On the McEliece Public-Key Cryptosystem

Johan van Tilburg

Dr. Neher Laboratories, Department of Applied Mathematics
P.O. Box 421, 2260 AK Leidschendam, the Netherlands

Abstract

Based on an idea by Hin, the method of obtaining the original message after selecting k of n coordinates at random in the McEliece public-key cryptosystem is improved. The attack, which is more efficient than the attacks previously proposed, is characterized by a systematic method of checking and by a random bit swapping procedure. An optimization procedure similar to the one proposed by Lee and Brickell is used to improve the attack. The attack is highly suitable for parallel and pipelined implementation. The work factor and the values, which yield 'maximum' security for the system are given.

It is shown that the public-key can be reduced to $k \times (n-k)$ bits.

1 Introduction

At Crypto'87 Adams and Meijer [1] presented a paper in which the 'optimum' values for the parameters of the McEliece public-key cryptosystem [9] are given. As shown in [1] these values improve the cryptanalytic complexity of the system and increase the information rate. As noted in [4,9] there are several ways of attacking McEliece's cryptosystem. Of the known attacks, the one which requires the least effort is based on decoding a more or less arbitrary linear code containing correctable errors. It has been proved in [2] that the general decoding problem for linear codes is NP-complete, so one certainly expects that for sufficiently large code parameters, the minimal effort for this attack will become computationally infeasable. The best known attack is based on selecting and solving k of n equations obtained from the (publicly known) encryption matrix and the cryptogram. Thereafter it is necessary to verify whether the obtained solution is unique and gives the correct plaintext. If the solution is not correct, then a new set of k equations has to be selected etc. For the attack it was shown in [1] that for a suitable choice of the parameters this minimal effort can be maximized.

This paper gives an improved method to obtain the original message after selecting k of n cryptogram bits. A bit swapping procedure is used to randomly renew the set of k-bits one bit at a time. A fast validation wether the selected k-bits are

S. Goldwasser (Ed.): Advances in Cryptology - CRYPTO '88, LNCS 403, pp. 119-131, 1990.

error-free and the corresponding columns of the publicly known encryption matrix are linearly independent is part of the algorithm.

At the same time when this paper was accepted for presentation at Crypto'88, Lee and Brickell [7] presented an elegant attack on the McEliece public-key cryptosystem at Eurocrypt'88. Their attack is based on a generalization of two well known attacks and includes a systematic method for checking whether the obtained message agrees with the original message and is closely related to our attack.

Sections 2 and 3 describe the public-key cryptosystem and some well known attacks on this system. Section 4 discusses the basics of the proposed attack including the way of validation. The algorithm, based on a bit swapping procedure, is subsequently given in the next section. Section 6 considers in more details the bit swapping procedure. In section 7 the work factor is discussed and in Section 8 an optimization similar to the one proposed by Lee and Brickell is used to improve the attack. Finally we note in Section 9, that the public-key can be reduced to $k \times (n-k)$ bits without affecting the security of the system.

2 McEliece's Cryptosystem

The McEliece public-key cryptosystem can be easily understood from the following description. Let C be a linear $[n, k, d]$ code over $GF(2)$ with code length n, dimension k and minimum distance d. Let the $k \times n$ matrix G be a generator matrix of C and let the $(n-k) \times n$ matrix H be a parity check matrix of C. The publicly known encryption matrix E is defined by

$$E = SGP, \tag{1}$$

where S is a $k \times k$ non-singular binary matrix over $GF(2)$ and P is an $n \times n$ permutation matrix. The scheme also uses a subset Z of $GF(2)^n$ with the property that the Hamming weight $w_H(\underline{z})$ of the vectors $\underline{z} \in Z$ is less or equal than $t = (d-1)/2$. Generally $w_H(\underline{z}) = t$.

A k-message $\underline{m}$ is encrypted into the n-bit ciphertext $\underline{e}$ as follows

$$\underline{e} = \underline{m}E + \underline{z} = \underline{c} + \underline{z}, \tag{2}$$

where $\underline{c}$ is a n-bit permuted codeword from C.

Decryption is straightforward. An enciphered message $\underline{m}$ is *formally* decrypted by the following steps.

1. Compute $\underline{e}' = \underline{e}P^T$ and obtain the error pattern $\underline{z}' = \underline{z}P^T$
   Let $\underline{c}' = \underline{e}' - \underline{z}'$.

2. Calculate $\underline{m} = \underline{m}' \times S^{-1}$, where $\underline{m}'$ represents the first k-bits of $\underline{c}'$. The result is the plaintext $\underline{m}$.

This encryption scheme must satisfy the properties introduced by Diffie and Hellman [3] to become a public-key cryptosystem. Therefore the decryption process must be fast if the private-keys S, P and G are known and the decryption process must be infeasible if only the public-key E is known. Furthermore the encryption process must be fast if one has only knowledge of the public-key E. McEliece based his cryptosystem on the existence of Goppa codes, which meet the conditions for a public-key cryptosystem and can easily be generated.

We note that Goppa codes are in general not maximum distance separable codes (MDS). The only binary MDS codes are the trivial ones which are of no use in the (binary) McEliece scheme. More details about Goppa codes can be found in e.g. [8].

3 Cryptanalysis of the McEliece Cryptosystem

In this section we will discuss some general and well known attacks on the McEliece scheme. We shall not pay attention to special cases for which fast cryptanalysis exist.

3.1 Factoring the encryption matix

Let G_s denote the generator matrix G in systematic form and let the encipher matrix E be SG_sP. The number of non-singulier matrices S is given by $0.29 \times 2^{k^2}$. There exist approximately $2^{mt}/t$ generator matrices G_s for a binary irreducible t-error correcting Goppa code. And there are $n!$ possible permutation matrices. Moreover as shown by Adams and Meijer [1] the only transformation which transforms the encryption matrix E into a generator matrix G which algebraic structure allows us to use a fast decoding algorithm, is the original transformation i.e. $G = S^{-1}EP^{-1}$. Therefore we may conclude that for sufficiently large parameters it will be infeasible to obtain the private-keys S, G and P by an exhaustive search.

3.2 Recover message from cryptogram and encryption matrix

McEliece states in [9] that probably the most promising attack on his scheme consists of actually solving the basic problem, i.e. decoding a more or less arbitrary $[n, k, d]$ linear code containing t correctable errors. As it has been proved that the general decoding problem is NP-complete [2], one certainly expects that for large code parameters this attack will be infeasible.

A straightforward approach is based on a brute force *distance* search; comparing the cryptogram $\underline{e}$ to each permuted codeword $\underline{c} = \underline{m}E$. If the Hamming distance result is: $d_H(\underline{e}, \underline{c}) \leq t$, then $\underline{m}$ is the original message. However this method has a work factor of about $O(2^k)$. For $k = 654$ this becomes $2^{654} \approx 10^{197}$, which is astronomically large.

Another approach is based on a brute force search for a correct *syndrome*. Let D be the matrix HP. Clearly $ED^T = 0$. Find an error vector $\underline{z}$ with minimum weight for which $\underline{e}D^T = \underline{z}D^T$. However, it seems to be necessary to search through all solutions of this equation to find the desired $\underline{z}$ of minimum weight and has a work factor of about $O(n^t)$.

McEliece proposes in [9] to select randomly k of n ciphertext bits from $\underline{e}$ in the hope that none of the k selected bits are in error, and based on this assumption, to obtain the correct plaintext $\underline{m}$. The probability p_k of no error in the chosen k-bits of $\underline{e}$, however, is equal to

$$p_k = \frac{\binom{n-t}{k}}{\binom{n}{k}} = \prod_{i=0}^{k-1}(1 - \frac{t}{n-i}). \tag{3}$$

Selecting k-bits, which are not in error, does not guarantee that the corresponding $k \times k$ sub-matrix of E is non-singular. This only holds for maximum distance separable codes (MDS, [8]). In case of an MDS code every k columns of the encryption matrix are linearly independent. Since the Goppa codes used in the McEliece scheme are not MDS, we will have k linearly independent columns with a probability $q_k > 0$. This also holds for the encryption matrix E, since S works on the message space and P permutes the code words. Clearly, q_k can not be estimated by assuming that E is a random matrix.

The amount of work involved in solving k simultaneous equations in k unkown is k^a (e.g. $a = 2.8$ [6]). Let V_k be the average work factor if k columns are linearly dependent. Hence, before finding the message $\underline{m}$ with this attack one expects a work factor of

$$W = \gamma \times [(1 - q_k)V_k + q_k k^a] \times q_k^{-1} \times p_k^{-1}. \tag{4}$$

We can use the Hamming distance to check whether the obtained message $\underline{m}$ is correct plaintext. If the result of the Hamming distance is: $d_H(\underline{e}, \underline{\hat{m}}E) \leq t$, then $\underline{\hat{m}}$ is the original message $\underline{m}$. The additional cost to validate each message $\underline{\hat{m}}$ is therefore $O(nk)$.

Adams and Meijer [1] established by exhaustive search that for values of 'a' between 2 and 3, the maximal work-factor (without validation, $\gamma = 1$ and $q_k = 1$) is reached at $t = 37$. In this case for $a = 3$ the work-factor is approximately $2^{84.1}$, while for $t = 50$ this becomes $2^{80.7}$. As a consequence of this improvement, the value of k is increased from 524 to 654; i.e. the information rate $R = k/n$ is increased from 0.51 to 0.64.

4 Main Idea

A straightforward approach is based on a brute force distance search as mentioned in the previous section. Despite the high work factor this approach has the advantage

that there are no additional validation costs, because the validation is part of the attack itself. As suggested by Hin [5], this attack can be improved by taking the constraints imposed by the cryptogram into account. For this reason we have to restate the above attack in terms of the cryptogram $\underline{e}$ instead of the message $\underline{m}$.

For the attack to be described in the next section we need a decomposition of the encryption matrix E in the following form

$$E = S_a[I_k \backslash A_a]P_j, \tag{5}$$

where I_k is the k x k identity matrix and A_a is a k x $(n - k)$ binary matrix. Since every linear code is equivalent to a systematic code, this decomposition is always possible.

If we apply a permutation matrix P_a to $\underline{e}=\underline{m}E + \underline{z}$, then we obtain the relation

$$\underline{e}P_a = \underline{c}P_a + \underline{z}P_a, \tag{6}$$

which will be denoted as $\underline{e}_a = \underline{c}^{\wedge} + \underline{z}_a$.

The function $FKB(\underline{x})$ is defined as

$$FKB(x) \_ = FKB(x_1,x_2,\ldots ,x_k,x_{k+1},\ldots ,x_n) = xi,x_{2>}\ldots ,x_k.$$

Hence, $FKB\{\underline{x}\}$ selects the first fc-bits from a n-bit vector $\underline{x}$.

We are now able to prove the next theorem.

Theorem 1 *If P_b is a permutation matrix for which $\underline{e}_= \underline{m}E + \underline{z}$ can be written as*

$$S^{\wedge}Pj, = \underline{T}nS_aS_b[I_k\backslash A_b\} + z^{\wedge}Pb,$$

then

$$w_H\{FKB\{z_a\underline{P}_by\backslash = 0 \Leftrightarrow^* d_H[FKB\{e_aP_h)[I_k\backslash A_b],e_a\underline{P}_b] \leq t. \tag{7}$$

Proof. We have

$$d_H[\underline{m}S_aS_b[I_k|A_b], \underline{e}_aP_b] = w_H[(\underline{z}_aP_b)] \leq t.$$

From which it follows that

$$d_H[FKB\{e_aP_b \_ z^{\wedge}lhlA,,\},^{\wedge}\} \leq t$$

$Uw_H[FKB\{z_a\underline{P}_by\backslash = 0$, then it follows that d_H[F«:B(e_aFfc)[/$_{fe}$|A_{fc}],e_aPi] $\leq t$. On the other hand, if $d^{\wedge}FKB^{\wedge}P_b)[I_k\backslash A_b]$, e,^] $< \underline{t}$, then FA'B($e_a\underline{P}_i$)[/$_{fc}$| A_b) must be the codeword with whom $\underline{e}_aP_b$ corresponds to. Therefore the $FKB(\underline{e}^{\wedge}P_b)$ must be error-free, i.e. $w_H[FKB^{\wedge}P,,)) = 0$.

a

Observe that this theorem describes McEliece attack (with validation) in a more general form. During the initial phase of the attack, k cryptogram bits are randomly selected (without replacement) from $\underline{e}$. The k selected bits form the set $\mathcal{A}$ and the remaining $(n-k)$-bits are assigned to set $\mathcal{B}$. Selecting k new bits in the McEliece attack is replaced by a permutation P_b which swaps *at most* k new bits from set $\mathcal{B}$ for k-bits from set $\mathcal{A}$. The permutation is only succesful if the corresponding columns of the encryption matrix E are linearly independent as has been mentioned in section 3. The theorem states that the obtained solution is unique and gives the correct message $\underline{m}$ if the distance verification is positive. In the next section we will describe an attack based on this kind of bit swapping.

5 One Bit Swapping Attack

The McEliece attack can be considered as a k-bit swapping attack. To obtain a low complexity and to determine in a fast way if a given permutation fulfils, we will present an algorithm for a one bit swapping procedure only.

The algorithm for a one bit swapping attack consists of the following 5 steps.

Step 1 - *initialisation.*
Decompose the encipher matrix E, i.e. calculate a permutation matrix P_a and a matrix A_a such that $E = S_a[I_k|A_a]P_a^T$. Set-up a pointer table: FOR $i := 1$ TO n DO $Ptable[i] := i$.
Calculate $\underline{e}_a = \underline{e}P_a$ and up-date the pointer table.

Step 2 - *checking.*
Check if it holds that $d_H(FKB(\underline{e}_a)[I_k|A_a], \underline{e}_a) \leq t$. This can be done by checking whether there are no more than t errors with respect to $FKB(\underline{e}_a)A_a$. If there are t errors or less in the *last* $(n-k)$-bits of $\underline{e}_a$, then proceed to step 5.

Step 3 - *swapping.*
The algorithm PRP produces a pseudo-random permutation P_b. The permutation P_b swaps one column, say i, from the I_k part of the matrix $[I_k|A_a]$ for one column, say j, from the A_a part. The swapping procedure is as follows.

REPEAT
    Select permutation P_b from PRP.
    IF column j has **not** an '1' as i-th entry
        THEN P_b does not fulfil
        ELSE P_b fulfils
            $swap(Ptable[i], Ptable[j])$;
            $\underline{e}_a := \underline{e}_a P_b$
    FI;
UNTIL P_b fulfils.

Step 4 - *up-dating.*
Compute $[I_k|A_a]P_b$ into the form $S_b[I_k|A_b]$. The new 'stripped' generator matrix will be defined as $[I_k|A_a] := [I_k|A_b]$. Compute $\underline{e}_a := \underline{e}_a P_b$.
Proceed to step 2.

Step 5 - *calculate plaintext.*
At this stage there are no errors in $FKB(\underline{e}_a)$ and consequently $FKB(\underline{z}_a)$ is 0. The first k positions of the pointer table (*Ptable*) show locations in $\underline{e}$ without error. Select the corresponding columns of the encryption matrix E which are guaranteed linearly independent and calculate the plaintext $\underline{m}$.

6 Number of Swaps

A ciphertext $\underline{e}$ is obtained by adding an error vector $\underline{z}$ with Hamming weight t to a permuted codeword $\underline{c} = \underline{m}E$. Therefore there are t 'disturbed' bits in the cryptogram $\underline{e}$ which differ from the permuted codeword bits in $\underline{c}$. In the attack bits are repeatedly swapped in order to obtain k non-disturbed ciphertext bits. During the initial phase of the attack, k cryptogram bits are randomly selected (without replacement) from $\underline{e}$. The k selected bits form the set $\mathcal{A} = \{e_\nu\}$ and the remaining $(n-k)$-bits are assigned to set $\mathcal{B} = \{e_\mu\}$. The procedure $swap(e_\nu, e_\mu)$, which swaps a bit from set $\mathcal{A}$ for a bit from set $\mathcal{B}$, has one of the following values

- $s = 0$ if a (non-)disturbed bit e_ν is swapped for a (non-)disturbed bit e_μ,
- $s = -1$ if a disturbed bit e_ν is swapped for a non-disturbed bit e_μ,
- $s = +1$ if non-disturbed bit e_ν is swapped for a disturbed bit e_μ.

For the conditional probability $Pr\{i+s|i\}$, i.e. the probability that an event with i disturbed bits e_ν in $\mathcal{A}$ is followed after a swap by an event with $i+s$ disturbed bits e_ν in $\mathcal{A}$, we find

$$\sum_s Pr\{i+s|i\} = 1,\ Pr\{i-1|i\} = \frac{i(n-k-t+i)}{k(n-k)} \text{ and } Pr\{i+1|i\} = \frac{(k-i)(t-i)}{k(n-k)} \quad (8)$$

If $N_i = N_i(n,k,t)$ denotes the average work factor for a state with i errors, then N_{i-1} follows from

$$\begin{aligned} N_{i-1} &= \sum_{r=0}^{\infty} \sum_{i=0}^{r} \binom{r}{i} \Big[Pr\{i-1|i-1\}^{i+1} Pr\{i|i-1\}^{r-i} + \\ &\qquad + Pr\{i-1|i-1\}^{i} Pr\{i|i-1\}^{r-i+1} (N_i+2) \Big] \\ &= \frac{Pr\{i-1|i-1\} + Pr\{i|i-1\} \cdot (N_i+2)}{Pr\{i-2|i-1\}} \end{aligned} \quad (9)$$

Table 1 The average number of swaps N_i for state i.

n=1024				k=624			t=39		
N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}
$2^{59.4}$	$2^{53.3}$	$2^{48.3}$	$2^{43.9}$	$2^{39.9}$	$2^{36.3}$	$2^{33.0}$	$2^{30.0}$	$2^{27.3}$	$2^{24.7}$

The average number of random swaps (with replacement) $N(n,k,t)$ depending on all the possible $\binom{n}{t}$ initial states is given by

$$N(n,k,t) = \sum_{j=1}^{t} \frac{\binom{k}{j}\binom{n-k}{t-j}}{\binom{n}{t}} \sum_{i=1}^{j}(N_i + j). \tag{10}$$

7 Work factor

Let W_i denotes the average work factor of step i. With a probability of approximately one half ($q_k \approx \frac{1}{2}$) a permutation P_b is found in step 3 which can be used. The permutation P_b can be generated and validated in a fast way and independent from the main algorithm. Steps 2 and 4 are only executed when a correct permutation P_b is determined. Therefore we can neglect W_3 ($V_k \approx 0$ in equation 4). Since steps 1 and 5 are executed only once, we can neglect W_1 and W_5 in view of the on average $N(n,k,t)$ repeated steps 2 and 4. Therefore the main algorithm has an average work factor

$$W = (W_2 + W_4) \times N(n,k,t). \tag{11}$$

$M(j,i)$ is a notation for j simultaneous i-bit multiplications and similarly $A(j,i)$ denotes j simultaneous i-bit additions. If simultaneous i-bit operations are left out of consideration, then e.g. $M(j,i)$ becomes, with a little ambiguity, $jM(1,i) = jM(i)$. Moreover if only 1 bit operations are considered, then this notation reduces to $M(j,i) = ji$.

For W_2 we find that

$$W_2 = M(n-k,k), \tag{12}$$

$$(t+1) \cdot M(k) \leq W_2 \leq (n-k) \cdot M(k). \tag{13}$$

On average W_2 will be $2(t+1) \cdot M(k)$. (14)

For W_4 we obtain

$$W_4 = A(k-1, n-k-1), \tag{15}$$

$$A(n-k-1) \leq W_4 \leq (k-1) \cdot A(n-k-1). \tag{16}$$

$$\text{On average } W_4 \text{ will be } \frac{k-1}{2} \cdot A(n-k-1). \tag{17}$$

In general the work factor (11) becomes

$$W = [M(n-k,k) + A(k-1,n-k-1)] \times N(n,k,t). \tag{18}$$

If we use for example the average values (14) and (17) in (11), then we obtain the following work factor

$$W = [2(t+1) \cdot M(k) + \frac{k-1}{2} \cdot A(n-k-1)] \times N(n,k,t). \tag{19}$$

This way we find for the overall average work factor (without parallelism etc.)

$$W = \frac{1}{2} \cdot [4k(t+1) + k(n-k) - (n-1)] \times N(n,k,t). \tag{20}$$

The maximum value of W is approximately $2^{76.8}$ for $t = 39$. The average number of swaps is in this case $2^{59.4}$.

8 Further Improvements

At Eurocrypt'88, Lee and Brickell [7] presented a generalized attack on the McEliece scheme. Briefly, the attack is as follows: a set of k-bits is selected at random from the cryptogram. The set is tested by an exhaustive search for an error pattern with no more than j errors. In case an error pattern is found with j or less errors, the algorithm stops, otherwise a new set of k-bits is selected. For $j = 0$ the *traditional* attack is obtained and a *brute force* distance search for $j = t$. Lee and Brickell have found (with some assumptions) that the optimum j which minimizes the maximum work factor is 2 for all values of useful code parameters.

8.1 Search for one correctable error

Lee and Brickell propose in [7] a random update of only one bit instead of all the k-bits at the same time. This bit swapping is actually one of the basics of our method. From section 6 it follows that the last steps, i.e. removing the last j errors, dominate the work factor. An optimization procedure similar to the Lee-Brickell method is used to speed-up our attack. While in our case there is a *trade-off* between the swap-complexity and the complexity of the exhaustive search with checking, the optimum j which minimizes the maximum work factor is found to be 1. This low optimum is due to the low complexity of the swap-procedure, which is $O(k \times (n-k))$. For a single error pattern search a new step has to be added to the attack described

in section 5. If $\underline{u}_i$ is the i-th unit vector, then the new step becomes

Step 2a - *search for one correctable error*
For $1 \leq i \leq k$, check if it holds that $d_H(FKB(\underline{e}_a - \underline{u}_i)[I_k|A_a], \underline{e}_a) \leq t$. This can be done by checking whether there are no more than t errors with respect to $FKB(\underline{e}_a - \underline{u}_i)A_a$. If for a certain i there are t or less errors in the *last* $(n-k)$-bits of $\underline{e}_a$, then correct bit $Ptable[i]$ in $\underline{e}$ and proceed to step 5.

For the average work factor W_{2a} for step $2a$ we find that

$$W_{2a} = 2(t+1)k. \tag{21}$$

The maximum overall work factor W is approximately $2^{71.1}$ for $t = 39$. The average number of swaps is in this case $2^{53.4}$.

8.2 Partial search for two correctable errors

Since the value of W_{2a} is small compared to $(W_2 + W_4)$, a partial search for patterns with two errors can be considered additionally. The number of partial search patterns used in step $2b$ below is denoted by n_s.

Step 2b - *partial search for two correctable errors*
For $1 \leq i < k$ and $i < j \leq k$, check if it holds that $d_H(FKB(\underline{e}_a - \underline{u}_i - \underline{u}_j)[I_k|A_a], \underline{e}_a) \leq t$. This can be done by checking whether there are no more than t errors with respect to $FKB(\underline{e}_a - \underline{u}_i - \underline{u}_j)A_a$. If for certain i and j there are t or less errors in the *last* $(n-k)$-bits of $\underline{e}_a$, then correct bit $Ptable[i]$ and $Ptable[j]$ in $\underline{e}$ and proceed to step 5. If $\#\{(i,j)\} = n_s$ then proceed to step 3.

For the average work factor W_{2b} for step $2b$ we find that

$$W_{2b} = 2(t+1) \cdot n_s. \tag{22}$$

If we assume a uniform distribution of the error patterns, then the probability of succes follows from

$$Pr\{\text{Succes Partial Search}|i=2\} = n_s / \binom{k}{2} \tag{23}$$

The average work factor N_i for states 3 to t follows from equation 9. The average work factor N_2 for state $i = 2$ becomes

$$N_2 = \frac{Pr\{i=2|i=2\} + Pr\{i=3|i=2\} \cdot (N_3+2)}{Pr\{i=1|i=2\} + Pr\{\text{Succes Partial Search}|i=2\}} \tag{24}$$

The maximum overall work factor W is approximately $2^{69.7}$ for $t = 39$ and $n_s = 5769$. The average number of swaps is in this case $2^{50.3}$.

8.3 General Attack

Let P_b be a permutation matrix which swaps *at most* i-columns from the I_k part for i-columns in the A_a part of the $[I_k|A_a]$ matrix. Let S be a subset of $GF(2)^k$ with the property that the Hamming weight $w_H(\underline{s})$ of the vectors $\underline{s} \in S$ is less or equal than j. If *all* vectors $\underline{s}$ with Hamming weight equal to j are used during one search, then the attack is called *complete* otherwise *partial.*

The general **[i,j] – swap attack** follows from

1. *initialisation*
   - decompose encipher matrix: $E = S_a[I_k|A_a]P_a^T$
   - calculate $\underline{e}_a = \underline{e}P_a$
   - set-up pointer table
2. *checking*
   - Check if there exists an $\underline{s} \in S$ such that $d_H[(FKB(\underline{e}_a) - \underline{s})[I_k|A_a], \underline{e}_a] \leq t$. If there exists such an $\underline{s} \in S$, then correct $\underline{e}$ with $\underline{s}$ using the pointer table and proceed to step 4.
3. *swapping*
   - select a permutation P_b which fulfils
   - P_b swaps at most i-columns from the I_k part for i-columns in the A_a part of the $[I_k|A_a]$ matrix
   - Transform $[I_k|A_a]P_b$ into $S_b[I_k|A_b]$
   - let $[I_k|A_a] := [I_k|A_b]$ and $\underline{a}_a := \underline{a}_a P_b$
   - up-date the pointer table and proceed to step 2
4. *calculate plaintext*
   - the first k positions of the pointer table show locations in $\underline{e}$ without error
   - select the corresponding columns of the encryption matrix E which are guaranteed linearly independent
   - calculate the plaintext $\underline{m}$

For a *complete* $[i,j]$-swap attack with $i < j$ all search patterns $\underline{s} \in S$ have to be used during the initial round. However for the subsequent rounds only the search patterns with $(j-i) \leq w_H(\underline{s}) \leq j$ have to be considered, since there are at least $(j-i)$ errors after each i-swap. For a *partial* attack this becomes $(j-i-1) \leq w_H(\underline{s}) \leq j$.

9 Reduced Public-Key

From the attack described in section 5 and the fact that the existence of more than one trapdoor in the system is unlikely [1], it follows that although a factorisation of E is found, no information about the original S, G and P matrices is revealed. For this reason a $k \times (n-k)$ matrix A of a decomposition of the encryption matrix $E = SGP = S'[I_k|A]P'$ can be published instead of E. Encryption can be done in the following way

- Use a publicly known seed $\underline{s}$. The seed $\underline{s}$ generates a new non-singular binary matrix S^* . The encryption scheme becomes

$$\underline{e} = \underline{m}E + \underline{z} = \underline{m}S^*[I_k|A] + \underline{z} = \underline{w}[I_k|A] + \underline{z}. \tag{25}$$

- Use an publicly known invertible function f which transforms a message $\underline{m} \in GF(2)^k$ into a word $\underline{w} \in GF(2)^k$. The function f may also depend on the error vector $\underline{z}$. In this case the following encryption scheme is obtained

$$\underline{e} = \underline{m}E + \underline{z} = f(\underline{m}, \underline{z}).[I_k|A] + \underline{z} = \underline{w}[I_k|A] + \underline{z}. \tag{26}$$

To keep the seed $\underline{s}$, used to generate a non-singulier matrix S^*, secret does not increase the security of the system. Since a chosen-plaintext attack by majority voting of each position of a row of the encipher matrix E will be successful and reveal $[S^*|S^*A]$ and consequently S^*.

In both cases it follows that

- **The sender** generates an error vector $\underline{z}$, computes $\underline{w}$ and calculates a cryptogram $\underline{e} = \underline{w}[I_k|A] + \underline{z}$.

- **The receiver** determines the error pattern $\underline{z}$, removes it from $\underline{e}$, computes $\underline{w} = FKB(\underline{e} - \underline{c})$ and calculates the message $\underline{m} = \underline{w}S^{*-1}$ or $\underline{m} = f^{-1}(\underline{w}, \underline{z})$.

It follows that the public-key can be reduced to $n \times (n-k)$-bits. For $n = 1024$ and $t = 39$ the reduced key becomes 399 kbits.

Acknowledgement

The author wishes to thank Paul Hin for the numerous discussions we had about cryptosystems based on coding theory during our final years at the Delft University of Technology and for reading this manuscript. I wish to thank IACR for the support given which made it possible to present this paper at Crypto'88. Finally, I wish to thank Jean-Paul Boly for his comments.

References

[1] C. Adams and H. Meijer, "Security-Related Comments Regarding McEliece's Public-Key Cryptosystem", in: *Advances in Cryptology - CRYPTO '87*, Carl Pomerance ed., Lecture Notes in Computer Science # 293, Springer-Verlag, pp. 22 4-228, 1988.

[2] E.R. Berlekamp, R.J. McEliece and H.C.A. van Tilborg, "On the inherent intractability of certain coding problems", *IEEE Trans. Inform. Theory*, vol. IT-24, pp. 384-386, 1978.

[3] W. Diffie and M.E. Hellman, "New Directions in Cryptography", *IEEE Trans. Inform. Theory*, vol. IT-22, pp. 644-654, 1976.

[4] P.J.M. Hin, "Channel-Error-Correcting Privacy Cryptosystems", M.Sc. Thesis, Delft University of Technology, 1986 (in Dutch).

[5] P.J.M. Hin, Private Communication, December 1986.

[6] D.E. Knuth, *The art of computer Programming, Vol.2 / Seminumerical Algorithms*, Addison-Wesley Publishing Company, 1981.

[7] P.J. Lee and E.F. Brickell, "An observation on the Security of McEliece's Public-Key Cryptosystem", Presented at Eurocrypt'88, Davos, May 1988.

[8] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland, 1978.

[9] R.J. McEliece, "A Public-Key Cryptosystem Based on Algebraic Coding Theory", DSN Progress Report 42-44, JPL Pasadena, pp. 114-116, 1978.