Cryptanalysis of the Original McEliece Cryptosystem

Anne Canteaut and Nicolas Sendrier

INRIA - projet CODES
BP 105
78153 Le Chesnay, France

Abstract. The class of public-key cryptosystems based on error-correcting codes is one of the few alternatives to the common algorithms based on number theory. We here present an attack against these systems which actually consists of a new probabilistic algorithm for finding minimum-weight words in any large linear code. This new attack notably points out that McEliece cipher with its original parameters does not provide a sufficient security level.

1 Introduction

Since the concept of public-key cryptography appeared in 1977, searching for secure public-key cryptosystems and identification schemes has been one of the most active areas in the field of cryptology. Many public-key ciphers emerged just after the invention of RSA and their underlying problems were as varied as computing a discrete logarithm, solving a knapsack problem, inverting some polynomial equations over a finite field.... But the development of some cryptanalysis methods have finally made most of them insecure. Twenty years after the fundamental paper of Diffie and Hellman, public-key cryptography has therefore become dangerously dependent on only two problems: integer factoring and discrete logarithm. However the class of public-key ciphers and identification schemes based on error-correcting codes still resists cryptanalysis. It relies on the hardness of decoding or equivalently of finding a minimum-weight codeword in a large linear code with no visible structure. The most famous of these systems are McEliece and Niederreiter ciphers [McE78,Nie86] — which are equivalent from the security point of view — and the identification schemes proposed by Stern [Ste89] and Véron [Vér95]. They are at the moment one of the few alternatives to the common public-key algorithms based on number theory. Studying their security seems therefore essential in order to anticipate a possible important progress in factoring methods for example. Moreover these public-key ciphers are particularly interesting since they run much faster than any algorithm relying on number theory.

In this paper we present an attack on these cryptosystems which consists of a new probabilistic algorithm for finding minimum-weight codewords in any linear code. We first briefly present in Section 2 some public-key cryptosystems based

K. Ohta and D. Pei (Eds.): ASIACRYPT'98, LNCS 1514, pp. 187–199, 2000.

on error-correcting codes. Section 3 then describes a new algorithm for finding minimum-weight words in any linear code. Using Markov chain theory we show in Section 4 how to compute the number of elementary operations it requires. In Section 5 we finally use these results to evaluate the security of these public-key cryptosystems. We notably prove that the parameters which were originally proposed by McEliece for his cryptosystem make it insecure.

2 Some Cryptosystems Based on Error-Correcting Codes

The class of public-key cryptosystems based on the hardness of decoding or of finding a minimum-weight word in a large code contains both McEliece and Niederreiter ciphers and some zero-knowledge identification schemes like the one proposed by Stern.

2.1 McEliece and Niederreiter Public-Key Ciphers

McEliece cryptosystem uses as a secret key a linear binary code chosen in a family Γ of $[n, k]$-linear codes with error-correcting capability t for which an efficient decoding algorithm is known. In his original paper [McE78], McEliece proposed to choose this secret code amongst the irreducible binary Goppa codes of length 1024, dimension 524 and minimum distance 101.

- **private key:** it is composed of an $[n, k]$-linear binary code $\mathcal{C}$ chosen in the family Γ, a random $k \times k$ binary invertible matrix S and a random $n \times n$ permutation matrix P.
- **public key:** it consists of the $k \times n$ matrix G' defined by $G' = SGP$ where G is a generator matrix of the secret code $\mathcal{C}$.
- **encryption:** the ciphertext corresponding to the k-bit message m is $x = mG' + e$, where e is a random n-bit error-vector of weight t.
- **decryption:** the decryption procedure consists in computing $xP^{-1} = mSG + eP^{-1}$ and using a fast decoding algorithm for $\mathcal{C}$ to recover mS. The message is then given by $m = (mS)S^{-1}$.

By definition the public key is therefore a generator matrix for an other linear code $\mathcal{C}'$ which is equivalent to $\mathcal{C}$. A ciphertext in McEliece cryptosystem then corresponds to a word of the public code $\mathcal{C}'$ with t corrupted positions.

Niederreiter proposed a dual version of this system [Nie86] where the public-key is a parity-check matrix H' of a code $\mathcal{C}'$ equivalent to the secret code. A plaintext m is here an n-bit vector of weight t and the associated ciphertext x corresponds to the syndrome of m relatively to the public code, $x = mH'^t$.

McEliece and Niederreiter cryptosystems are actually equivalent from the security point of view when set up for corresponding choices of parameters [LDW94]. But for given parameters Niederreiter cipher presents many advantages. First of all it allows a public key in systematic form at no cost for security whereas this

would reveal a part of the plaintext in McEliece system. The public key in Niederreiter system is then $(n-k)/n$ times smaller than in McEliece version. The systematic form of the public matrix H' and the low-weight of vector m significantly reduce the computational cost involved in the encryption in Niederreiter system. For [1024,524,101]-binary codes its transmission rate, *i.e.* the number of information symbols divided by the number of transmitted symbols, is smaller that in McEliece system. Another disadvantage of McEliece system is that it is easy to recover the plaintext if it has been encrypted twice with the same public-key. On the contrary Niederreiter cipher is deterministic since encrypting a given plaintext always leads to the same ciphertext.

Table 1 sums up the characteristics of these systems when they both use [1024,524,101]-binary codes. It then shows that it is preferable to use the version proposed by Niederreiter.

	McEliece [1024,524,101] binary code	Niederreiter [1024,524,101] binary code	RSA 1024-bit modulus public exponent = 17
public-key size	67,072 bytes	32,750 bytes	256 bytes
number of information bits transmitted per encryption	512	276	1024
transmission rate	51.17 %	56.81 %	100 %
number of binary operations performed by the encryption per information bit	514	50	2,402
number of binary operations performed by the decryption per information bit	5,140	7,863	738,112

Table 1. Performance of McEliece, Niederreiter and RSA public-key ciphers

We give for information the values corresponding to the RSA system with a 1024-bit modulus $n = pq$ when the public exponent is 17 — we here suppose that RSA encryption and decryption uses Karatsuba's method for large integer multiplication. These results point out that these public-key systems run much faster than RSA (about 50 times faster for encryption and 100 times faster for decryption). Their main disadvantages are the size of the public key and the lack of related signature scheme.

Cryptanalysis Methods There are mainly two guidelines to cryptanalyze McEliece cryptosystem :

- recover the original structure of the secret code from a generator (or parity-check) matrix of an equivalent code.
- decode the public code which has no visible structure.

The first class of attacks imposes some conditions on the family of secret codes Γ. For given length, dimension and minimal distance the family Γ must be large enough to avoid any enumeration. This aims at protecting the system from the attack which consists in enumerating all the elements of Γ until a code equivalent to the public code is found. This can be performed with an algorithm due to Sendrier [Sen96] which is able to determine from two generator matrices whether they correspond to equivalent codes and then to recover the permutation. A second condition is that a generator or parity-check matrix of a permutation equivalent code gives no information about the structure of the secret code, that means that the fast decoding algorithm requires some parameters of the secret code besides a generator matrix G'. This dismisses many families of codes like generalized Reed-Solomon codes [SS92] or concatenated codes [Sen94,Sen95].

But the family of irreducible Goppa codes is well-suited to such systems insofar as at present there exists no algorithm which is able to compute the characteristic parameters of a Goppa code from one of its permuted generator matrix. This class can even be extended to all [1024,524,101]-binary Goppa codes defined by a monic square-free polynomial of degree 50 in $GF(1024)[X]$ which has no root in $GF(1024)$. The cardinality of Γ is then $2^{498.5}$. In the case where the used family of codes satisfies the above properties, the equivalent code $\mathcal{C}'$ defined by the public key presents no visible structure; recovering a plaintext from the corresponding ciphertext then comes down to decoding any linear code.

2.2 Stern's Public-Key Identification Scheme

Stern presented at Crypto'93 [Ste93] a public-key identification scheme which relies on the hardness of finding a low-weight codeword of given syndrome. This scheme uses an $[n, k]$-random linear code over $GF(2)$. All users share a fixed parity-check matrix H for this code and an integer w slightly below the expected value for the minimal distance of a random linear code. Each user receives a secret key s which is an n-bit vector of weight w. His public key is then the syndrome sH^t. Any user can identify himself to another one by proving he knows s without revealing it thanks to an interactive zero-knowledge protocol. The minimal parameters proposed by Stern are $n = 512$, $k = 256$ and $w = 56$. Véron [Vér95] also proposed a dual version of this scheme similar to McEliece's original approach: it uses a generator matrix of the code instead of a parity-check matrix. He then suggested a new choice for the parameters in order to reduce the number of transmitted bits: $n = 512$, $k = 120$ and $w = 114$.

3 A new Algorithm for Finding Low-weight Codewords

Let $\mathcal{C}$ be a linear binary code of length n, dimension k and minimum distance d about which nothing is known but a generator matrix. We now develop an algorithm for finding a word of weight w in $\mathcal{C}$ where w is closed to d. This algorithm can also be used for decoding up to the correction capability $t = \lfloor \frac{d-1}{2} \rfloor$. If a message x is composed of a codeword corrupted by an error-vector e of weight

$w \leq t$, e can be recovered with this algorithm since it is the only minimum-weight word in the linear code $\mathcal{C} \oplus x$. Decoding an $[n, k]$-linear code then comes down to finding the minimum-weight codeword in an $[n, k+1]$-code.

Let $N = \{1, \cdots, n\}$ be the set of all coordinates. For any subset I of N, $G = (V, W)_I$ denotes the decomposition of matrix G onto I, that means $V = (G_i)_{i \in I}$ and $W = (G_j)_{j \in N \setminus I}$, where G_i is the ith column of matrix G. The restriction of an n-bit vector x to the coordinate subset I is denoted by $x_{|I} = (x_i)_{i \in I}$. As usual $\mathrm{wt}(x)$ is the Hamming weight of the binary word x.

Definition 1. *Let I be a k-element subset of N. I is an information set for the code $\mathcal{C}$ if and only if $G = (Id_k, Z)_I$ is a systematic generator matrix for $\mathcal{C}$.*

Our algorithm uses a probabilistic heuristic proposed by Stern [Ste89] which generalizes the well-known information set decoding method. But instead of exploring a set of randomly selected systematic generator matrices by performing at each iteration a Gaussian elimination on an $(n \times k)$-matrix as most algorithms do [LB88,Leo88], we choose at each step the new information set by modifying only one element of the previous one. This procedure is similar to the one used in the simplex method and it was first introduced in [Omu72]. If I is an information set and $G = (Id_k, Z)_I$ the corresponding systematic generator matrix, $I' = (I \setminus \{\lambda\}) \cup \{\mu\}$ is still an information set for the code if and only if the coefficient $Z_{\lambda,\mu}$ equals 1. In this case, the systematic generator matrix associated with I' is obtained from the previous one by a simple pivoting procedure which only requires $k(n-k)/2$ binary operations. Using this iterative method then leads to the following algorithm:

Initialization:

Randomly choose an information set I and apply a Gaussian elimination in order to obtain a systematic generator matrix $(Id_k, Z)_I$.

Until a codeword of weight w will be found:

1. Randomly split I in two subsets I_1 and I_2 where $|I_1| = \lfloor k/2 \rfloor$ and $|I_2| = \lceil k/2 \rceil$. The rows of Z are then split in two parts Z_1 and Z_2. Randomly select a σ-element subset L of the redundant set $J = N \setminus I$.

2. For each linear combination Λ_1 (resp. Λ_2) of p rows of matrix Z_1 (resp. Z_2), compute $\Lambda_{1|L}$ (resp. $\Lambda_{2|L}$) and store all these values in a hash table with 2^σ entries.

3. Using the hash table consider all pairs of linear combinations (Λ_1, Λ_2) such that $\Lambda_{1|L} = \Lambda_{2|L}$ and check whether $\mathrm{wt}((\Lambda_1 + \Lambda_2)_{|J \setminus L}) = w - 2p$.

4. Randomly choose $\lambda \in I$ and $\mu \in J$ such that $Z_{\lambda,\mu} = 1$. Replace I with $(I \setminus \{\lambda\}) \cup \{\mu\}$ by updating matrix Z by a pivoting operation.

A codeword c of weight w is then exhibited when the selections I, I_1 and L satisfy

$$\mathrm{wt}(c_{|I_1}) = \mathrm{wt}(c_{|I_2}) = p \text{ and } \mathrm{wt}(c_{|L}) = 0 \tag{1}$$

Parameters p and σ have to be chosen in order to minimize the running-time of the algorithm.

4 Theoretical Running-Time

We give here an explicit and computable expression for the work factor of this algorithm, *i.e.* the average number of elementary operations it requires. This analysis is essential in particular for finding the values of parameters p and σ which minimize the running-time of the algorithm.

4.1 Modelization of the Algorithm by a Markov Chain

The average number of iterations performed by the algorithm is not the same as the one performed by the initial Stern's algorithm since the successive information sets are not independent anymore. Hence the algorithm must be modelized by a discrete-time stochastic process.

Let c be the codeword of weight w to recover and $\mathrm{supp}(c)$ its support. Let I be the information set and I_1, I_2 and L the other selections corresponding to the i-th iteration. The i-th iteration can then be represented by a random variable X_i which corresponds to the number of non-zero bits of c in I. This random variable then takes its values in the set $\{1, \ldots, w\}$. But if this number equals $2p$ we have to distinguish two cases depending of whether condition (1) is satisfied or not. The state space of the stochastic process $\{X_i\}_{i\in\mathbf{N}}$ is therefore $\mathcal{E} = \{1, \ldots, 2p-1\} \cup \{(2p)_S, (2p)_F\} \cup \{2p+1, \ldots, w\}$ where

$$\begin{array}{lll} X_i = u & \text{iff} & |I \cap \mathrm{supp}(c)| = u, \quad \forall u \in \{1, \ldots, 2p-1\} \cup \{2p+1, \ldots, w\} \\ X_i = (2p)_F & \text{iff} & |I \cap \mathrm{supp}(c)| = 2p \text{ and } (|I_1 \cap \mathrm{supp}(c)| \neq p \text{ or } |L \cap \mathrm{supp}(c)| \neq 0) \\ X_i = (2p)_S & \text{iff} & |I_1 \cap \mathrm{supp}(c)| = |I_2 \cap \mathrm{supp}(c)| = p \text{ and } |L \cap \mathrm{supp}(c)| = 0 \end{array}$$

The success space is then $\mathcal{S} = \{(2p)_S\}$ and the failure space is $\mathcal{F} = \mathcal{E} \setminus \{(2p)_S\}$.

Definition 2. *A stochastic process $\{X_i\}_{i\in\mathbf{N}}$ is a* Markov chain *if the probability that it enters a certain state only depends on the last state it occupied. A Markov chain $\{X_i\}_{i\in\mathbf{N}}$ is* homogeneous *if for all states u and v, the conditional probability $Pr[X_i = v / X_{i-1} = u]$ does not depend on i.*

Proposition 1. *The stochastic process $\{X_i\}_{i\in\mathbf{N}}$ associated with the algorithm is an homogeneous Markov chain.*

Proof. The selections I, I_1, I_2 and L corresponding to the i-th iteration only depend on the previous information window since I_1, I_2 and L are randomly chosen. We then have for all i and for all $(u_0, u_1, \cdots, u_i) \in \mathcal{E}$,

$$Pr[X_i = u_i / X_{i-1} = u_{i-1}, X_{i-2} = u_{i-2}, \cdots X_0 = u_0] = Pr[X_i = u_i / X_{i-1} = u_{i-1}]$$

Furthermore this probability does not depend on the iteration. Hence there exists a matrix P such that :

$$\forall i \in \mathbf{N},\ \forall (u,v) \in \mathcal{E}^2,\ \ Pr[X_i = v / X_{i-1} = u] = P_{u,v}$$

The Markov chain $\{X_i\}_{i\in\mathbf{N}}$ is therefore completely determined by its initial probability vector $\pi_0 = (Pr[X_0 = u])_{u\in\mathcal{E}}$ and its transition matrix P. Both of these quantities can be easily determined as two successive information sets differ from only one element.

Proposition 2. *The transition matrix P of the homogeneous Markov chain associated with the algorithm is given by:*

$$\begin{aligned}
P_{u,u} &= \frac{k-u}{k} \times \frac{n-k-(w-u)}{n-k} + \frac{u}{k} \times \frac{w-u}{n-k} \text{ for all } u \notin \{(2p)_S, (2p)_F\} \\
P_{u,u-1} &= \frac{u}{k} \times \frac{n-k-(w-u)}{n-k} \text{ for all } u \neq 2p+1 \\
P_{u,u+1} &= \frac{k-u}{k} \times \frac{w-u}{n-k} \text{ for all } u \neq 2p-1 \\
P_{u,v} &= 0 \text{ for all } v \notin \{u-1, u, u+1\} \\
P_{(2p)_F,(2p)_F} &= (1-\beta)\left[\frac{k-2p}{k} \times \frac{n-k-(w-2p)}{n-k} + \frac{2p}{k} \times \frac{w-2p}{n-k}\right] \\
P_{2p+1,(2p)_F} &= (1-\beta)\left[\frac{2p+1}{k} \times \frac{n-k-(w-(2p+1))}{n-k}\right] \\
P_{2p-1,(2p)_F} &= (1-\beta)\left[\frac{k-(2p-1)}{k} \times \frac{w-(2p-1)}{n-k}\right] \\
P_{2p+1,(2p)_S} &= \beta\left[\frac{2p+1}{k} \times \frac{n-k-(w-(2p+1))}{n-k}\right] \\
P_{2p-1,(2p)_S} &= \beta\left[\frac{k-(2p-1)}{k} \times \frac{w-(2p-1)}{n-k}\right] \\
P_{(2p)_F,(2p)_S} &= \beta\left[\frac{k-2p}{k} \times \frac{n-k-(w-2p)}{n-k} + \frac{2p}{k} \times \frac{w-2p}{n-k}\right] \\
P_{(2p)_S,(2p)_S} &= 1 \quad \text{and} \quad P_{(2p)_S,u} = 0 \text{ for all } u \neq (2p)_S
\end{aligned}$$

$$\text{where } \beta = Pr[X_i = (2p)_S \ / \ |I \cap supp(e)| = 2p] = \frac{\binom{2p}{p}\binom{k-2p}{k/2-p}}{\binom{k}{k/2}} \ \frac{\binom{n-k-w+2p}{\sigma}}{\binom{n-k}{\sigma}}$$

The initial probability vector π_0 is

$$\begin{aligned}
\pi_0(u) &= \frac{\binom{w}{u}\binom{n-w}{k-u}}{\binom{n}{k}} \text{ if } u \notin \{(2p)_F, (2p)_S\} \\
\pi_0((2p)_F) &= \frac{(1-\beta)\binom{w}{2p}\binom{n-w}{k-2p}}{\binom{n}{k}} \\
\pi_0((2p)_S) &= \frac{\beta\binom{w}{2p}\binom{n-w}{k-2p}}{\binom{n}{k}}
\end{aligned}$$

The only persistent space of this Markov chain, *i.e.* a maximal state subset which cannot be left once it is entered, exactly corresponds to the success space $\mathcal{S}$. Since this subset contains only one state which is an absorbing state, *i.e.* a state which once entered is never left, this chain is by definition an absorbing chain. A basic property of absorbing Markov chains with a finite state space is that, no matter where the process starts, the probability that the process is in an absorbing state after n steps tends to 1 as n tends to infinity. We then deduce that our algorithm converges.

Expected Number of Iterations The absorbing chain property also enables us to compute the average number of iterations performed by the algorithm.

Proposition 3. [KS60] *If $\{X_i\}_{i\in\mathbf{N}}$ is a finite absorbing Markov chain with transition matrix P, and Q is the sub-stochastic matrix corresponding to transitions among the transient states — the non-persistent states —,* i.e. $Q = (P_{u,v})_{u,v\in\mathcal{F}}$ *then $(Id - Q)$ has an inverse R called the fundamental matrix of the chain and*

$$R = \sum_{m=0}^{\infty} Q^m = (Id - Q)^{-1}.$$

The average number of iterations performed by the algorithm can then be deduced from this fundamental matrix.

Theorem 1. *The expectation of the number of iterations N required until $\{X_i\}_{i\in\mathbf{N}}$ reaches the success state $(2p)_S$ is given by:*

$$E(N) = \sum_{u\in\mathcal{F}} \pi_0(u) \sum_{v\in\mathcal{F}} R_{u,v}$$

where R is the corresponding fundamental matrix.

Proof.

$$\begin{aligned} E(N) &= \sum_{n=0}^{\infty} nPr[X_n \in \mathcal{S} \text{ and } X_{n-1} \in \mathcal{F}] \\ &= \sum_{n=0}^{\infty} \sum_{m=0}^{n-1} Pr[X_n \in \mathcal{S} \text{ and } X_{n-1} \in \mathcal{F}] \end{aligned}$$

Applying Fubini's theorem, we get

$$\begin{aligned} E(N) &= \sum_{m=0}^{\infty} \sum_{n=m+1}^{\infty} Pr[X_n \in \mathcal{S} \text{ and } X_{n-1} \in \mathcal{F}] \\ &= \sum_{m=0}^{\infty} Pr[X_m \in \mathcal{F}] \end{aligned}$$

$$= \sum_{m=0}^{\infty} \sum_{u \in \mathcal{F}} \sum_{v \in \mathcal{F}} Pr[X_m = v \ / \ X_0 = u]$$

$$= \sum_{u \in \mathcal{F}} \pi_0(u) \sum_{v \in \mathcal{F}} \sum_{m=0}^{\infty} (Q^m)_{u,v} = \sum_{u \in \mathcal{F}} \pi_0(u) \sum_{v \in \mathcal{F}} R_{u,v}$$

Variance of the Number of Iterations The fundamental matrix also gives the variance of the number of iterations, which estimates the deviation from the average work factor of the effective computational time required by the algorithm.

Theorem 2. *The variance of the number of iterations N required until $\{X_i\}_{i \in \mathbf{N}}$ reaches the success state is given by:*

$$V(N) = \sum_{u \in \mathcal{F}} \pi_0(u) \sum_{v \in \mathcal{F}} (2R_{u,v} - \delta_{u,v}) E_v(N) \ - \left(\sum_{u \in \mathcal{F}} \pi_0(u) E_u(N) \right)^2$$

where $\delta_{i,j}$ is the Kronecker symbol and $E_u(N)$ is the average number of iterations performed by the process when it starts in state u, i.e.

$$E_u(N) = \sum_{v \in \mathcal{F}} R_{u,v}$$

Distribution of the Number of Iterations Besides the average number of iterations we often want to estimate the probability that the algorithm will succeed after a fixed number of iterations. But the approximation given by Tchebychev's inequality is usually very rough. A much more precise evaluation is obtained by raising the transition matrix of the Markov chain to the corresponding power. We actually have:

Proposition 4. *Let P be the transition matrix of the Markov chain associated with the algorithm. If $P = L^{-1} \Lambda L$ where Λ is a diagonal matrix, then the probability that the algorithm will succeed after N iterations is given by*

$$\sum_{u \in \mathcal{E}} \pi_0(u) \ \left(L^{-1} \Lambda^N L \right)_{u,(2p)_S}$$

4.2 Average Number of Operations by Iteration

We now give an explicit expression of the average number of operations performed at each iteration.

1. There are exactly $\binom{k/2}{p}$ linear combinations of p rows of matrix Z_1 (resp. Z_2); computing each of them on a σ-bit selection and putting it in the hash table requires $p\sigma$ binary additions.

2. The average number of pairs (Λ_1, Λ_2) such that $(\Lambda_1 + \Lambda_2)_{|L} = 0$ is equal to $\frac{\binom{k/2}{p}^2}{2^\sigma}$. For each of them we perform $2p-1$ additions of $(n-k-\sigma)$-bit words for computing $(\Lambda_1 + \Lambda_2)_{|J \setminus L}$ and a weight-checking.
3. We need $K(p\binom{k/2}{p} + 2^\sigma)$ more operations to perform the dynamic memory allocation where K is the size of a computer word (K=32 or 64).
4. The average work factor involved in the pivoting procedure for updating matrix Z is $\frac{1}{2}k(n-k)$.

Hence the average number of elementary operations performed at each iteration is:

$$\Omega_{p,\sigma} = 2p\sigma\binom{k/2}{p} + 2p(n-k-\sigma)\frac{\binom{k/2}{p}^2}{2^\sigma} + K\left(p\binom{k/2}{p} + 2^\sigma\right) + \frac{k(n-k)}{2} \quad (2)$$

Proposition 5. *Suppose that the number of codewords of weight w is $\mathcal{A}_w$. The overall work factor required by the algorithm is:*

$$W_{p,\sigma} = \frac{\Omega_{p,\sigma} E(N)}{\mathcal{A}_w} \quad (3)$$

where $E(N)$ is given by Theorem 1 and $\Omega_{p,\sigma}$ by Equation (2).

Since each term in the previous expression can be explicitly computed, we are now able to determine the parameters p and σ which minimize the work factor required by the algorithm when the size of the code and the weight w of the searched codeword are given. Such a theoretical expression of the work factor is commonly used to assess the efficiency of an algorithm and to decide whether a given problem is computationally feasible. It is also applied to the automatic optimization of the parameters. But the sharpest optimization can only be performed by replacing in Equation (3) the theoretical value of $\Omega_{p,\sigma}$ by the effective average CPU time of an iteration.

5 Cryptanalysis of McEliece Cryptosystem

5.1 Work Factor Versus Probability of Success

Table 2 gives the optimal parameters and the number of binary operations involved in an attack of the previous cryptosystems.

Cryptanalyzing McEliece cipher with its original parameters then requires $2^{64.2}$ binary operations [CC98]. This new attack is certainly still infeasible but it runs 128 times faster than Lee-Brickell's attack [LB88]. As a comparison the cryptanalysis of Stern's identification scheme using van Tilburg's algorithm has an average number of iterations of $2^{57.0}$, and an estimated work factor of $2^{72.9}$ [vT94]. An obvious method for speeding up the cryptanalysis consists in distributing the algorithm: using a network of 1000 computers we only need $2^{54.2}$ operations for breaking McEliece cipher.

cryptosystem	McEliece	Stern	Véron
code	[1024,524]	[512,256]	[512,120]
w	50	56	114
optimal parameters	$p = 2, \sigma = 18$	$p = 2, \sigma = 15$	$p = 2, \sigma = 13$
average number of iterations	$9.85\ 10^{11}$	$2.16\ 10^{14}$	$1.74\ 10^{12}$
standard deviation of the number of iterations	$9.85\ 10^{11}$	$2.16\ 10^{14}$	$1.74\ 10^{12}$
work factor	$2^{64.2}$	$2^{69.9}$	$2^{61.2}$

Table 2. Work factor required for cryptanalyzing some public-key systems based on error-correcting codes

But the standard deviation of the number of iterations involved in cryptanalyzing all these systems roughly equals its average. This spread implies that an infeasible average work factor is not sufficient to guarantee that these cryptosystems are secure: it is necessary to estimate the probability that our algorithm will be successful after a feasible number of iterations. This can be done by raising the transition matrix of the associated Markov chain to the corresponding power as described in Proposition 4. We then obtain that the work factor required for decoding a [1024,524,101]-binary code up to its error-correcting capability with probability 0.5 only represents 69 % of the average work factor. And if the work factor is limited to 2^{51}, *i.e.* to 10^8 iterations, the probability that a message in McEliece cipher will be decrypt is 10^{-4}. Since 1000 iterations of the optimized algorithm are performed in 10 minutes on a workstation DEC alpha at 433 MHz, decrypting one message out of 10,000 requires 2 months and 14 days with 10 such computers (see Figure 1). The relatively high proportion of decrypted messages in a reasonable time implies that McEliece system with its original parameters is not secure as long as the enemy has a few ten fast workstations. A similar study shows that the parameters proposed in Stern's identification scheme make it much more secure. An eleven-month computation time on 10 DEC alpha enables us to recover the secret key of a user in only one case out of 100,000. This only implies that the lifetime of the keys must be less than one year. The parameters proposed by Véron significantly reduce the number of transmitted bits in each identification procedure but they impose a much shorter lifetime of the keys since 56 days on 10 of our workstations are sufficient to find the secret key of a user with a probability greater than 1/3500.

5.2 Partial Attacks on McEliece and Niederreiter Cryptosystems

McEliece and Niederreiter cryptosystems otherwise present some weaknesses since the knowledge of a small number of bits of the plaintext is sufficient to recover it in its entirety. The knowledge of some plaintext bits in McEliece cipher allows to accordingly reduce the dimension of the code we consider in the

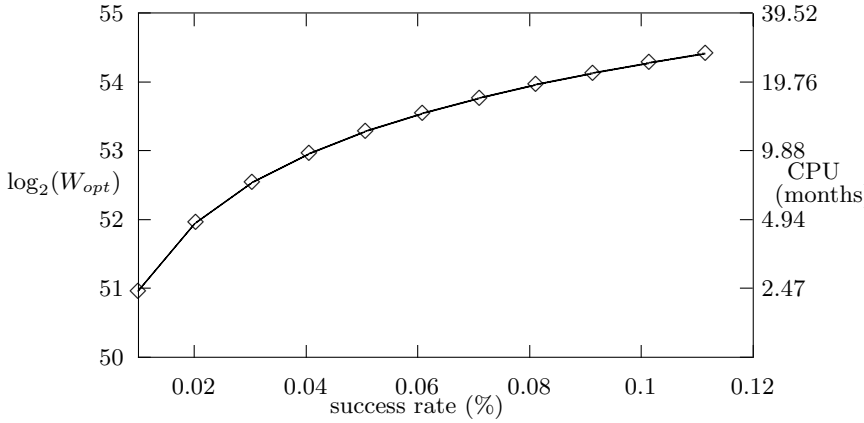

Fig. 1. Computational effort required for cryptanalyzing McEliece cryptosystem as a function of the proportion of messages successfully decrypted: the CPU time is given for 10 workstations DEC alpha at 433 MHz in parallel.

attack. If we assume that 2^{50} binary operations is a feasible work factor, it is then possible to decode up to distance 50 a [1024,404]-binary code with our algorithm. This means that the knowledge of 120 plaintext bits (*i.e.* 23 % of the plaintext) is sufficient to recover the whole plaintext in a reasonable time.

A similar attack on Niederreiter cryptosystem consists in assuming that some error positions are known by the enemy. The problem is then to determine the distance up to which a [1024,524]-binary code can be decoded. If the work factor is limited to 2^{50} binary operations, we obtain that the knowledge of 15 error positions out of the 50 introduced in McEliece and Niederreiter systems enables us to recover the plaintext. This small proportion notably implies that generating the error-vector with a noisy channel is insecure if this provides some errors whose weight is too small.

6 Conclusion

We have then proved that the security of McEliece cipher is insufficient when its original parameters are used. But this public-key system is still a valid alternative to RSA once its parameters are modified. For example if the secret key is chosen amongst the Goppa codes of length 2048, dimension 1608 and minimum distance 81, the average work factor of our attack is roughly 2^{100}. Even with these parameters the performance of McEliece cipher remains much better than the one of RSA: the costs of encryption and decryption per information bit with Niederreiter's version are respectively 45 times and 70 times lower than with RSA-1024. But the huge size of the public-key (more than 88000 bytes in this case) may often dissuade from using this cipher.

References

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

KS60. J.G. Kemeny and J.L. Snell. *Finite Markov chains*. Springer-Verlag, 1960.

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

LDW94. Y.X. Li, R.H. Deng, and X.M. Wang. On the equivalence of McEliece's and Niederreiter's public-key cryptosystems. *IEEE Transactions on Information Theory*, IT-40(1):271–273, 1994.

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

McE78. R.J. McEliece. A public-key cryptosystem based on algebraic coding theory. *JPL DSN Progress Report*, pages 114–116, 1978.

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

Omu72. J.K. Omura. Iterative decoding of linear codes by a modulo-2 linear programm. *Discrete Math*, (3):193–208, 1972.

Sen94. N. Sendrier. On the structure of a randomly permuted concatenated code. In P. Charpin, editor, *EUROCODE 94 - Livre des résumés*, pages 169–173. INRIA, 1994.

Sen95. N. Sendrier. On the structure of a randomly permuted concatenated code. Technical Report RR-2460, INRIA, January 1995.

Sen96. N. Sendrier. An algorithm for finding the permutation between two equivalent binary codes. Technical Report RR-2853, INRIA, April 1996.

SS92. V.M. Sidelnikov and S.O. Shestakov. On cryptosystems based on generalized Reed-Solomon codes. *Diskretnaya Math*, 4:57–63, 1992.

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

Ste93. J. Stern. A new identification scheme based on syndrome decoding. In D.R. Stinson, editor, *Advances in Cryptology - CRYPTO'93*, number 773 in Lecture Notes in Computer Science, pages 13–21. Springer-Verlag, 1993.

Vér95. P. Véron. *Problème SD, Opérateur Trace, schémas d'identification et codes de Goppa*. PhD thesis, Université de Toulon et du Var, 1995.

vT94. J. van Tilburg. *Security-analysis of a class of cryptosystems based on linear error-correcting codes*. PhD thesis, Technische Universiteit Eindhoven, 1994.