PRIVATE-KEY ALGEBRAIC-CODED CRYPTOSYSTEMS *

T. R. N. Rao
The Center for Advanced Computer Studies
University of Southwestern Louisiana
Lafayette, Louisiana 70504

Kil-Hyun Nam **
National Defense College
Seoul, Korea

ABSTRACT

Public-key cryptosystems using very large distance algebraic codes have been studied previously. Private-key cryptosystems using simpler codes have also been subject of some study recently. This paper proposes a new approach to the private-key cryptosystems which allows use of very simple codes such as distance 3 and 4 Hamming codes.

This new approach gives not only very efficient encoding/decoding and very high information rates but also appears to be secure even under chosen-plaintext attacks.

Keywords : cryptosystems, public-key cryptosystems, private-key cryptosystems, Algebraic codes, crypto-complexity, chosen-plaintext attack, Joint Encryption and Error-control Coding

* This research is supported by a grant from National Security Agency grant # MDA 904-84-H-005.

** This author's research was performed while he was with the Center for Advanced Computer Studies at University of Southwestern Louisiana.

1. INTRODUCTION

McEliece introduced a public-key cryptosystem based on algebraic coding theory using t-error correcting Goppa codes [McEliece '78]. But McEliece Public-key Cryptosystem (MPBC) requires large block lengths with capabilities to correct large number of errors ($n \approx 1000$ bits, $t \approx 50$ bits) to be effective. This involves very large computational (encryption and decryption) overhead to be practical in computer communications.

Private-key Algebraic-coded Cryptosystems (PRAC) were suggested by Rao [Rao '84b] using the same techniques as MPBC but keep the public generator matrix as private. PRAC provides better security with simpler error correcting codes, hence, requires relatively low computational overhead. However, we show that PRAC can be broken easily by a chosen-plaintext attack. Both MPBC and PRAC are classified as Algebraic-Coded Cryptosystems (ACC) here.

This paper introduces a new approach to PRAC, which requires simple error correcting codes (i.e. distance 3 codes) and also provides much higher security level.

1.1. McEliece Public-key Cryptosystems (MPBC)

Encryption

Let G be a t-error correcting k*n generator matrix of a linear code over GF(2) capable of t-error correction. The rate of the code is $\frac{k}{n}$. We can select a random k*k nonsingular matrix S called scrambler and a random n*n permutation matrix P. Having G, S and P, we can compute the public generator matrix G' such that G' = SGP, which is combinatorially equivalent to G.

Then the encryption is done by:

$C = MG' + Z$

where C : ciphertext of length n,

M : plaintext message of length k,

Z : random error vector of length n with weight t.

Note that the vectors are italic lettered, and weight means Hamming weight.

Decryption

The decryption is very straight forward.
From the encryption equation

$$\begin{aligned} G' &= SGP \\ C &= MG' + Z \\ &= MSGP + Z \\ &= M'\,GP + Z \quad \text{where } M' = MS \end{aligned}$$

Hence, we can recover M as given by the following steps.

Step 1 compute C' :

$$\begin{aligned} C' &= CP^{T} = M'\,G + ZP^{T} \\ &= M'\,G + Z' \qquad \text{where } Z' = ZP^{T} \end{aligned}$$

(Note: Z' has same weight as Z since
P and P^{T} are permutation matrices)

Step 2 Decoding and error correction:
(Patterson Algorithm [MCEL 77]).

Step 3 recover plaintext M:

$$M = M'\,S^{-1}$$

Cryptanalysis of MPBC

As suggested by McEliece in his paper [McEliece '78], there could be two kinds of basic attacks for the cryptanalyst to try.

(a) Factoring S, G and P from G'

Since the number of codes which are combinatorially equivalent to a given code is astronomical, it is hopeless task to find out exact keys S, G and P used for G'. However, the cryptanalyst needs only some

S_i, G_i and P_i such that $S_i G_i P_i = G'$ and G_i is t-error correcting code. For the given G', the cryptanalyst can obtain S_j, G_s and P_j satisfying the equation $S_j G_s P_j = G'$, where G_s is a generator in systematic form. G_s is obtained from G' by elementary row operations (row canonical reduction) and column operations. G', G_s and G are all said to be combinatorially equivalent. Where as G corresponds directly to a Goppa code which has well understood and well-known decoding algorithms, no such would be possible for G_s. Trial and error manipulation to obtain a G_s coinciding with an equivalent Alternant code generator would require an astronomically large work factor.

(b) Recovering M from C directly without keys

Another approach involves solving a set of k-unknowns from n simultaneous equations for all possible Z values.

Let M and C be a plaintext pair

$$\begin{aligned} M &= m_1 m_2 m_3 \dots m_k \\ C &= c_1 c_2 c_3 \dots c_k \dots c_n \\ Z &= z_1 z_2 z_3 \dots z_k \dots z_n \end{aligned}$$

$$G' = [\ G_{ij}'\ ], \quad i = 1, \dots, k$$
$$j = 1, \dots, n$$

(t-error correcting algebraic code)

Then, for j= 1, ... , n

$$\begin{aligned} c_1 &= m_1 G_{11}' + m_2 G_{21}' + \dots + m_k G_{k1}' + z_1 \\ c_2 &= m_1 G_{12}' + m_2 G_{22}' + \dots + m_k G_{k2}' + z_2 \\ &\vdots \\ c_n &= m_1 G_{1n}' + m_2 G_{2n}' + \dots + m_k G_{kn}' + z_n \end{aligned}$$

To solve k unknowns $(m_1, m_2, \cdots, m_k)$, k^3 operations are required because k equations are sufficient to solve the equations if the code is maximal distance separable (MDS) code. Otherwise, at most k' = n-d+1 equations are required to solve for k-unknowns [Pless '82].

Since t is smaller than n-k, it is possible that the cryptanalyst could select k equations containing no errors from n equations. Therefore, the cryptanalyst could repeat solving equations by selecting arbitrary k equations from n simultaneous equations with the assumption of no errors in selected equations until a meaningful plaintext is obtained.

The probability of no errors in k equations, P_k is:

$$P_k = \prod_{i=0}^{k-1} \left(1 - \frac{t}{n-i}\right)$$

and the average number of repetition is P_k^{-1}.

Hence, the average work factor, T is:

$$T = k^3 * P_k^{-1}$$

However, this does not include the work factor to check whether the plaintext M obtained by solving equations is correct (i.e., meaningful) or not. It is assumed that the plaintexts are from a source such as natural language or a programming language which contains an enormous amount of redundancy [Denning '82]. Redundancy in M helps to determine the validity of the plaintext derived.

1.2. Private-key Algebraic-coded Cryptosystems (PRAC)

To increase information rate and to reduce computational (encryption and decryption) overhead of MPBC, Private-key Algebraic-coded Cryptosystems (PRAC) were suggested [Rao '84b]. PRAC can provide better security with simpler error correcting codes, hence, require relatively low computational overhead compared to MPBC.

PRAC keeps G' private as well as S, P and G to provide higher security level. A known-plaintext attack to PRAC is feasible by solving matrices for each column vector of G' independently but this method requires a very large set of known (M,C) pairs. Hence, this attack can be foiled by periodic change or modification of the keys by the cryptographer. However, the analysis given below shows that PRAC still requires large t to be secure from a chosen-plaintext attack.

Chosen-Plaintext Attack

The cryptanalyst is required to go through two steps.

Step 1 : Solve for G' from a large set of (M,C) pairs.

Step 2 : Determine M from C using G' obtained in Step 1 (same work factor as in MPBC).

It can be safely assumed that a chosen plaintext of the form $M = (00\ldots010\ldots0)$ with only one 1 in i th position (for $i = 1, \ldots, k$) is not allowed by the cryptosystem. However, a chosen-plaintext attack may proceed as follows.

Let M_1 and M_2 are two plaintext differing in one position only, that is,

$$M_1 - M_2 = (00\ldots0\underset{\substack{| \\ i^{th}}}{1}0\ldots0) \quad \text{position for } i = 1, \ldots, k$$

then,

$$C_1 - C_2 = g_i' + (Z_1 - Z_2) \qquad \text{(Eq. 1)}$$

where g_i' is the i th row vector of G'.

The Hamming weight of $(Z_1 - Z_2)$ is at most 2t. Since t is much smaller than n, the majority of the bits of the vector $C_1 - C_2$ correspond directly with g_i'. We can let $C_1 - C_2$ represent one estimate of g_i'. By repeating the step several times a number of estimates of g_i' can be obtained. From these estimates of g_i' and by majority voting for each position, the vector g_i' can be correctly determined. This step repeated for all $i = 1,2,\ldots k$ will give us G', which can be used to break the code by step 2. This step 2 will require a relatively small work factor because t is small.

However, a chosen-plaintext attack of the above nature can succeed only when $\frac{t}{n}$ is small and it will not if $t \approx \frac{n}{2}$.

2. MODIFIED CRYPTOSYSTEMS

2.1. Introduction

Our intent here is to obtain private-key cryptosystems using simple algebraic codes such as Hamming codes or distance 5 BCH codes. Furthermore, we would still want the Z vector to have a weight t sufficiently large to provide good security. By a clever design we will show that we could obtain t $\approx \frac{n}{2}$. Obviously it would not be possible unless we change or modify the original encryption method.

Here we develop such a modification and show that it is indeed possible to use simple (i.e., short distance) algebraic codes for PRAC which are very secure from chosen-plaintext attacks. Clearly a system that is secure from such an attack is also secure from other attacks including known-plaintext attacks.

2.2. Encryption of Modified PRAC

This approach uses a minimum distance 3 code generator G (as an example) and uses specific error patterns for the random error vector Z of which the average Hamming weight is approximately $\frac{n}{2}$. Encryption method is modified as follow.

Let G' = SG

where S : k*k nonsingular matrix

G : k*n distance 3 code generator matrix

G': k*n encryption matrix

Then

$$C = (MG' + Z)P \qquad \text{(Eq. 2)}$$

where M : plaintext of length k

C : ciphertext of length n

P : n*n permutation matrix

Z : a random ATE (Method 1) or an entry of the Syndrome-error table (Method 2)

(Method 1 and 2 are described below.)

Since the security of PRAC crucially depends on the weight of Z, the selection of Z is very important. We introduce two kinds of error patterns.

Method 1 : Use adjacent t errors for Z.

Definition 1 : Adjacent t Errors (ATE)

An ATE is a vector of length n with t ($\leq \frac{n}{2}$) adjacent errors, i.e., an ATE consists of n-t 0's and t consecutive 1's. ATE must not be a codeword.

A random ATE can be used for Z. There exist exactly n-t+1 ATE's for the given n and t (and n ATE's for cyclic codes).

Method 2 : Use of predetermined set of vectors (Syndrome-error table).

A predetermined set of vectors consisting one from each coset of the standard array decoding table can be used for Z. Each coset has a distinct syndrome and there are exactly 2^{n-k} cosets [Blahut '83, Lin '83]. Therefore, we could select any set of vectors one from each of the 2^{n-k} cosets. The set is predetermined in the sense the decryptor knows the Syndrome-error table used for Z. Fig.1 shows an example of standard array and Syndrome-error table. The vectors in the rectangular boxes are selected as Z - vectors.

$$H = \begin{bmatrix} 0&1&1&1&0&0 \\ 1&0&1&0&1&0 \\ 1&1&0&0&0&1 \end{bmatrix}$$

Coset leader								Syndrome
000000	001110	010101	$\boxed{100011}$	011011	101101	110110	111000	000
000001	001111	010100	100010	$\boxed{011010}$	101100	110111	111001	001
000010	001100	010111	100001	011001	101111	$\boxed{110100}$	111010	010
000100	001010	010001	100111	011111	$\boxed{101001}$	110010	111100	100
001000	000110	$\boxed{011101}$	101011	010011	100101	111110	110000	110
010000	011110	000101	$\boxed{110011}$	001011	111101	100110	101000	101
100000	$\boxed{101110}$	110101	000011	111011	001101	010110	011000	011
001001	000111	011100	101010	010010	100100	111111	$\boxed{110001}$	111

Fig. 1. Standard array for the (6, 3, 3) code

G' and P are secret encryption keys, and the Syndrome-error table is also secret in the Method 2.

2.3. Decryption of Modified Cryptosystems

From the encryption algorithm (Eq. 2)

$$\begin{aligned} C &= (MG' + Z)P \\ &= MSGP + ZP \\ &= M'\,GP + ZP. \qquad (M' = MS) \end{aligned}$$

Decryption can be done using secret keys S^{-1}, H^T $(GH^T = 0)$ and P^T through following steps.

Step 1 Obtain C' :

$$C' = CP^T = M'\,G + Z$$

Step 2 : Find the error pattern and recover M' :

$$\begin{aligned} C'\,H^T &= M'\,GH^T + ZH^T \\ &= ZH^T \qquad \text{(Syndrome)} \end{aligned}$$

Identify the error pattern.

(use the Syndrome-error table look-up for the Method 2).

Recover M' by correcting for the error pattern.

Step 3 Recover plaintext M:

$$M = M'\,S^{-1}$$

Note : It appears that this approach requires long keys (S, P, G and the Syndrome-error table for the Method 2). However, the keys could be generated by using a pseudo-random number generator algorithm. In that case the user may require only short seeds for keys S, P and the Syndrome-error table. This problem is not addressed here and it would be a topic for future work.

2.4. Application to JOEEC

Recently Joint Encryption and Error-control Coding (JOEEC) was suggested [Rao '84a]. This approach combines data encryption and error-control coding steps into one step to gain speed and efficiency in implementation.

The modified cryptosystems could also be implemented as JOEEC by using higher distance codes. But the application to JOEEC of this approach is presently being studied.

3. CRYPTANALYSIS OF MODIFIED CRYPTOSYSTEMS

The encryption algorithm (Eq. 2) can be rewritten as follows.

$$\begin{aligned} C &= (M\mathrm{G}' + Z)\,\mathrm{P} \\ &= M\mathrm{G}'' + Z\mathrm{P} \end{aligned}$$

where $\mathrm{G}'' = \mathrm{G}'\mathrm{P} = [\ g_i''\ ]$ for $i = 1, \dots, k$,

and g_i'' is a row vector.

The following lemmas help us to establish the high level of security provided by this new approach.

Lemma 1 : The number of P's that transform ATE's into non-ATE's is at least $(n - \lfloor \frac{n}{t} \rfloor - 1)!$ if $2 < t \le \frac{n}{2}$, where n is the length of ATE and t is the length of adjacent errors.

Outline of Proof: Let vector V be an ATE of length n. We select a set of positions, $\{1, 2, t, 2t, \dots, bt\}$, from V where $b = \lfloor \frac{n}{t} \rfloor$. We reorder these positions as an ordered set, $B = \{1, t, 2t, \dots, bt, 2\}$. This mapping is illustrated in the figure below.

```
V =  |--+-------+-------+-------+--- - - - ----+------|        (ATE)
     1  2       t       2t      3t             bt     n        b = ⌊n/t⌋

B =  {1, t, 2t, . . . , bt, 2}

V' = |---- - - - ----+-----B-----+-------- - - - -------|      (non-ATE)
```

We consider a permutation map of vector V to V' with B embedded in V'. The purpose is to make V' a non-ATE. This is achieved because B contains at least one '1' separated by '0's. Strictly B could start from any position of V' and therefore, we have n-b-1 choices. In addition the number of permutations possible for V —> V' of the remaining positions is (n-b-2)!. Thus the total number of permutations of transforming an ATE vector V to non-ATE vector V', N_p can be shown to be at least

$$N_p = (n\text{-}b\text{-}1) * (n\text{-}b\text{-}2)!$$
$$= (n\text{-}b\text{-}1)! \qquad \text{QED.}$$

This formula gives a lower bound for N_p of (n - 3)! when $t = \lfloor \frac{n}{2} \rfloor$.

Lemma 2 : The number of code generators combinatorially equivalent to a (n, k, 3) code generator is at least k!.

Proof: Let G be a (n, k, 3) code generator in systematic form.

$$G = [I_k \; P_{k,n-k}]$$

where I_k is an identity matrix and

$P_{k,n-k}$ is a parity check matrix.

Then, there are k! row combinations of parity check matrix, which are distinct (n, k, 3) code generators also. All of these code generators can be obtained by row exchange and column permutation of G, and hence, are combinatorially equivalent to G [Peterson '72].

Lemma 3: The number of k*k non-singular matrices over GF(2), N_s is given by

$$N_s = \prod_{i=0}^{k-1} (2^k - 2^i) > 2^{k^2-k} \qquad \text{(Eq. 3)}$$

Proof: We can start with any non-zero vector for the first row of non-singular matrix S and we have $2^k - 1$ choices. The second row must be linearly independent of the first. That is we have $2^k - 2$ choices for the second row. For the third row the choice is any vector linearly independent of the first two. Clearly it has $(2^k - 2^2)$ choices. Continuing this way, the number of non-singular matrices are given by the equality (Eq. 3). Since there are k terms in the product, the smallest of which is 2^{k-1}, the inequality is easily proved.

An attack by exhaustive search for S, G and P is considered hopeless task due to the results of above Lemmas. The previously described method of the chosen-plaintext attack (described in Section 1.2.1) can not be applied here because the average Hamming weight of $(Z_1 - Z_2)P$ is about $\frac{n}{2}$, which is very large. Therefore, we have to look for a different method to cryptanalysis and it could be as follows.

Let C_j and C_k be two distinct ciphertexts obtained for the same plaintext M. Then

$$\begin{aligned} C_j &= MG'' + Z_j P \\ C_k &= MG'' + Z_k P \\ C_j - C_k &= (Z_j - Z_k)P \end{aligned}$$

The above step provides one value for $(Z_j - Z_k)P$. This step needs to be repeated until all possible pairs of Z's are used. The number of distinct Z's is given by

$$\begin{aligned} N &= \frac{n}{2} \quad \text{for the Method 1,} \\ &\geq n \quad \text{for the Method 2;} \end{aligned}$$

and the number of possible distinct values of $(Z_i - Z_j)P$ is $\frac{N^2 - N}{2}$.

An expression for g_i'' by a computation as described in Section 1.2.1 is given by

$$\begin{aligned} C_1 - C_2 &= g_i'' + (Z_1 - Z_2)P \\ g_i'' &= C_1 - C_2 - (Z_1 - Z_2)P. \qquad \text{(Eq. 4)} \end{aligned}$$

Hence, every possible value of $(Z_i - Z_j)P$ should be tested for $(Z_1 - Z_2)P$ of Eq. 4. Since the correctness of each row vector of G", g_i, can not be verified independently, the complete solution of G" should be obtained and verified.
This involves on the average work factor, T given by

$$T \geq \frac{1}{2}\left(\frac{N^2}{2}\right)^k.$$

Substituting for N, T can be shown to be $\Omega\ (n^{2k})$. Thus we establish the following.

Claim : To determine G" from a chosen- plaintext attack (as discussed above) has a work factor T = Ω (n^{2k}).

It can be easily shown that the above step, namely, the determination of G" is the really dominant factor. Determination of P and Z vectors are straight forward after that. As of now, the analysis and procedure explained seems to be the only possible approach to break the code and it requires an enormous work factor Ω (n^{2k}).

4. CONCLUSION

We have introduced a new approach to the private-key algebraic-coded cryptosystems requiring only simple codes such as distance 3 codes. These systems will be very efficient because of high information rates and low overhead for encoding and decoding logic. The chosen-plaintext attack given here appears to be the only plausible approach for cryptanalyst.

It requires a work factor Ω (n^{2k}) and is therefore, computationally secure even for small $k \approx 50$. It will be a challenge to find alternate methods of attack which can be successful.

REFERENCES

[Blahut '83] Richard E. Blahut, *Theory and Implementation of Error Correcting Code,* Addison-Wesley, 1983.

[Denning '82] Dorothy E. Denning, *Cryptography and Data Security,* Addison Wesley, 1982.

[Lin '83] Shu Lin, Daniel J. Costello, Jr., *Error Control Coding: Fundamentals and Applications,* Prentice-Hall, 1983.

[McEliece '77] McEliece R. J. "The theory of Information and coding," *(vol. 3 of the encyclopedia of mathematics and its Applications)* Reading, Mass Addison-Wesley, 1977.

[McEliece '78] R.J. McEliece, "A Public-Key Cryptosystem Based on Algebraic Coding Theory," *DSN Progress Report,* Jet Propulsion Laboratory, CA., Jan. & Feb. 1978, pp 42 - 44.

[Peterson '72] W. Wesley Peterson and E. J. Weldon, Jr., *Error-Correcting Codes,* Second edition, The MIT Press, 1972.

[Rao '84a] T.R.N. Rao, "Joint Encryption and Error Correction Schemes," *Proc. 11th Intl. Symp. on Comp. Arch.,* Ann Arbor, Mich., May 1984.

[Rao '84b] T.R.N. Rao, "Cryptosystems Using Algebraic Codes," *Intl. Conf. on Computer Systems & Signal Processing,* Bangalore, India, Dec. 1984.