On the Security of Some Cryptosystems Based on Error-correcting Codes

Florent Chabaud [*]
Florent.Chabaud@ens.fr

Laboratoire d'Informatique de l'ENS [**]
45, rue d'Ulm
75230 Paris Cedex 05
FRANCE

Abstract. A certain number of public-key cryptosystems based on error-correcting codes have been proposed as an alternative to algorithms based on number theory. In this paper, we analyze algorithms that can be used to attack such cryptosystems in a very precise way, and optimize them. Thus, we obtain some more efficient attacks than those previously known. Even if they remain unfeasible, they indicate the cryptosystems parameters forbidden by the existence of these algorithms.

1 Introduction

1.1 An NP-complete problem

It is known [BMT78] that the problem of finding a codeword of given weight in a linear binary code is NP-complete. This property can be used to build cryptosystems or identification systems. But, as for other NP-complete problems, some cases of this problem can be solved by probabilistic algorithms. This means that cryptographic systems such as the following ones must take into account the performances of these algorithms.

1.2 The McEliece public key cryptosystem

Presentation This cryptosystem is one of the first that has used error-correcting codes. Its purpose is public-key ciphering. Though Sidelnikov and Shestakov have shown that Generalized Reed-Solomon codes can not be used directly [SS92b], it is still not broken in its original description [McE78].

Principle We assume that we have a (n, k) linear code on $GF(2)$, described by its generator matrix G, for which we have a decoding algorithm that corrects at most t errors (the original description of McEliece uses Goppa codes [Ber73],

[*] On leave from Délégation Générale de l'Armement.

[**] Supported by the Centre National de Recherche Scientifique URA 1327.

but the larger class of alternant codes [MS83] can also be used). We now choose at random some invertible matrix S and a permutation matrix P. The triplet (S, G, P) will be the secret key and $(t, \hat{G} = SGP)$ will be the public key of the cryptosystem. Then, to transmit a k-bits message m, the sender uses the ciphered message $c = m\hat{G} \oplus e$ with e a random error of weight at most t. To decipher, the recipient decodes $cP^{-1} = mSG \oplus eP^{-1}$ with his decoding algorithm.

Security The security of this cryptosystem is based on the secrecy of e, and on the fact that the decoding algorithm can only be applied if one knows a canonical form of the generator matrix G. But, if we consider the public matrix

$$G' = \left(\begin{array}{c} \hat{G} \\ \hline m\hat{G} + e \end{array} \right)$$

it is the matrix of a linear code of which a minimum weight codeword is e. Hence, an algorithm that can find the shortest word of a linear code is an attack of the McEliece cryptosystem.

1.3 The J. Stern public key identification scheme

J. Stern has presented at Crypto'93 a new public-key authentication scheme [Ste93] that uses property 1.1. In this scheme, the public key is an (n, k) code parity check matrix H, and each sender receives a secret key consisting in a n-bits word s of weight t. The public key is the decoding syndrome Hs and the sender has to prove that his secret key is of weight t, which can be done without revealing s thanks to the described protocol. As for McEliece cryptosystem, if we have an algorithm that finds codewords of weight $t + 1$ in a code, we can apply it to the matrix $H' = (H \quad | \quad Hs)$ until it finds a codeword $\bar{s}$ with last bit at one. The first n bits of $\bar{s}$ then consist of a signature s' for the public signature Hs.

1.4 Trying to hash with error-correcting codes

An implicit idea in [Ste93] is to use error-correcting codes in a hashing way. Given the parity check matrix H of a (n, k) linear code chosen at random, then the minimum weight w of this code can be evaluated by $\binom{n}{w} \gtrsim 2^{n-k}$. Let us assume that we have a set M of messages to hash such that $|M| = \binom{n}{m}$ with $m > w$. Then we can identify each $m \in M$ with a n-bits word of weight m. Using the property 1.1 we can hash such a message by its $n - k$-bits syndrome Hm.

2 The attack algorithms

2.1 Algorithm of J.S. Leon

Principle This algorithm [Leo88] can be used for any linear code. It uses the generator matrix to find short codewords through a probabilistic method. The decription below is slightly different from those of [Leo88], but it is more simple to implement, and it runs a little faster than the original description. Let us assume that the parameters of this algorithm are p and s, then one iteration of the algorithm is as follows:

1. Permute the columns of the generator matrix randomly.
2. Apply a gaussian elimination on the rows of the matrix to obtain this form:

$$G = (\, I_k \;\mid\; Z \;\mid\; B\,),$$

with Z a $(s-k) \times k$ matrix.

3. Search combinations of at most p rows of Z that lead to codewords of the restricted code $(\, I \;\mid\; Z\,)$ of weight less than p.
4. For these codewords, compute the whole word and check its weight.

Work factor We assume that a codeword c of weight w exists. Then the probability π that a permutation of the columns leads to a favorable configuration is

$$\pi_{L(p,s)}(n,k,w) = \sum_{i=1}^{p} \frac{\binom{n-w}{s-i}\binom{w}{i}}{\binom{n}{s}}.$$

For each iteration, an estimation of the number of operations is:

1. $k \times k/2 \times n$ for the gaussian elimination.
2. $\sum_{i=1}^{p} \binom{k}{i}(i-1)$ additions of $(s-k)$-bits words.
3. the average number of cases such that the whole computation is needed is

$$\sum_{i=1}^{p} \binom{k}{i} \times \frac{\sum_{j=0}^{p-i} \binom{s-k}{j}}{2^{s-k}}.$$

Consequently, an estimation of the work factor for this algorithm is

$$W_{L(p,s)}(n,k,w) = \frac{k \times k/2 \times n + \sum_{i=1}^{p} \binom{k}{i}(i-1)\left[(s-k) + n\frac{\sum_{j=0}^{p-i}\binom{s-k}{j}}{2^{s-k}}\right]}{\pi_{L(p,s)}(n,k,w)}.$$

2.2 Algorithm of P.J. Lee and E.F. Brickell

Principle This algorithm [LB89] is the best known attack against McEliece cryptosystem [McE78]. Its principle is the same as for J.S Leon's algorithm, but it takes advantage from the fact that the shortest codeword is brought by the ciphered text. Hence, one iteration of the algorithm is as follows:

1. Permute the columns of the generator matrix randomly.
2. Apply a gaussian elimination on the rows of the matrix to obtain thes form $G = (\, I_k \mid A\,)$, with the corresponding permuted cipher text $c = (\, c_1 \oplus e_1 \mid c_2 \oplus e_2\,)$.
3. Guess that the error e_1 is of weight at most p and checks whether the error $e = (\, e_1 \mid e_2\,)$ is of weight w.

Work factor In this case, we are sure that the error e of weight w exists. The probability π that a permutation of the columns leads to a favorable configuration is

$$\pi_{Le(p)}(n,k,w) = \sum_{i=0}^{p} \frac{\binom{n-w}{k-i}\binom{w}{i}}{\binom{n}{k}}.$$

For each iteration, an estimation of the number of operations is:

1. $k \times k/2 \times n$ for the gaussian elimination.
2. About $k/2 + \sum_{i=1}^{p} \binom{k}{i} i$ additions on the $(n-k)$-bits words of A.

Consequently, an estimation of the work factor for this algorithm is

$$W_{Le(p)}(n,k,w) = \frac{k \times k/2 \times n + (n-k)\left[k/2 + \sum_{i=1}^{p} \binom{k}{i} i\right]}{\pi_{Le(p)}(n,k,w)}.$$

2.3 Principle

This probabilistic algorithm [Ste89] can be used for any linear code. It uses the parity check control matrix. As for the above algorithms, we first reduce the problem with a gaussian elimination after a random permutation of the columns of the check control matrix:

$$H = \left(\begin{array}{c|c} I_{n-k} & \begin{array}{c} Z \\ \hline B \end{array} \end{array} \right)$$

with Z a $(\ell \times k)$ matrix.

The idea of the algorithm is to consider Z as a new parity check control matrix. If we find a short codeword of Z then the corresponding complete codeword of H is a good candidate to be short.

How to find codewords of Z The original algorithm [Ste89] randomly splits the column of Z in two sets X and Y. First, it computes all the linear combinations of p columns of X and stores them. Second, it computes all the linear combinations of p columns of Y and checks the collisions with the stored values. This allows to build codewords of Z of weight $2p$.

Work factor In fact it is more simple and more efficient to consider X and Y as the left half and right half of the matrix Z:

$$H = \left(I_{n-k} \;\middle|\; \begin{array}{c|c} X & Y \\ \hline \multicolumn{2}{c}{B} \end{array} \right).$$

Then the probability that an iteration is successful is slightly different than the one given in [Ste89]:

$$\pi_{S(p,\ell)}(n,k,w) = \frac{\binom{w}{p}\binom{n-w}{k/2-p}\binom{w-p}{p}\binom{n-w-(k/2-p)}{k/2-p}\binom{n-w-(k-2p)}{(n-k-\ell)-(w-2p)}}{\binom{n}{k/2}\binom{n-k/2}{k/2}\binom{n-k}{n-k-\ell}}.$$

For each iteration, an estimation of the number of operations is:

1. $(n-k) \times (n-k)/2 \times n$ for the gaussian elimination.
2. $2 \times \binom{k/2}{p}(p-1)$ additions of ℓ-bits words.
3. Each ℓ-bits value of X-vectors linear combination is reached about $\frac{\binom{k/2}{p}}{2^\ell}$ times. Hence, the total number of collisions is about $\frac{\binom{k/2}{p}^2}{2^\ell}$. Besides, for one collision, one has to compute $2p-1$ additions of $(n-k-\ell)$-bits words.

Consequently, an estimation of the work factor for this algorithm is

$$W_{S(p,\ell)}(n,k,w) = \frac{(n-k) \times (n-k)/2 \times n + 2\ell \times \binom{k/2}{p}(p-1) + (n-k-\ell)(2p-1)\frac{\binom{k/2}{p}^2}{2^\ell}}{\pi_{S(p,\ell)}(n,k,w)}.$$

2.4 A variant of J. Stern algorithm

How to find codewords of Z The problem with the above algorithm [Ste89] is that it needs great amounts of memory for large dimensions. Besides, the memory access isn't instantaneous. Hence the results in table 4 must be taken with care. But keeping the same principle, one can find short codewords of Z by considering linear combinations of at most p columns of Z. More generally, if the syndrome of a $n-k$-bits word of weight p by Z is of low weight (less than s), then it is possible to build a codeword of H which is a good candidate to be short.

Work factor Let us assume that a codeword c of weight w exists. Then gaussian reduction yields the following form of H:

$$H = \left(\begin{array}{c|c|c} I_\ell & 0 & Z \\ \hline 0 & I_{k-\ell} & A \end{array}\right).$$

Then $c = (\, c_1 \ \mid \ c_2 \ \mid \ c_3\,)$ must be such that $w(c_1) \leq s$ and $1 \leq w(c_3) \leq p$. The probability π that a permutation of the columns leads to a favorable configuration is

$$\pi_{Sv(p,s,\ell)}(n,k,w) = \sum_{i=1}^{p} \sum_{j=0}^{s} \frac{\binom{n-w}{k-i}\binom{w}{i}\binom{w-i}{j}\binom{n-w-(k-i)}{l-j}}{\binom{n}{k}\binom{n-k}{l}}.$$

For each iteration, an estimation of the number of operations is:

1. $(n-k) \times (n-k)/2 \times n$ for the gaussian elimination.
2. $\sum_{i=1}^{p} \binom{k}{i}(i-1)$ additions of ℓ-bits words.
3. The average number of cases such that the whole computation is needed is

$$\sum_{i=1}^{p} \binom{k}{i} \times \frac{\sum_{j=0}^{s} \binom{\ell}{j}}{2^\ell}.$$

Consequently, an estimation of the work factor for this algorithm is

$$W_{Sv(p,s,\ell)}(n,k,w) = \frac{(n-k) \times (n-k)/2 \times n + \sum_{i=1}^{p} \binom{k}{i}(i-1) \left[\ell + (n-k-\ell)\frac{\sum_{j=0}^{s} \binom{\ell}{j}}{2^\ell}\right]}{\pi_{Sv(p,s,\ell)}(n,k,w)}.$$

3 "Factorization" of the attacks

One can remark that certain attacks may be slightly increased if the number of problems to solve is great. For instance, we can assume in the McEliece cryptosystem that we have several ciphered text to cryptanalyze, let us say N. Then it is possible, once the gaussian elimination has been performed, to achieve the remaining operations of Lee algorithm for the N ciphered text. Hence, the work factor to decrypt one of the N words turns in

$$W'_{Le(p)}(N,n,k,w) = \frac{k \times k/2 \times n + N(n-k)\left[k/2 + \sum_{i=1}^{p} \binom{k}{i} i\right]}{N\pi_{Le(p)}(n,k,w)}.$$

This means that cryptanalyzing one cipher text in McEliece cryptosystem requires more work than cryptanalyzing one cipher text among N.

4 Performances of the algorithm

4.1 Optimal parameters

A first attempt in optimizing some of these algorithms was given in [Cha93]. Unfortunately, this kind of asymptotic evaluation doesn't lead to practically good parameters. Using the preceding work factors, one can find the optimal parameters to solve a problem with one of these algorithms. Some examples are given in table 1. The first two codes can be broken. The third corresponds to J.Stern's authentication scheme specifications, and the last to the parameters proposed by Adams and Meier [AM88] for McEliece's cryptosystem. For this last example the results can be compared to those of Lee and Brickell' algorithm:

$$W_{Le(p=2)}(1024, 644, 38) = 2^{73.3},$$

which shows an improvement of a factor $2^{6.9}$ using the optimized Stern's attack. Note that if we use the remark of section 3, then

$$W'_{Le(p=1)}(1024, 644, 38, N) = 2^{68.4}$$

where the number of texts we use is $N \geq 2^{15}$.

	Leon (p, s)	Stern (p, ℓ)	Stern var. (p, ℓ, s)
(128,64,16)	$p = 3$ $s = 69$ $W = 2^{28.6}$	$p = 2$ $\ell = 8$ $W = 2^{28.1}$	$p = 2,\ s = 7$ $\ell = 20$ $W = 2^{28.1}$
(256,128,30)	$p = 3$ $s = 133$ $W = 2^{44.7}$	$p = 3$ $\ell = 18$ $W = 2^{42.3}$	$p = 3,\ s = 3$ $\ell = 13$ $W = 2^{43.9}$
(513,257,57)	$p = 3$ $s = 263$ $W = 2^{74.8}$	$p = 2$ $\ell = 12$ $W = 2^{72.3}$	$p = 3,\ s = 4$ $\ell = 17$ $W = 2^{73.9}$
(1024,645,38)	$p = 3$ $s = 653$ $W = 2^{71.7}$	$p = 2$ $\ell = 18$ $W = 2^{66.4}$	$p = 3,\ s = 1$ $\ell = 11$ $W = 2^{71.1}$

Table 1. Optimal parameters

4.2 What weights can be found ?

If we take as a limit the work factor 2^{45}, then it is possible to find very short codewords even in large codes.

On the other hand, the number $N_{(n,k)}(w)$ of codewords of weight w in a (n,k) random code can be estimated by the following formula:

$$N_{(n,k)}(w) \sim \frac{\binom{n}{w}}{2^{n-k}}.$$

For sufficient large values of w, this number grows enough to allow the algorithms to find at least one of the codewords. In fact the work factor can be divided by $N_{(n,k)}(w)$. Table 2 shows an estimation of the weights one can find using the described algorithms. These weights can be seen as the lower and upper bound of the non-accessible codewords area.

Code	Algo.	(256,128)	(512,256)	(1024,645)	(1024,512)
Upper bound	Leon	30	29	18	26
	Stern var.		29	19	26
	Stern		31	22	29
Estimated minimum weight		30	58	75	115
Lower bound	Stern	30	77	125	184
	Stern var.		77	128	186
	Leon		78	129	186

Table 2. Accessible weights

4.3 Consequences on the security of cryptographic systems

These results prove that use of codes of large dimensions is not enough to ensure security for cryptosystems based on error-correcting codes. For instance, in the McEliece cryptosystem, the weight of the random error cannot be less than 22. Besides, these results forbid the use of error-correcting codes as hash-functions in the way described in part 1.4. In fact, it is possible to build couples of words of given low weight w that have the same syndrome. For instance if we try to use a $(1024, 512, \sim 115)$ random linear code, and use syndrome of words of weight $w = 130$ to hash messages (see 1.4), then the algorithms cannot find a message given a syndrome, but they can find word of weight about 190. Let us assume we have a codeword c of weight 190, we can split it in two words of weight $190/2 = 95$ c_1 and c_2. Then we select $130 - 95 = 35$ bits among the $1024 - 130 = 894$ zero-bits of c. This gave us a word c_3 of weight 35. We have

$$Hc = 0 = Hc_1 \oplus Hc_2.$$

Hence, $c_1 \oplus c_3$ and $c_2 \oplus c_3$ are of weight 130 and

$$H(c_1 \oplus c_3) = H(c_2 \oplus c_3).$$

5 Conclusion

We have estimated and improved existing attacks against cryptosystems based on the NP-complete problem of finding codewords of given weight in a linear error-correcting codes. These attacks don't succeed in breaking McEliece's cryptosystem or Stern's authentication scheme, but they forbid the use of some parameters.

Acknowledgement

I wish to thank JACQUES STERN who suggested this work, and Pascal Veron for fruitful discussions.

References

[AM88] C. Adams and H. Meijer. Security-related comments regarding McEliece's public-key cryptosystem. In *Lecture Notes in Computer Science, Advances in Cryptology – CRYPTO '87*, pages 221–228. Springer-Verlag, 1988.

[Ber73] E.R. Berlekamp. Goppa codes. *IEEE Trans. Inform. Theory*, IT-19(5):590–592, September 1973.

[BMT78] E.R. Berlekamp, R.J. McEliece, and H.C.A. Van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. Inform. Theory*, IT-24(3):384–386, May 1978.

[Cha93] F. Chabaud. Asymptotic analysis of probabilistic algorithms for finding short codewords. In *CISM Courses and Lectures – EUROCODE '92*, volume 339, pages 175–183. Springer-Verlag, 1993.

[LB89] P.J. Lee and E.F. Brickell. An observation on the security of McEliece's public-key cryptosystem. In *Lecture Notes in Computer Science, Advances in Cryptology – EUROCRYPT '88*, pages 275–280. Springer-Verlag, 1989.

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

[McE78] R.J. McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN progress report 42-44*, pages 114–116, 1978.

[MS83] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-correcting Codes*. North-Holland, 1983.

[SS92a] V.M. Sidelnikov and S.O. Shestakov. On cryptosystems based on generalized Reed-Solomon codes. *Diskretnaya Math*, 4:57–63, 1992. in Russian.

[SS92b] V.M. Sidelnikov and S.O. Shestakov. On unsecurity of cryptosystems based on generalized Reed-Solomon codes. manuscript, 1992.

[Ste89] J. Stern. A method for finding codewords of small weight. In *Lecture Notes in Computer Science, Coding Theory and Applications*, volume 388, pages 106–113. Springer, 1989. G. Cohen and J. Wolfmann editors.

[Ste93] J. Stern. A new identification scheme based on syndrome decoding. In *proceedings of CRYPTO '93*, 1993. to appear.