A Distinguisher for High Rate McEliece Cryptosystem – Extended Abstract –

Jean-Charles Faugère[1], Ayoub Otmani[2,3], Ludovic Perret[1], and Jean-Pierre Tillich[2]

[1] SALSA Project - INRIA (Centre Paris-Rocquencourt)
UPMC, Univ Paris 06 - CNRS, UMR 7606, LIP6
104, avenue du Président Kennedy 75016 Paris, France
jean-charles.faugere@inria.fr, ludovic.perret@lip6.fr
[2] SECRET Project - INRIA Rocquencourt
Domaine de Voluceau, B.P. 105 78153 Le Chesnay Cedex - France
ayoub.otmani@inria.fr, jean-pierre.tillich@inria.fr
[3] GREYC - Université de Caen - Ensicaen
Boulevard Maréchal Juin, 14050 Caen Cedex, France.

Abstract. The purpose of this talk is to study the difficulty of the Goppa Code Distinguishing (GD) problem, which is the problem of distinguishing the public matrix in the McEliece cryptosystem from a random matrix. It is widely believed that this problem is computationally hard as proved by the increasing number of papers using this hardness assumption. One can consider that disproving/mitigating this hardness assumption is a breakthrough in code-based cryptography. In this paper, we present an efficient distinguisher for alternant and Goppa codes over binary/non binary fields. Our distinguisher is based on a recent algebraic attack against compact variants McEliece which reduces the key-recovery to the problem of solving an algebraic system of equations. We exploit a defect of rank in the (linear) system obtained by linearizing this algebraic system. It turns out that our distinguisher is also highly discriminant. Indeed, we are able to precisely quantify the defect of rank for "generic" binary and non-binary random, alternant and Goppa codes. We have verified these formulas with practical experiments, and a theoretical explanation for such defect of rank is also provided. To our knowledge, this is the first serious cryptographic weakness observed on McEliece since thirty years.

Keywords : public-key cryptography, McEliece cryptosystem, algebraic cryptanalysis, distinguisher.

1 Introduction

Code-based public key cryptography appeared with McEliece's pioneering work [17] where the author proposed one-way trapdoor functions based on irreducible binary Goppa codes. The class of Goppa codes represents one of the most important example of linear codes having an efficient decoding algorithm [3, 21]. A binary Goppa code requires in order to be defined a polynomial $g(z)$ of degree $r \geq 1$ with coefficients in some extension $\mathbb{F}_{2^m}$ of degree $m > 1$ over $\mathbb{F}_2$, and a n-tuple $\mathscr{L} = (x_1, \ldots, x_n)$ of distinct elements in $\mathbb{F}_{2^m}$ with $n \leq 2^m$. The trapdoor of the McEliece public-key scheme consists of the randomly picked $g(z)$ with $\mathscr{L}$ which together provide all the information to decode efficiently. The public key is a randomly picked generator matrix of the chosen Goppa code. A ciphertext is obtained by multiplying a plaintext with the public generator matrix and adding a random error vector of prescribed Hamming weight. The receiver decrypts the message thanks to the decoding algorithm that can be derived from the secrets. Niederreiter [19] brings a significant modification of the McEliece cryptosystem by proposing to describe public linear codes through parity-check matrices. The resulting public key cryptosystem is as as secure as the McEliece's one. The first code-based signature scheme came out in [8] almost twenty years after the McEliece's

proposal. The only difference between the encryption and the signature scheme rests on the choice of the parameters of the binary Goppa codes. In order to be efficient, Goppa codes for the signature have to be chosen such that they correct very few errors. This leads to a very high rate $R = k/n$ with n is its length and k being the dimension of the code. It holds that $k = n - tm$ where by definition t is the number of errors and generally n is chosen to be equal to 2^m. For instance [13], a 80-bit security signature scheme imposes $t = 10$ and $m = 21$ which leads to $R = 0.99$.

All these cryptographic primitives base their security under two assumptions: the intractability of decoding random linear codes [2], and the difficulty of recovering the private key or an equivalent one. The problem of decoding an unstructured code is a long-standing problem whose most effective algorithms [14, 15, 23, 6, 4] have an exponential time complexity. Thus, one may reasonably not expect much progress in this direction. On the other hand no significant breakthrough has been observed during the last thirty years regarding the problem of recovering the private key. Indeed, although some weak keys have been identified in [16], the only known key-recovery attack is the exhaustive search of the secret polynomial $\Gamma(z)$ of the Goppa code, and applying the *Support Splitting Algorithm* (SSA) [22] to check whether the Goppa code candidate is *permutation-equivalent* to the code defined by the public generator matrix. Despite the fact that there still does not exist a practical attack against the McEliece's proposal of using binary Goppa codes, one should not exclude the possibility of breakthrough in that field. The authors of [8] alleviated the McEliece assumptions by introducing the *Goppa Code Distinguishing (GD) problem*. It assumes that there exists no polynomial time algorithm that distinguishes a generator matrix of a Goppa code from a random generator matrix which is a classical belief in code-based cryptography. For instance, according to the authors of [8], 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*". Currently, the only known algorithm that solves GD problem is based on the enumeration of Goppa codes and the SSA algorithm [22], as explained below. The time complexity of this method is $\mathcal{O}(2^{mt})$ if one assumes that the cost of the SSA algorithm is negligible (which is a reasonable assumption for Goppa codes, but not for all linear codes).

As a consequence, it is widely believed that distinguishing the public matrix in McEliece from a random matrix is computationally hard. Furthermore, the hardness of the Goppa Code Distinguishing (GD) problem is mandatory to prove the semantic and CCA2 security of McEliece in the random oracle model and in the standard model [20, 11, 5], the security in the random oracle model against existential forgery [8, 9] of the CFS signature [8] scheme, the provable security of several primitives such as a threshold ring signatures scheme [10], an identity-based identification scheme [7], which are build upon CFS. Therefore, showing that the Goppa Code Distinguishing problem is easier than expected will "unprove" most of the provable primitives based on McEliece, and more importantly will be the first serious cryptographic weakness observed on this scheme since thirty years. The purpose of this paper is to study the difficulty of the Goppa Code Distinguishing (GD) problem:

Definition 1 (Goppa Code Distinguishing (GD) Problem). *Let n and k be two integers such that $k \leq n$. We denote by $\mathsf{Goppa}(n,k)$ the set of $k \times n$ generator matrices of Goppa codes. Similarly, $\mathsf{Random}(n,k)$ is the set of $k \times n$ random generator matrices. A* distinguisher *$\mathcal{D}$ is an algorithm that takes as input a matrix $\mathbf{G}$ and returns a bit. We say that $\mathcal{D}$ solves the GD problem if it wins the following game:*

– $b \xleftarrow{R} \{0,1\}$ *If* $b = 0$ *then* $\mathbf{G} \xleftarrow{R} \mathsf{Goppa}(n,k)$ *otherwise* $\mathbf{G} \xleftarrow{R} \mathsf{Random}(n,k)$
– *If* $\mathscr{D}(\mathbf{G}) = b$ *then* $\mathscr{D}$ *wins the games else* $\mathscr{D}$ *loses.*

$\mathscr{D}$ *outputs* 1 *with probability* $\Pr[H \xleftarrow{R} \mathsf{Goppa}(n,k) : \mathscr{D}(H) = 1]$ *if* H *is a random binary parity check matrix of a Goppa code* $\mathsf{Goppa}(n,k)$ *and outputs* 1 *with probability* $\Pr[H \xleftarrow{R} \mathsf{Random}(n,k) : \mathscr{D}(H) = 1]$ *We call the advantage of a distinguisher* $\mathscr{D}$ *the following quantity:*

$$Adv^{GD}(\mathscr{D}) = \left|\Pr[H \xleftarrow{R} \mathsf{Goppa}(n,k) : \mathscr{D}(H) = 1] - \Pr[H \xleftarrow{R} \mathsf{Random}(n,k) : \mathscr{D}(H) = 1]\right|.$$

In this paper, we present a polynomial-time and deterministic distinguisher for solving the GD problem defined below with a maximal advantage. Along the way, we will also solve the code distinguishing problem for alternant codes. The key ingredient is to use a new algebraic technique introduced in [12] to attack two variants [1, 18] of McEliece. Namely, it has been observed [12] that a key recovery attack of these cryptosystems, as well as the genuine McEliece's system, can be reduced to solving the following algebraic set of equations:

$$\left\{ g_{i,1} Y_1 X_1^j + \cdots + g_{i,n} Y_n X_n^j = 0 \;\middle|\; i \in \{1,\ldots,k\}, j \in \{0,\ldots,r-1\} \right\} \tag{1}$$

where the unknowns are the X_i's and the Y_i's and the $g_{i,j}$'s are known coefficients (with $1 \leq i \leq k, 1 \leq j \leq n$) which are nothing but the coefficients (belonging to a certain field $\mathbb{F}_{2^s}$) of the public generator matrix of the scheme. Finally, k is equal to $n - mr$ here, where m is some divisor of s. In other words we have $2n$ unknowns and $rk = r(n - mr)$ polynomial equations. In the cases of [1, 18], additional structures permits to drastically reduce the number of variables and solve (1) efficiently using dedicated Gröbner bases techniques [12]. For the McEliece cryptosystem, solving (1) seems to be out of the scope of such dedicated techniques. However, we will see that the can use this algebraic approach to construct an efficient distinguisher.

Summary of the Results. In the first part of the talk, we will precisely explain how we can mount an algebraic attack against McEliece-like schemes, i.e. namely how the algebraic system (1) is constructed. Recall that solving (1) allows to recover the secret-key is McEliece-like schemes. The distinguisher that we will present is actually due to an unsuccessful attempt to solve the algebraic system (1) by linearization. Typically for the parameters proposed in CFS [8], there is a special method to linearize (1) yielding to a linear system with as many variables as equations. It turns out that the linearized system is not of full rank. Although this is an obstacle to break the system, this particular feature permits to construct an efficient distinguisher for alternant, binary and non-binary Goppa codes. Note that the distinguisher is efficient since we only have to compute the rank of a linear system. It turns out that our distinguisher is also highly discriminant. We will present explicit formulas on the dimension of the kernel of the linearized system for "generic" random, alternant, binary and non-binary Goppa codes. We performed extensive experiments to compare our theoretical results with practical results obtained on valid public matrices of McEliece. The results confirm that the generic formula obtained are accurate on concrete instances. As already pointed out, this is the first serious cryptographic weakness observed on Goppa codes since the initial proposal of McEliece 30 years ago. We emphasize that the Goppa Code Distinguishing problem has been widely considered as a hard problem in code-based cryptography as proved by the increasing number of papers using this assumption [20, 11, 5, 8–10, 7]. To our point of view, disproving/mitigating this hardness assumption is a breakthrough in code-based cryptography and may open a new direction to attack the McEliece cryptosystem. Finally, we will explain how the formulas have been obtained. To do so, we have used the combinatorial properties of the linearized system as well as the distinguishing features of Alternant/Goppa codes.

References

1. 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.
2. 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.
3. E. R. Berlekamp. Factoring polynomials over finite fields. In E. R. Berlekamp, editor, *Algebraic Coding Theory*, chapter 6. McGraw-Hill, 1968.
4. D. J. Bernstein, T. Lange, and C. Peters. Attacking and defending the McEliece cryptosystem. In *PQCrypto*, volume 5299 of *LNCS*, pages 31–46, 2008.
5. Bhaskar Biswas and Nicolas Sendrier. Mceliece cryptosystem implementation: Theory and practice. In *PQCrypto*, pages 47–62, 2008.
6. 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.
7. Pierre-Louis Cayrel, Philippe Gaborit, David Galindo, and Marc Girault. Improved identity-based identification using correcting codes. *CoRR*, abs/0903.0069, 2009.
8. 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.
9. Léonard Dallot. Towards a concrete security proof of courtois, finiasz and sendrier signature scheme. In *WEWoRC*, pages 65–77, 2007.
10. Léonard Dallot and Damien Vergnaud. Provably secure code-based threshold ring signatures. In *IMA Int. Conf.*, pages 222–235, 2009.
11. Rafael Dowsley, Jörn Müller-Quade, and Anderson 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.
12. Jean-Charles Faugère, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. In *Proceedings of Eurocrypt 2010*. Springer Verlag, 2010. to appear.
13. 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.
14. 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.
15. 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.
16. P. Loidreau and N. Sendrier. Weak keys in the mceliece public-key cryptosystem. *IEEE Transactions on Information Theory*, 47(3):1207–1211, 2001.
17. R. J. McEliece. *A Public-Key System Based on Algebraic Coding Theory*, pages 114–116. Jet Propulsion Lab, 1978. DSN Progress Report 44.
18. 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.
19. H. Niederreiter. A public-key cryptosystem based on shift register sequences. In *EUROCRYPT*, volume 219 of *LNCS*, pages 35–39, 1985.
20. Ryo Nojima, Hideki Imai, Kazukuni Kobara, and Kirill Morozov. Semantic security for the mceliece cryptosystem without random oracles. *Des. Codes Cryptography*, 49(1-3):289–305, 2008.
21. N. Patterson. The algebraic decoding of Goppa codes. *IEEE Transactions on Information Theory*, 21(2):203–207, 1975.
22. N. Sendrier. Finding the permutation between equivalent linear codes: The support splitting algorithm. *IEEE Transactions on Information Theory*, 46(4):1193–1203, 2000.
23. 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.