I)^{\pi}) = \Sigma(\boldsymbol{x}, \mathscr{C}(I))$$
$$\Sigma(\boldsymbol{y}^{\pi}, \mathscr{C}(I)^{\pi}) = \Sigma(\boldsymbol{y}, \mathscr{C}(I))$$

Therefore $\Sigma(\boldsymbol{x}^{\pi}, \mathscr{C}(I)^{\pi})$ and $\Sigma(\boldsymbol{y}^{\pi}, \mathscr{C}(I)^{\pi})$ are different since $\Sigma(\boldsymbol{x}, \mathscr{C}(I))$ and $\Sigma(\boldsymbol{y}, \mathscr{C}(I))$ are different.

B.3 Proof of Proposition 5

Proposition. *The orbit of $\boldsymbol{c}_{min}$ under $\mathbb{LTA}_m$ consists of 2^{r_+} codewords that are of the form $\boldsymbol{c}_{min}(\varepsilon_0, \ldots, \varepsilon_{r_+-1})$ where the ε_i's are arbitrary element of $\mathbb{F}_2$. The orbit of $\boldsymbol{c}^{\pi}_{min}$ under $\pi^{-1}\mathbb{LTA}_m\pi$ is given by 2^{r_+} codewords of weight 2^{m-r_+} that have disjoint supports which are the permuted versions $A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})^{\pi}$ of the affine spaces $A(\varepsilon_0, \ldots, \varepsilon_{r_+-1})$.*

Proof. Let f be the monomial $x_0 \ldots x_{r_+-1}$ (i.e. $\boldsymbol{c}_{\min} = \mathsf{ev}(f)$). Under the action of π in $\mathbb{LTA}_m$ this monomial is transformed into $x'_0 \ldots x'_{r_+-1}$ where $x'_i = \varepsilon_i + x_i + \sum_{j<i} a_{ij}x_j$ where the ε_i's and the a_{ij}'s are binary. The support of such a monomial is given by the affine space $x'_0 = 1, \ldots, x'_{r_+-1} = 1$, but this is readily seen to be an affine space of the form $x_0 = \varepsilon'_0, \ldots, x_{r_+-1} = \varepsilon'_{r_+-1}$ where the ε'_i's are binary. This implies the first claim. The claim on the orbit of $\boldsymbol{c}^{\pi}_{\min}$ follows from the fact that for any $\gamma \in \mathbb{LTA}_m$ we have

$$(\boldsymbol{c}^{\pi}_{\min})^{\pi^{-1}\gamma\pi} = (\boldsymbol{c}^{\gamma}_{\min})^{\pi}.$$