

# On Private-Key Cryptosystems Based on Product Codes

Hung-Min Sun<sup>1</sup> and Shiu-Pyng Shieh<sup>2</sup>

<sup>1</sup> Department of Information Management, Chaoyang University of Technology,  
Wufeng, Taichung County, Taiwan 413  
Email: hmsun@mail.cyut.edu.tw

<sup>2</sup> Department of Computer Science and Information Engineering,  
National Chiao Tung University, Hsinchu, Taiwan 30050  
ssp@csie.nctu.edu.tw

**Abstract.** Recently J. and R.M. Campello de Souza proposed a private-key encryption scheme based on the product codes with the capability of correcting a special type of structured errors. In this paper, we show that J. and R.M. Campello de Souza's scheme is insecure against chosen-plaintext attacks, and consequently propose a secure modified scheme.

## 1 Introduction

In 1978, McEliece [1] proposed a public-key cryptosystem based on algebraic coding theory. The idea of the cryptosystem is based on the fact that the decoding problem of a general linear code is an NP-complete problem. Compared with other public-key cryptosystems [2,3], McEliece's scheme has the advantage of high-speed encryption and decryption. However, the scheme is subjected to some weaknesses [4,5]. Rao and Nam [6,7] modified McEliece's scheme to construct a private-key algebraic-code cryptosystem which allows the use of simpler codes. The Rao-Nam system is still subjected to some chosen-plaintext attacks [7-10], and therefore is also insecure. Many modifications to Rao-Nam private-key cryptosystem were proposed [7,11,12]. These schemes are based on either allowing nonlinear codes or modifying the set of allowed error patterns. The Rao-Nam scheme using Preparata codes [7,11] was proven to be insecure against a chosen-plaintext attack [12]. The Rao-Nam scheme using burst-error-correcting codes, such as Reed-Solomon codes [6], was also proven to be insecure [8]. The use of burst-error-correcting codes for private-key encryption was generalized elsewhere [13,14]. The idea of these two cryptosystems is based on the fact that the burst-correcting capacity of a binary linear block burst-error-correcting code is, in general, larger than its random error-correcting capacity. Sun *et al.* and Al Jabri [15,16] showed that these two schemes are insecure against chosen-

plaintext attacks. In 1995, J. and R.M. Campello de Souza [17] proposed a private-key encryption scheme based on product codes (CDS scheme in short). The idea of their scheme is to use a product code which is capable of correcting a special kind of structured errors and then disguise it as a code that is only linear. This makes it unable to correct the errors as well as their permuted versions.

In this paper, we first show CDS scheme is still insecure against chosen-plaintext attacks. And then we propose a modified private-key cryptosystem based on product codes, which is secure against the chosen-plaintext attacks proposed to break CDS and other similar schemes.

This paper is organized as follows. In section 2, we give some basic preliminaries. Sections 3 and 4, respectively, introduce CDS scheme, and analyze the security of CDS scheme. In section 5, we propose a modified private-key cryptosystem based on product codes, and analyze the security of the proposed scheme. In section 6, we discuss the long-key problem in our scheme. Finally, we conclude the paper in section 7.

## 2 Preliminaries

**Definition 1:** The direct mapping with parameters  $r$  and  $s$ , denoted  $DM_{r,s}(\cdot)$ , is the

one that maps the vector  $V = (v_1, \dots, v_{rs})$  into the matrix  $A = \begin{bmatrix} a_{0,0} & \dots & a_{0,s-1} \\ \cdot & \cdot & \cdot \\ a_{r-1,0} & \dots & a_{r-1,s-1} \end{bmatrix}_{r \times s}$  so

that  $a_{i,j} = v_{is+j+1}$ , for  $i=0, 1, \dots, r-1, j=0, 1, \dots, s-1$ .

**Definition 2:** The vector  $E = (e_1, \dots, e_{rs})$ ,  $e_i \in$  the congruence class modulo  $q$  where  $q$  is a positive integer, is said to be a biseparable error, denoted  $BSE_q(r, s)$  if (i) its nonzero components are nonzero distinct elements in the congruence class modulo  $q$  and (ii) each row and each column of  $DM_{r,s}(E)$  contains, at most, one nonzero component.

Note that the maximum weight of a  $BSE_q(r, s)$  is  $w_{\max} = \min(q-1, \min(r, s))$  and the number of  $BSE_q$ 's with a given weight  $w$  is

$$N_{BSE_q(r,s)}(w) = C_w^{q-1} \prod_{i=1}^w (r+1-i)(s+1-i).$$

**Example 1:** Let  $q = 5$ ,  $r = 3$ ,  $s = 4$ ,  $E = (0, 0, 1, 0, 4, 0, 0, 0, 0, 0, 0, 3)$ .

Because  $DM_{r,s}(E) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}_{3 \times 4}$ ,  $E$  is a biseparable error over  $GF(5)$ .

**Theorem 1:** A product code  $PC(n = (r + 1)(s + 1), k = rs, d = 4)$  over  $GF(q)$ , whose constituent row and column codes are single parity-check codes  $C_1(n_1 = r + 1, k_1 = r, d_1 = 2)$  and  $C_2(n_2 = s + 1, k_2 = s, d_2 = 2)$  respectively, can correct a  $BSE_q(r + 1, s + 1)$  of weight up to  $w_{\max}$ .

**Proof:** We assume that there exactly exists an error vector  $E$  which is a  $BSE_q(r + 1, s + 1)$  in the received word,  $R$ , and the  $(i, j)$ -th entry of  $DM_{r+1,s+1}(E)$  is a nonzero element of  $GF(q)$ , say  $e$ . Because no other error will be found in the  $i$ -th row of  $DM_{r+1,s+1}(E)$ , we can detect an error  $e$  contained in the  $i$ -th row of  $DM_{r+1,s+1}(R)$  by the single parity check codes  $C_1$ . Similarly, we can detect an error  $e$  contained in the  $j$ -th column of  $DM_{r+1,s+1}(R)$  by the single parity check codes  $C_2$ . Because the error  $e$  in  $E$  is unique, the error  $e$  in the  $(i, j)$ -th entry of  $DM_{r+1,s+1}(R)$  can be identified.

Note that product codes still work well in the congruence class modulo  $m$ , where  $m$  is not a prime. Total number of the codes defined in Theorem 1 is  $NC = (r + 1) \cdot (s + 1) \cdot (r \cdot s)!$ . In the following, we give an example of  $PC(n = 3 \cdot 4, k = 2 \cdot 3, d = 4)$  over  $GF(5)$ .

**Example 2:** We assume that the information word is  $M = (m_1, m_2, m_3, m_4, m_5, m_6)$ , where  $m_i \in GF(5)$ . The encoding rule is denoted by the matrix,

$$\begin{bmatrix} m_2 & p_1 & m_4 & m_6 \\ p_3 & p_6 & p_4 & p_5 \\ m_5 & p_2 & m_1 & m_3 \end{bmatrix}_{3 \times 4}, \text{ where } p_i \text{'s } (1 \leq i \leq 6) \text{ satisfy the following equations:}$$

$$p_1 + m_2 + m_4 + m_6 = 0 \pmod{5},$$

$$p_2 + m_5 + m_1 + m_3 = 0 \pmod{5},$$

$$p_3 + m_2 + m_5 = 0 \pmod{5},$$

$$p_4 + m_4 + m_1 = 0 \pmod{5},$$

$$p_5 + m_6 + m_3 = 0 \pmod{5}, \text{ and}$$

$$p_6 + p_1 + p_2 = 0 \pmod{5}.$$

Note that  $p_6 + p_3 + p_4 + p_5 = 0 \pmod{5}$  also holds if the above equations hold. The codeword is  $C = (m_2, p_1, m_4, m_6, p_3, p_6, p_4, p_5, m_5, p_2, m_1, m_3)$ . We can also use the concept of  $C = M G$  to denote the encoding procedure, where the generator matrix of the code  $G =$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 4 & 0 & 0 & 4 & 1 & 0 \\ 1 & 4 & 0 & 0 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 4 & 0 & 4 & 0 & 1 \\ 0 & 4 & 1 & 0 & 0 & 1 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 1 & 0 & 0 & 1 & 4 & 0 & 0 \\ 0 & 4 & 0 & 1 & 0 & 1 & 0 & 4 & 0 & 0 & 0 & 0 \end{bmatrix}_{6 \times 12}$$

Assume  $M = (3, 2, 0, 1, 4, 2)$  and  $E = (0, 0, 1, 0, 4, 0, 0, 0, 0, 0, 0, 3)$ . From Example 1, we know that  $E$  is a  $BSE_5(3, 4)$ . Thus the received word is

$$\begin{aligned} R &= M G + E = (2, 0, 1, 2, 4, 2, 1, 3, 4, 3, 3, 0) + (0, 0, 1, 0, 4, 0, 0, 0, 0, 0, 0, 3) \\ &= (2, 0, 2, 2, 3, 2, 1, 3, 4, 3, 3, 3). \end{aligned}$$

The decoding procedure need only put  $R$  into the matrix  $DM_{3,4}(R)$ , find the errors by checking these rows and columns, and remove the errors. The procedure of finding errors is demonstrated in the following.

$$\text{From } R, \text{ we know that } DM_{3,4}(R) = \begin{bmatrix} 2 & 0 & 2 & 2 \\ 3 & 2 & 1 & 3 \\ 4 & 3 & 3 & 3 \end{bmatrix}_{3 \times 4}.$$

By checking these parity-check bits, we find that the first row contains one error 1, the second row contains one error 4, the third row contains one error 3, the first column contains one error 4, the third column contains one error 1, and the fourth column contains one error 3. It is clear that the error vector (assumed to be a  $BSE_5(3, 4)$ ) is  $(0, 0, 1, 0, 4, 0, 0, 0, 0, 0, 0, 3)$ . Therefore, the errors can be found and removed.

### 3 Campello de Souza's Scheme

In this section, we introduce the private-key encryption proposed by J. and R.M. Campello de Souza. Their scheme is based on product codes introduced in the above section.

**Secret key:**  $G$  is the generator matrix of a  $PC(n = (r+1)(s+1), k = rs, d=4)$  over  $GF(q)$ ,  $S$  is a random  $k \times k$  nonsingular matrix over  $GF(q)$ , called the scrambling matrix, and  $P$  is an  $n \times n$  permutation matrix.

**Encryption:** Let the plaintext  $M$  be a  $q$ -ary  $k$ -tuple. That is,  $M = (m_1, \dots, m_k)$ , where  $m_i \in GF(q)$ . The ciphertext  $C$  is calculated by the sender:

$$C = (MSG + E_{(r+1)(s+1),w})P, \text{ where } E_{(r+1)(s+1),w} \text{ is a } BSE_q(r+1, s+1) \text{ of weight } w.$$

**Decryption:** The receiver first calculates  $C' = C P^{-1} = M'G + E_{(r+1)(s+1),w}$ , where  $M' = MS$  and  $P^{-1}$  is the inverse of  $P$ . Then the receiver removes the errors embedded in  $C'$  to obtain  $M'$ . At last, the receiver recovers  $M$  by computing  $M = M' S^{-1}$ .

The encryption algorithm can be rewritten as  $C = (MSG + E_{(r+1)(s+1),w})P = MG' + E'_{(r+1)(s+1),w}$  where  $G' = SGP$  and  $E'_{(r+1)(s+1),w} = E_{(r+1)(s+1),w}P$ . The matrix  $G'$  can be found by a type of attack called the Majority Voting Attack suggested in [7,9,14]. We state this attack in the following. The cryptanalyst chooses a plaintext of the form  $M_i$  with only one 1 in the  $i$ th position for  $i = 1, \dots, k$ . He encrypts  $M_i$  a number of times and obtains an estimate of  $g'_i$ , the  $i$ th row of the matrix  $G'$ , with a desired degree of certainty. Repeating this step for  $i = 1, \dots, k$  gives  $G'$ . Note that one who knows  $G'$  cannot recover  $M$  because  $E'_{(r+1)(s+1),w}$  is the permuted version of  $E_{(r+1)(s+1),w}$ .

The work factor for breaking this system is related with the number of product codes. The total number of the product codes  $PC(n = (r+1)(s+1), k = rs, d=4)$  is  $NC = (r+1) \cdot (s+1) \cdot (r \cdot s)!$ . With  $G' = SGP$ , the cryptanalyst must find, among all  $NC$  matrices, one of the  $(r+1)! \cdot (s+1)!$  matrices that can be used to decode the corrupted received words (ciphertext). This means that the work factor is the size of  $(r \cdot s)! / (r! \cdot s!)$ .

## 4 Cryptanalysis of CDS Scheme

In this section, we show that the errors embedded in the ciphertext can be removed without knowledge of the permutation matrix  $P$  in CDS scheme. Thus  $M$  can be computed by  $M = C(G')^{-1}$ , where  $(G')^{-1}$  is the inverse of  $G'$ . Therefore, CDS scheme is not secure.

Because  $C = MG' + E'_{(r+1)(s+1),w}$  and  $G'$  can be known from the analysis in section 3, we can easily collect error patterns  $E'_{(r+1)(s+1),w}$  as follows. Given a pair of plaintext and ciphertext,  $(M, C)$ , an error pattern  $E'_{(r+1)(s+1),w}$  can be computed by  $E'_{(r+1)(s+1),w} = C - MG'$ . We assume that  $E_{(r+1)(s+1),w} = \langle e_1, e_2, \dots, e_i, \dots, e_n \rangle$  and  $E'_{(r+1)(s+1),w} = \langle e_1, e_2, \dots, e_i, \dots, e_n \rangle$ . Because  $E_{(r+1)(s+1),w}P = E'_{(r+1)(s+1),w}$  where  $P$  is a permutation matrix, we write

$$\begin{aligned} E_{(r+1)(s+1),w} P &= \langle e_1, e_2, \dots, e_i, \dots, e_n \rangle P \\ &= \langle e_{\tau(1)}, e_{\tau(2)}, \dots, e_{\tau(i)}, \dots, e_{\tau(n)} \rangle \\ &= \langle e_1', e_2', \dots, e_i', \dots, e_n' \rangle, \text{ where } \tau(\cdot) \text{ is a one-to-one and onto function from } \{1, 2, \dots, n\} \text{ to itself.} \end{aligned}$$

We define the checking set of  $e_i'$  is the set  $CS(e_i') = \{ e_i' \} \cup \{ e_j' \mid e_{\tau(i)}$  and  $e_{\tau(j)}$  are in the same row or same column of  $DM_{r+1,s+1}(E_{(r+1)(s+1),w}) \}$  and the complement of the set  $CS(e_i')$  is the set  $\overline{CS(e_i')} = \{ e_j' \mid e_{\tau(i)}$  and  $e_{\tau(j)}$  are neither in the same row nor in the same column of  $DM_{r+1,s+1}(E_{(r+1)(s+1),w}) \}$ . Note that each  $CS(e_i')$  has  $r+s+1$  elements and each  $\overline{CS(e_i')}$  has  $rs$  elements, i.e.,  $|CS(e_i')| = r+s+1$  and  $|\overline{CS(e_i')}| = rs$ . For an error pattern  $E'_{(r+1)(s+1),w} = \langle e_1', e_2', \dots, e_i', \dots, e_n' \rangle$ , if  $e_i' \neq 0$  and  $e_j' \neq 0$ , we say that there exists a relation between  $e_i'$  and  $e_j'$ . It is clear that if  $e_i' \neq 0$  and  $e_j' \neq 0$ , then  $e_i' \in \overline{CS(e_j')}$  (i.e.,  $e_i' \notin CS(e_j')$ ) and  $e_j' \in \overline{CS(e_i')}$  (i.e.,  $e_j' \notin CS(e_i')$ ). Therefore, from an error pattern  $E'_{(r+1)(s+1),w}$  with weight  $w$ , we can obtain  $\binom{w}{2} = \frac{w(w-1)}{2}$  pairs of relations between  $e_i'$  and  $e_j'$ . We assume that each  $E'_{(r+1)(s+1),w}$  gives us at least one relation between  $e_i'$  and  $e_j'$ . Because the total number of relations is only  $(r+1)(s+1)rs/2$ , the *expected* average number of pairs  $(M, C)$  required to collect *all* the sets  $\overline{CS(e_i')}$ ,  $1 \leq i \leq n$ , is about  $((r+1)(s+1)rs/2) \cdot \log((r+1)(s+1)rs/2)$  [18]. Once all the sets  $\overline{CS(e_i')}$  are obtained,  $CS(e_i')$  can be determined. In the following, we give an example to show the sets  $CS(e_i')$  and demonstrate how these sets can be used to remove the errors.

### Example 3:

Let  $E_{(2+1)(3+1),w} = \langle e_1, e_2, \dots, e_i, \dots, e_{12} \rangle$  and hence

$$DM_{3,4}(E_{(2+1)(3+1),w}) = \begin{bmatrix} e_1 & e_2 & e_3 & e_4 \\ e_5 & e_6 & e_7 & e_8 \\ e_9 & e_{10} & e_{11} & e_{12} \end{bmatrix}.$$

Assume  $P$  is a permutation matrix such that

$$\begin{aligned} & E_{(2+1)(3+1),w} P \\ &= \langle e_1, e_2, \dots, e_i, \dots, e_{12} \rangle P \\ &= \langle e_{\tau(1)}, e_{\tau(2)}, \dots, e_{\tau(i)}, \dots, e_{\tau(12)} \rangle \\ &= \langle e_8, e_6, e_{10}, e_2, e_3, e_{11}, e_1, e_9, e_4, e_{12}, e_5, e_7 \rangle \\ &= \langle e_1', e_2', e_3', e_4', e_5', e_6', e_7', e_8', e_9', e_{10}', e_{11}', e_{12}' \rangle \end{aligned}$$

We assume that the sets  $CS(e_i')$ 's have been determined as follows:

$$\begin{aligned} CS(e_1') &= \{ e_1', e_2', e_9', e_{10}', e_{11}', e_{12}' \}, CS(e_2') = \{ e_1', e_2', e_3', e_4', e_{11}', e_{12}' \}, \\ CS(e_3') &= \{ e_2', e_3', e_4', e_6', e_8', e_{10}' \}, CS(e_4') = \{ e_2', e_3', e_4', e_5', e_7', e_9' \}, \end{aligned}$$

$$\begin{aligned}
CS(e_5') &= \{e_4', e_5', e_6', e_7', e_9', e_{12}'\}, CS(e_6') = \{e_3', e_5', e_6', e_8', e_{10}', e_{12}'\}, \\
CS(e_7') &= \{e_4', e_5', e_7', e_8', e_9', e_{11}'\}, CS(e_8') = \{e_3', e_6', e_7', e_8', e_{10}', e_{11}'\}, \\
CS(e_9') &= \{e_1', e_4', e_5', e_7', e_9', e_{10}'\}, CS(e_{10}') = \{e_1', e_3', e_6', e_8', e_9', e_{10}'\}, \\
CS(e_{11}') &= \{e_1', e_2', e_7', e_8', e_{11}', e_{12}'\}, CS(e_{12}') = \{e_1', e_2', e_5', e_6', e_{11}', e_{12}'\}.
\end{aligned}$$

It is clear that if  $CS(e_i') \cap CS(e_j') \neq \emptyset$ ,  $CS(e_i') \cap CS(e_j')$  denotes the set of elements whose original positions in  $DM_{3,4}(E_{(2+1)(3+1),w})$  are in the same row (or column). By observing the set  $CS(e_i') \cap CS(e_j')$ , we can find a parity-checking rule in the ciphertext  $C$ . Let  $C = (c_1, c_2, \dots, c_n)$ . Therefore, we can check if there is any error embedded in those  $c_i$  with respect to the set  $CS(e_i') \cap CS(e_j')$ . For example, if  $CS(e_1') \cap CS(e_2') = \{e_1', e_2', e_{11}', e_{12}'\}$ , we can check if there is any error embedded in the set  $\{c_1, c_2, c_{11}, c_{12}\}$  by checking whether the value of  $c_1 + c_2 + c_{11} + c_{12} \pmod{q}$  is 0. If  $c_1 + c_2 + c_{11} + c_{12} = 0 \pmod{q}$ , no error is embedded in the set  $\{c_1, c_2, c_{11}, c_{12}\}$ . If  $c_1 + c_2 + c_{11} + c_{12} = a \pmod{q}$ , where  $a \neq 0$ , there exists an error  $a$  embedded in the set  $\{c_1, c_2, c_{11}, c_{12}\}$ . Note that the set  $\{e_1', e_2', e_{11}', e_{12}'\}$  can be also obtained from any of the following intersections:  $CS(e_2') \cap CS(e_{11}')$ ,  $CS(e_{11}') \cap CS(e_{12}')$ ,  $CS(e_1') \cap CS(e_{11}')$ ,  $CS(e_2') \cap CS(e_{12}')$ , or  $CS(e_1') \cap CS(e_{12}')$ .

By comparing all pairs of  $CS(e_i')$  and  $CS(e_j')$ , we can obtain other 6 parity-checking rules as follows:

$$\begin{aligned}
c_4 + c_5 + c_7 + c_9 &= 0 \pmod{q}, \\
c_3 + c_6 + c_8 + c_{10} &= 0 \pmod{q}, \\
c_7 + c_8 + c_{11} &= 0 \pmod{q}, \\
c_2 + c_3 + c_4 &= 0 \pmod{q}, \\
c_5 + c_6 + c_{12} &= 0 \pmod{q}, \\
c_1 + c_9 + c_{10} &= 0 \pmod{q}.
\end{aligned}$$

By these parity-checking rules, the errors can be identified and removed without knowledge of the permutation matrix  $P$ . Note that these parity-checking rules are equivalent to those in the product code with the generator matrix  $G$ .

In Table 1, we show the number of pairs  $(M, C)$  required to collect all sets  $CS(e_i')$  for different product codes, i.e., the work factor to break CDS system. The parameter  $q$  is not given because the value  $q$  don't influence the values of other parameters listed here. Note that larger  $q$  provides longer plaintext and ciphertext (the information rate is unchanged), and more number of biseparable errors which can be selected to be embedded in the ciphertext.

**Table 1. Number of pairs required to collect all sets  $CS(e_i')$  for different product codes**

$r$	$s$	$n=(r+1)(s+1)$	$k=rs$	Info. Rate.	$NC$	Work Factor (# of Pairs needed to collect all $CS(e_i')$ )
5	5	36	25	0.694	$2^{89}$	2749
6	6	49	36	0.735	$2^{144}$	5982
7	7	64	49	0.766	$2^{215}$	11537
8	8	81	64	0.790	$2^{302}$	20374
9	9	100	81	0.810	$2^{408}$	33641
10	10	121	100	0.826	$2^{532}$	52682

## 5 A Modified Private-Key Cryptosystem Based on Product Codes

From the cryptanalysis of CDS scheme in section 4, we know that the major weakness of CDS scheme is the knowledge of  $G'$ . Therefore, the possible method to repair CDS scheme is to protect  $G'$  from leakage. In the following, we propose a modified private-key cryptosystem based on product codes. Because the product codes work well in the congruence class modulo  $m$  where  $m$  is not a prime, we use a product code in the congruence class modulo  $2^l$  in our scheme.

**Secret key:**  $G$  is the generator matrix of a PC( $n = (r + 1)(s + 1)$ ),  $k = rs$ ,  $d = 4$ ) in the congruence class modulo  $m$ , where  $m = 2^l$ .  $S$  is a random binary  $(nt) \times (kt)$  matrix, called the scrambling matrix, and  $P$  is an  $(nt) \times (nt)$  permutation matrix.

**Encryption:** Let the plaintext  $M$  be a binary  $kt$ -tuple. That is,  $M = (m_1, \dots, m_{kt})$ , where  $m_i \in GF(2)$ . The ciphertext  $C$  is calculated by the sender:

$C = \{[M \oplus (E_{(r+1)(s+1),w} \otimes S)] \cdot G + E_{(r+1)(s+1),w}\} \otimes P$ , where  $E_{(r+1)(s+1),w}$  is a  $BSE_m(r+1, s+1)$  of weight  $w$ ,  $1 \leq w \leq \min(m-1, \min(r, s))$ . Here we use  $\oplus$  and  $\otimes$  to denote the XOR operation and the matrix multiplication operation in  $GF(2)$ , and  $+$  and  $\cdot$  to denote the vector addition operation and the matrix multiplication operation in the congruence class modulo  $m$ . When  $\oplus$  and  $\otimes$  operations are executed, every bit is regarded as a number in  $GF(2)$ . When  $+$  and  $\cdot$  operations are executed, every  $t$ -bits is regarded as a number in the congruence class modulo  $m$ .

**Decryption:** The receiver first calculates  $C' = C \otimes P^{-1} = M' \cdot G + E_{(r+1)(s+1),w}$ , where  $M' = [M \oplus (E_{(r+1)(s+1),w} \otimes S)]$  and  $P^{-1}$  is the inverse of  $P$ . Secondly, by using the decoding algorithm of the product code  $G$ , the receiver can find and remove the error  $E_{(r+1)(s+1),w}$  embedded in  $C'$  to obtain  $M'$ . At last, the receiver recovers  $M$  by computing  $M' \oplus (E_{(r+1)(s+1),w} \otimes S) = M$ .

The information rate of this scheme is  $\frac{k}{n} = \frac{rs}{(r+1)(s+1)}$ . As an example, we use

the following parameters to construct the system:  $t=3$ ,  $m=8$ ,  $r=6$ ,  $s=6$ ,  $n=49$ , and  $k=36$ . In this case, the length of the plaintext is 108 bits, the length of the ciphertext is 147 bits, hence the information rate is about 0.735. The total number of possible codes which may be chosen is  $NC = 49 \cdot (36!) \approx 2^{144}$ . Note that larger  $r$  and  $s$  in this scheme provide higher security and information rate. Basically, this scheme has the same parameters as those in CDS scheme except  $q = m = 2^t$ . Therefore some suggestions about  $r$ ,  $s$ ,  $n$ ,  $k$ , the information rate, and the total number of product codes  $NC$  are referred to Table 1.

### Security Analysis:

Here we remind that the distributive law for the operations  $+$  and  $\otimes$  doesn't hold. That is  $(a+b)\otimes c \neq a\otimes c + b\otimes c$ , where  $+$  denotes the addition operation in the congruence class modulo  $m$  ( $m \neq 2$ ) and  $\otimes$  denotes the multiplication operation in  $GF(2)$ . As an example, let  $m=8$ ,  $a=111_2=7_8$ ,  $b=110_2=6_8$ , and  $c=100_2$ . It is clear that  $(a+b)\otimes c = (7+6 \bmod 8) \otimes 100_2 = 101_2 \otimes 100_2 = \langle 1,0,1 \rangle \otimes \langle 1,0,0 \rangle = 1_2 = 001_2$ . However,  $a\otimes c + b\otimes c = 111_2 \otimes 100_2 + 110_2 \otimes 100_2 = \langle 1,1,1 \rangle \otimes \langle 1,0,0 \rangle + \langle 1,1,0 \rangle \otimes \langle 1,0,0 \rangle = 1_2 + 1_2 \pmod{8} = 010_2$ . Thus  $(a+b)\otimes c \neq a\otimes c + b\otimes c$ . Note that this inequality still holds when  $a$ ,  $b$  are vectors, and  $c$  is a matrix. Therefore,  $C = \{[M \oplus (E_{(r+1)(s+1),w} \otimes S)] \cdot G + E_{(r+1)(s+1),w}\} \otimes P \neq [M \oplus (E_{(r+1)(s+1),w} \otimes S)] \cdot G \otimes P + E_{(r+1)(s+1),w} \otimes P$ . On the other hand, the distributive law for the operations  $\oplus$  and  $\cdot$  doesn't hold, either. That is  $(a\oplus b) \cdot c \neq (a \cdot c) \oplus (b \cdot c)$ , where  $\oplus$  denotes the XOR operation and  $\cdot$  denotes the multiplication operation in the congruence class modulo  $m$  ( $m \neq 2$ ). As an example, let  $m=8$ ,  $a=011_2$ ,  $b=110_2$ , and  $c=011_2$ . It is clear that  $(a\oplus b) \cdot c = 101_2 \cdot 011_2 = 5 \cdot 3 \bmod 8 = 7 = 111_2$ . However,  $(a \cdot c) \oplus (b \cdot c) = (011_2 \cdot 011_2) \oplus (110_2 \cdot 011_2) = (3 \cdot 3 \bmod 8) \oplus (6 \cdot 3 \bmod 8) = 001_2 \oplus 010_2 = 011_2$ . Hence  $(a\oplus b) \cdot c \neq (a \cdot c) \oplus (b \cdot c)$ . This inequality still holds when  $a$ ,  $b$  are vectors, and  $c$  is a matrix. Therefore,  $[M \oplus (E_{(r+1)(s+1),w} \otimes S)] \cdot G \neq (M \cdot G) \oplus [(E_{(r+1)(s+1),w} \otimes S) \cdot G]$ .

These properties of the operations suggest that the encryption function,  $C = \{[M \oplus (E_{(r+1)(s+1),w} \otimes S)] \cdot G + E_{(r+1)(s+1),w}\} \otimes P$ , is unable to be reduced to a simpler form. Especially, the encryption function cannot be rewritten in the form:  $C = F_1(M) + F_2(E)$ , where  $F_1$  and  $F_2$  are two mapping functions with inputs  $M$  (the plaintext) and  $E$  (the added error), respectively. It is remarked that an encryption function which can be reduced into this form may be insecure, especially, in the case when  $F_1$  is linear and the number of possible added errors is small. So far most algebraic-code cryptosystems as introduced in Section 1 belong to this type. These schemes are vulnerable to the chosen-plaintext attacks.

Because it is difficult to distinguish the embedded error from the ciphertext and to reduce the encryption algorithm, our scheme is secure against the similar attacks used to attack CDS and other well-known private-key algebraic-code schemes. Further

research can be done on systematically analyzing the structure of this new scheme or finding efficient attacks on this scheme.

## 6 Long-Key Problem

All algebraic-code cryptosystems have the same problem that the used keys are too long. For example, McEliece's scheme [1] suggested the use of a  $(524 \times 524)$  nonsingular matrix over  $GF(2)$ , a  $(524 \times 1024)$  generator matrix, and a  $(1024 \times 1024)$  permutation matrix as keys. In the Rao-Nam scheme [6-7], a  $(64 \times 64)$  nonsingular matrix over  $GF(2)$ , a  $(64 \times 72)$  generator matrix, and a  $(72 \times 72)$  permutation matrix were suggested. If these matrices are used directly as keys, over  $2 \times 10^6$  bits are required for each user in McEliece's scheme, and over  $18 \times 10^3$  bits are needed for each pair of users in the Rao-Nam scheme. Fortunately, this problem can be solved. In [19,20], they proposed the use of a short sequence of bits (called seed-key or seed) to specify these matrices.

In our scheme, the secret keys are the generator matrix  $G$  of a product code  $PC(n = (r + 1)(s + 1), k = rs, d = 4)$  in the congruence class modulo  $m$  where  $m = 2^t$ , a random binary  $(nt) \times (kt)$  matrix  $S$ , and an  $(nt) \times (nt)$  permutation matrix  $P$ . If  $t=3, m=8, r=6, s=6, n=49$ , and  $k=36$ , then  $G$  is a  $36 \times 49$  matrix (each entry has the length of 3-bits),  $S$  is a random binary  $147 \times 108$  matrix, and  $P$  is a  $147 \times 147$  permutation matrix. Thus the total length of these used keys is over  $42 \times 10^3$ -bits. For the permutation matrix  $P$ , we can use a short seed-key to generate it, as suggested in [18,19]. For the random binary matrix  $S$ , we can use a random number generator with a short seed to generate  $nkt^2$  random bits and put them into an  $(nt) \times (kt)$  matrix  $S$ . For the generator matrix  $G$  of the product code, we don't need to save it directly. All we need to know is the encoding rule of the product code. The encoding rule can be expressed into a matrix (not the generator matrix), as described in section 2. For example,

$$\begin{bmatrix} m_2 & p_1 & m_4 & m_6 \\ p_3 & p_6 & p_4 & p_5 \\ m_5 & p_2 & m_1 & m_3 \end{bmatrix}_{3 \times 4} \quad \text{denotes the encoding rule of a product code introduced in}$$

section 2.

We can use a short seed-key to specify the matrix as follows:

- (1) Use a few bits of the short seed-key to describe the information,  $(i, j)$ , where the parity-check bits are located (assume these bits are located in the  $i$ th row and the  $j$ th column, for  $1 \leq i \leq r+1, 1 \leq j \leq s+1$ ).
- (2) Use the remaining bits to describe the positions of  $m_i$ 's, for  $1 \leq i \leq rs$ . In fact, if we ignore the positions of the parity-check bits, the positions of  $m_i$ 's can be mapped to a permutation. Therefore, we can specify the positions of  $m_i$ 's by specifying the corresponding permutation matrix.

## 7 Conclusions

In this paper, we analyze the security of CDS private-key cryptosystem, which is based on product codes. We show that this system is insecure against chosen-plaintext attacks, and propose a modified private-key cryptosystem based on product codes. Because of the difficulty to distinguish the embedded error from the ciphertext and to reduce the encryption algorithm, our scheme is secure against the chosen-plaintext attacks proposed to attack CDS and other well-known private-key algebraic-code schemes.

## References

1. McEliece, R.J., "A Public-Key Cryptosystem Based on Algebraic Coding Theory," DSN Progress Report, 42-44 (1978) 114-116
2. Rivest, R.L., Shamir, A., and Adleman, L.M., "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM* 21 (2) (1978) 120-126
3. ElGamal, T., "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," *IEEE Trans. IT-31* (4) (1985) 469-472
4. Korzhik, V.I., and Turkin, A.I., "Cryptanalysis of McEliece's Public-Key Cryptosystem", *Advances in Cryptology-EUROCRYPT'91, Lecture Notes in Computer Science, Springer-Verlag* (1991) 68-70
5. Berson, T.A., "Failure of the McEliece Public-Key Cryptosystem under Message-resend and Related-message Attack," *Advances in Cryptology-CRYPTO'97, Lecture Notes in Computer Science, Vol. 1294. Springer-Verlag* (1997) 213-220
6. Rao, T.R.N., and Nam, K.H., "Private-Key Algebraic-Coded Cryptosystems," *Advances in Cryptology-CRYPTO'86, Lecture Notes in Computer Science, Springer-Verlag* (1987) 35-48
7. Rao, T.R.N., and Nam, K.H., "Private-Key Algebraic-Code Encryption," *IEEE Trans., IT-35* (4) (1987) 829-833
8. Hin, P.J.M., "Channel-Error-Correcting Privacy Cryptosystems," Ph.D. Dissertation (in Dutch), Delft University of Technology (1986)
9. Struik, R., and Tilburg, J., "The Rao-Nam Scheme Is Insecure Against a Chosen-Plaintext Attack," *Advances in Cryptology-CRYPTO'87, Lecture Notes in Computer Science, Springer-Verlag* (1988) 445-457
10. Brickell, E.F., and Odlyzko, A., "Cryptanalysis: A Survey of Recent Results," *Proc. IEEE* 76 (5) (1988) 153-165
11. Denny, W.F., "Encryptions Using Linear and Non-Linear Codes: Implementation and Security Considerations," Ph.D. Dissertation, The Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette (1988)
12. Struik, R., "On the Rao-Nam Scheme Using Nonlinear Codes," in *Proc. of the 1991 IEEE Int. Symp. Information Theory* (1991) 174
13. Alencar, F.M.R., Léo, A.M.P., and Campello de Souza, R.M., "Private-Key Burst Correcting Code Encryption," in *Proc. of the 1993 IEEE Int. Symp. Information Theory* (1993) 227
14. Campello de Souza, R.M., and Campello de Souza, J., "Array Codes for Private-Key Encryption," *Electronics Letters* 30 (17) (1994) 1394-1396

15. Sun, H.M., and Shieh, S.P., "Cryptanalysis of Private-Key Encryption Schemes Based on Burst-Error-Correcting Codes," Proc. Third ACM Conference on Computer and Communications Security (1996) 153-156
16. Al Jabri, A., "Security of Private-Key Encryption Based on Array Codes", Electronics Letters 32 (24) (1996) 2226-2227
17. Campello de Souza, J., and Campello de Souza, R.M., "Product Codes and Private-Key Encryption," in Proc. of the 1995 IEEE Int. Symp. Information Theory (1995) 489
18. Ross, S., A First Course in Probability, Prentice-Hall (1994)
19. Hwang, T., and Rao, T.R.N., "On the Generation of Large  $(s, s^{-1})$  Pairs and Permutation Matrices over the Binary Field," Tech. Rep. Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette (1986)
20. Sun, H.M., and Hwang, T., "Key Generation of Algebraic-Code Cryptosystems", Computers and Mathematics with Applications 27 (2) (1994) 99-106