Improving the Security of the McEliece Public-Key Cryptosystem[1]

Hung-Min Sun

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

Abstract. At Crypt'97, Berson showed that the McEliece public-key cryptosystem suffers from two weaknesses: (1) failure to protect any message which is encrypted more than once, (2) failure to protect any messages which have a known linear relation to one another. In this paper, we propose some variants of the McEliece scheme which can prevent from these attacks. These variants will not reduce the information rate in the original scheme. In addition, to improve the information rate, we also propose some variants of the McEliece scheme which can prevent from Berson-like attacks.

1 Introduction

In 1978, McEliece [16] proposed a public-key cryptosystem (*the McEliece scheme*) based on algebraic coding theory. The idea of this cryptosystem is based on the fact that the decoding problem of an arbitrary linear code is an NP-hard problem [4]. Compared with other public-key cryptosystems [8,21] which involve modular exponentiation, the McEliece scheme has the advantage of high-speed encryption and decryption. In addition, the McEliece scheme is a probabilistic encryption [6,9] that is better than other deterministic encryptions [19,21] in preventing from elimination of any information leaked with public-key cryptography. Up to now, the McEliece scheme is still not widely used. This is because the information rate of this scheme is low (close to 0.5) and it requires large binary matrices as secret key and public key. Some methods [15,18,23] were proposed to improve the information rate of the McEliece scheme. These methods use the added error vector to carry additional information. Some information bits are mapped into an error vector to be added to a codeword. Once the error vector can be identified, the additional information can be recovered. By using these methods, the information rate can be up to around 0.8 or more. For the large key problem, Sun and Hwang [24] proposed the use of a short sequence of bits (called seed-key) to specify secret key. Thus each user only needs to keep a short key, e.g., 64-bit sequence. However, the problem of large public key is still unsolved.

[1] This work was supported in part by the National Science Council, Taiwan, under contract NSC-87-2213-E-324-003.

K. Ohta and D. Pei (Eds.): ASIACRYPT'98, LNCS 1514, pp 200-213, 1998

In the past, many researchers [1,2,7,13,14,25] attempted to break the McEliece scheme. None of these were successful in the general case. Among them, Korzhik and Turkin [13] claimed that they had broken the McEliece scheme. However, most cryptographers don't believe their result to be effective because of lack of obvious evidence to confirm the time bound they claimed. At Crypt'97, Berson [5] showed that the McEliece scheme suffers from two weaknesses: (1) failure to protect any message which is encrypted more than once, (2) failure to protect any messages which have a known linear relation to one another. Although these weaknesses don't lead the McEliece scheme to be broken immediately (i.e., the private key doesn't be recovered), it is possible for an attacker to act on some behavior such that these weaknesses happen. For example, an attacker introduces some errors into the ciphertext, which is sent from the sender to the receiver, such that the receiver cannot decrypt the ciphertext correctly. If the receiver thinks this cause comes from faults in encryption phase, he will request the sender to resume again (encrypt the message and send the ciphertext again). Thus the weakness (1) will occur.

To overcome these weaknesses, Berson [5] suggested spreading randomness through the plaintext in some complicated fashion. Bellare and Rogaway's OAEP [3] *et seq.* which are commonly used to enhance the security of RSA are instructive. Thus the linear relation between the messages will be unable be found by some action of a cryptanalyst. However, these improvements will also reduce the information rate of this scheme.

In this paper we propose some variants of the McEliece public-key cryptosystem which can prevent from the attacks proposed by Berson. These variants will not reduce the information rate in the original scheme. In addition, to improve the information rate, we also propose some variants of the McEliece scheme which can prevent from Berson-like attacks. This paper is organized as follows. In section 2, we provide some background information. In section 3, we present some variants of the McEliece public-key cryptosystem which can prevent from the attacks proposed by Berson. In section 4, we propose more variants of the McEliece public-key cryptosystem which can prevent from Berson-like attacks and improve the information rate. Finally, we conclude this paper in section 5.

2 Preliminaries

2.1 The McEliece Public-Key Cryptosystem

Secret key: S is a random $(k\times k)$ nonsingular matrix over GF(2), called the scrambling matrix,
G is a $(k\times n)$ generator matrix of a binary Goppa code G with the capability of correcting an n-bit random error vector of weight less than or equal to t, and
P is a random $(n\times n)$ permutation matrix.

Public key: $G' = S\,G\,P$

Encryption: $c = mG' + e$, where m is a k-bit message, c is an n-bit ciphertext, and e is an n-bit random error vector of weight t.

Decryption: The receiver first calculates $c' = c P^{-1} = mSG + e P^{-1}$, where P^{-1} is the inverse of P. Because the weight of $e P^{-1}$ is the same as the weight of e, the receiver uses the decoding algorithm of the original code G to obtain $m' = mS$. At last, the receiver recovers m by computing $m = m' S^{-1}$, where S^{-1} is the inverse of G.

In the original version of the McEliece scheme, the parameters k, n, and t were suggested to be 524, 1024, and 50 respectively. Many works [1,2,11,12] were to study the optimal value of these parameters such that a cryptanalyst must take the highest cost to break this system. Optimizations were suggested that if n=1024, k ranges from 524 to 654, and t ranges from 37 to 50. In this paper we use the parameter sizes of the original version without loss of generality.

An obvious attack on the McEliece scheme is to guess 524 positions of c that are not distorted by e, and then find m from $c^* = mG^*$ if G^* is invertible, where c^* and G^* are restrictions onto these positions of c and G'. Because there exist 50 errors embedded in 1024 positions, we need $C_{524}^{1024} / C_{524}^{974} \approx 1.37 \times 10^{16}$ guesses to succeed.

2.2 Berson's Attacks on the McEliece Scheme

Berson [5] proposed two attacks on the McEliece scheme, called message-resend attack and related-message attack. We restate these two attacks in the following.

Message-Resend Attack:

We assume a message m is encrypted twice because of some accident or the special action of a cryptanalyst. Then the cryptanalyst knows: $c_1 = mG' + e_1$, and $c_2 = mG' + e_2$, where $e_1 \neq e_2$ (this is called the message-resend condition). Therefore, $c_1 + c_2 = e_1 + e_2$. It is remarked that the weight of $e_1 + e_2$ is even and at most 100 because the weight of each error vector added in the McEliece scheme is 50. According to Berson's analysis, the expected Hamming weight of $e_1 + e_2$ is about 95.1 if a message-resend condition occurs. If the underlying messages are different, the excepted Hamming weight of $c_1 + c_2$ is 512. Therefore, it is easy to detect the occurrence of a message-resend condition and the weight of $e_1 + e_2$ by observing the Hamming weight of $c_1 + c_2$. If the weight of $e_1 + e_2$ is 94, we need to guess 524 positions of c_1 (c_2) that are not distorted by e_1 (e_2) from 930 possible positions with 3 wrong positions. The probability that we get a correct guess is $C_{524}^{927} / C_{524}^{930} \approx 0.0828$. This means that the cryptanalyst needs only about 12 guesses to succeed.

Similarly, if the weight of $e_1 + e_2$ is 96, only about 5 guesses are required for the cryptanalyst to succeed.

Note that the main cause that Berson's attack succeeds is that by observing the value $c_1 + c_2$ we can obtain more *information* about the positions in which the errors probably occur. In the following, we show how much information for each bit in the error vector goes through observing $c_1 + c_2$. Let $e_1(i)$, $e_2(i)$, $c_1(i)$, and $c_2(i)$ denote the *i*-th bit in e_1, e_2, c_1, and c_2 respectively. Here we assume the value of each bit in the ciphertext is a random variable with probability $p(c_1(i)=0) = p(c_1(i)=1) = 0.5$.

The entropy function [10] $H(e_1(i) \mid c_1(i))$

$$= p(c_1(i)=0) \cdot H(e_1(i) \mid c_1(i)=0) + p(c_1(i)=1) \cdot H(e_1(i) \mid c_1(i)=1)$$

$$= 0.5 \cdot \left(\frac{974}{1024}\log\frac{1024}{974} + \frac{50}{1024}\log\frac{1024}{50}\right) + 0.5 \cdot \left(\frac{974}{1024}\log\frac{1024}{974} + \frac{50}{1024}\log\frac{1024}{50}\right)$$

$$= 0.2814$$

It is clear that $H(e_1(i)) = H(e_1(i) \mid c_1(i))$ and $H(e_2(i)) = H(e_2(i) \mid c_2(i))$. This means that one cannot obtain more information on $e_1(i)$ (or $e_2(i)$) through observing $c_1(i)$ (or $c_2(i)$). However, if the message-resend condition occurs and the weight of $e_1 + e_2$ is 94, then

$H(e_1(i) \mid c_1(i) + c_2(i))$

$$= p(c_1(i) + c_2(i) = 0) H(e_1(i) \mid c_1(i) + c_2(i) = 0) +$$
$$\quad p(c_1(i) + c_2(i) = 1) H(e_1(i) \mid c_1(i) + c_2(i) = 1)$$

$$= \frac{930}{1024} \cdot \left(\frac{927}{930}\log\frac{930}{927} + \frac{3}{930}\log\frac{930}{3}\right) + \frac{94}{1024} \cdot \left(\frac{1}{2}\log 2 + \frac{1}{2}\log 2\right)$$

$$= 0.1203$$

If the message-resend condition occurs and the weight of $e_1 + e_2$ is 96,

then $H(e_1(i) \mid c_1(i) + c_2(i))$

$$= p(c_1(i) + c_2(i) = 0) H(e_1(i) \mid c_1(i) + c_2(i) = 0) +$$
$$\quad p(c_1(i) + c_2(i) = 1) H(e_1(i) \mid c_1(i) + c_2(i) = 1)$$

$$= \frac{928}{1024} \cdot \left(\frac{926}{928}\log\frac{928}{926} + \frac{2}{928}\log\frac{928}{2}\right) + \frac{96}{1024} \cdot \left(\frac{1}{2}\log 2 + \frac{1}{2}\log 2\right)$$

$$= 0.1139$$

Related-Message Attack:

We assume two messages m_1 and m_2 are encrypted and a cryptanalyst knows a linear relation, e.g., the value $m_1 + m_2$, between these two messages. Then the cryptanalyst knows: $c_1 = m_1 G' + e_1$, and $c_2 = m_2 G' + e_2$, where $m_1 \neq m_2$, and $e_1 \neq e_2$. Therefore, $c_1 + c_2 = m_1 G' + e_1 + m_2 G' + e_2 = (m_1 + m_2)G' + (e_1 + e_2)$. Because the value $m_1 + m_2$ is known previously, $(m_1 + m_2)G'$ can be computed. Hence $c_1 + c_2 +$

$(m_1+m_2)G' = e_1+e_2$. As the analysis in the message-resend attack, the number of guesses required to succeed is small.

Basically, the message-resend attack is the special case of the related-message attack where the linear relation between the messages is $m_1+m_2=0$. To overcome these weaknesses, Berson [5] suggested spreading randomness through the plaintext in some complicated fashion. Bellare and Rogaway's OAEP [3] *et seq.* which are commonly used to enhance the security of RSA are instructive. Thus the linear relation between the messages will be unable be found by some action of a cryptanalyst. However, these improvements will also reduce the information rate of this scheme. In the following sections, we propose some variants of the McEliece scheme, which can prevent from the attacks proposed by Berson. Some of them have the same information rate as the original McEliece scheme, and some of them have higher information rate than the original scheme.

3 Some Variants of the McEliece Scheme

In this section, we propose some variants of the McEliece scheme. These variants can prevent the McEliece scheme from the message-resend attack and the related-message attack. In addition, these variants will not reduce the information rate. The public key and the secret key in these variants are the same as those in the original McEliece scheme.

Variant I:

Encryption: $c = (m+h(e))G' + e$, where e is an n-bit random error vector of weight t, and h is a one-way hash function with an input e and an output of a k-bit vector. It is necessary to consider how to apply a well-known one-way hash function, e.g., MD5 [20], to be the function h. We omit the details here.

Decryption: First $m+h(e)$ can be obtained by using the decryption algorithm in the original scheme (the error vector can also be found in the decoding process). Secondly the receiver computes $m = (m+h(e)) + h(e)$.

Security: Let m_1 and m_2 be two messages. If $m_1 = m_2$, then $c_1 + c_2 = (h(e_1)+h(e_2))G' + e_1 + e_2$. The value $(h(e_1)+h(e_2))G'$ is unknown because of lacking the knowledge of $h(e_1)$ and $h(e_2)$. We cannot obtain more information about the positions in which the error occurs. Thus the message-resend attack fails. If the value m_1+m_2 is known, then $c_1 + c_2 = (m_1+m_2+h(e_1)+h(e_2))G' + e_1 + e_2$. Although the value m_1+m_2 is known, $(m_1+m_2+h(e_1)+h(e_2))G'$ will not be known because of lacking the knowledge of $h(e_1)$ and $h(e_2)$. We are not able to obtain any information about the positions in which the error occurs. Thus the related-message attack cannot work.

Variant II:

Encryption: $c = f(m, e)G' + e$, where e is an n-bit random error vector of weight t, and f is a trapdoor one-way function [21] with two inputs (m and e) and an output of a k-bit vector. Here f must have the property that given $f(m, e)$ it is computationally infeasible to find m and e, but it is easy to compute m given $f(m, e)$ and e. For example, DES [17], which has two inputs (message and key) and an output (ciphertext), can be one of candidates. If DES is applied, it is necessary to consider how to implement it to be the function f because DES has a 56-bit key, a 64-bit message, and a 64-bit ciphertext, while f needs an n-bit e, a k-bit m, and a k-bit output. We omit the details here.

Decryption: First $f(m, e)$ can be recovered by using the decryption algorithm in the original scheme (the receiver keeps the error vector in the decoding process). Secondly the receiver computes m by inverting the function f.

Security: If $m_1 = m_2$, then $c_1 + c_2 = (f(m_1, e_1) + f(m_2, e_2))G' + e_1 + e_2$. The value $(f(m_1, e_1) + f(m_2, e_2))G'$ is unknown because of lacking the knowledge of $f(m_1, e_1)$ and $f(m_2, e_2)$. We cannot obtain any information about the positions in which the error occurs. Thus the message-resend attack fails. If the value $m_1 + m_2$ is known, we cannot still erase the item $(f(m_1, e_1) + f(m_2, e_2))G'$. Therefore, this scheme is also secure against the related-message attack.

4 More Variants on Improving the Information Rate

In the past, some researchers [15,18,23] studied how to improve the information rate of the McEliece scheme. They use the added error vector to carry additional information. Thus the information rate of the McEliece scheme can be increased. In this section, we first formally describe their ideas as Variant III. We show that the variant is not secure against Berson-like attacks. And then, we propose some variants which can prevent from Berson-like attacks and improve the information rate.

Variant III:

Encryption: Let $m = (m_a, m_b)$ be the message. $c = m_a G' + e$, where $e = g(m_b)$, g is an invertible function which maps m_b into an n-bit error vector of weight t. Some good candidates of the function g can be found in [15,18,23].

Decryption: First m_a can be recovered by using the decryption algorithm of the code G. In the meantime, the value $g(m_b)$ can also be obtained. Then the receiver computes $m_b = g^{-1}(g(m_b))$, where g^{-1} is the inverse of g.

Information rate: By using this method, the information rate can be improved from 0.51 to 0.79 if k=524, n=1024, and t=50 (additional 284-bit information

is carried), and from 0.63 to 0.87 if k=654, n=1024, and t=37 (additional 225-bit information is carried).

Security: Basically, the idea of this variant is the same as that of the original McEliece scheme. The main difference between both is the randomness of the error vector. The error vector of the former is not truly random, but dependent on the probability distribution of m_b. To provide better security, it is suggested that data compression technique is applied before encryption. Note that this variant is a deterministic encryption.

Let $m_1 = (m_{1a}, m_{1b})$ and $m_2 = (m_{2a}, m_{2b})$ be two messages encrypted. Because each message in this variant contains two parts, we extend the linear relation between two messages to many cases. In Table 1, we show the possible weaknesses of these cases. We give some explanations for these cases in the following.

Case III.A: If m_{1a} is known previously, then $g(m_{1b}) = c_1 + m_{1a}G'$. Thus $m_{1b} = g^{-1}(g(m_b))$.

Case III.B: If m_{1b} is known previously, then we know $m_{1a}G' = c_1 + g(m_{1b})$. It is easy to compute m_{1a} by finding $m_{1a}G^* = (c_1 + g(m_{1b}))^*$, where $(c_1 + g(m_{1b}))^*$ and G^* are restrictions onto some positions of $c_1 + g(m_{1b})$ and G' such that G^* is invertible.

Case III.C: If $m_{1a} = m_{2a}$ and $m_{1b} = m_{2b}$ are known previously, then $c_1 = c_2$. That is, $c_1 + c_2 = 0$. We cannot obtain any information about the positions in which the error occurs.

Case III.D: If $m_{1a} = m_{2a}$ and $m_{1b} \neq m_{2b}$ are known previously, then $e_1 \neq e_2$. Thus $c_1 + c_2 = (m_{1a} + m_{2a})G' + e_1 + e_2 = e_1 + e_2$. Therefore, we can obtain any information about the positions in which the errors occur. Thus m_{1a}, m_{1b}, m_{2a}, and m_{2b} can be known.

Case III.E: If $m_{1a} \neq m_{2a}$ and $m_{1b} = m_{2b}$ are known previously, then $(m_{1a} + m_{2a})G' = c_1 + c_2$. Similar to Case III.B, it is easy to compute $m_{1a} + m_{2a}$.

Case III.F: Similar to Case III.E except that $m_{1a} + m_{2a}$ has been known previously.

Case III.G: If the value $m_{1a} + m_{2a}$ and $m_{1b} \neq m_{2b}$ are known previously, then $c_1 + c_2 = (m_{1a} + m_{2a})G' + e_1 + e_2$. Because the value $m_{1a} + m_{2a}$ is known, $(m_{1a} + m_{2a})G'$ can be computed. Hence $c_1 + c_2 + (m_{1a} + m_{2a})G' = e_1 + e_2$. Therefore, we can obtain any information about the positions in which the errors occur. Thus m_{1a}, m_{1b}, m_{2a}, and m_{2b} can be known.

Table 1. The possible weaknesses in Variant III

	Information Known Previously	Information Leaked
Case III.A	m_{1a} (or m_{2a})	m_{1b} (or m_{2b})
Case III.B	m_{1b} (or m_{2b})	m_{1a} (or m_{2a})
Case III.C	$m_{1a} = m_{2a}$, $m_{1b} = m_{2b}$	None
Case III.D	$m_{1a} = m_{2a}$, $m_{1b} \neq m_{2b}$	m_{1a}, m_{1b}, m_{2a}, m_{2b}
Case III.E	$m_{1a} \neq m_{2a}$, $m_{1b} = m_{2b}$	$m_{1a} + m_{2a}$
Case III.F	$m_{1a} + m_{2a}$, $m_{1b} = m_{2b}$	None
Case III.G	$m_{1a} + m_{2a}$, $m_{1b} \neq m_{2b}$	m_{1a}, m_{1b}, m_{2a}, m_{2b}

From Table 1, it is clear that there are still many weaknesses in Variant III. To overcome these weaknesses and improve the information rate of the McEliece scheme, we propose two variants of the McEliece scheme in the following.

Variant VI:

Encryption: Let $m = (m_a, m_b)$ be the message. $c = (m_a + h(e))\ G' + e$, where $e = g(r \| m_b)$, r is a q-bit random vector, g is an invertible function which maps m_b into an n-bit error vector of weight t, h is a one-way hash function with an input e and an output of a k-bit vector. Here we need the function g to have the following property. Let E be the set of 2^n possible strings of n binary digits, E_{m_b} be the set of all possible outputs of $g(r \| m_b)$ given m_b, x_i be the i-th item in E_{m_b} and $d_i = \min_{j, j \neq i} \{dist.(x_i, x_j)\}$. If we regard E as an n-dimensional Hamming space, we require that the E_{m_b} is uniformly distributed (located) in E. That is, we expect that the E_{m_b} has an approximately maximal value of $\frac{\sum d_i}{2^q}$. Those proposals in [15,18,23] may be the candidates of the function g.

Decryption: First $m_a' = m_a + h(e)$ and e can be found by using the decryption algorithm of the code G. Secondly the receiver computes $r \| m_b = g^{-1}(e)$, where $g^{-1}(e)$ is the inverse of g, and then discards the part r. Thus m_b is obtained. Finally, m_a can be computed by $m_a = m_a' + h(e)$.

Information rate: By using this method, the information rate can be improved from 0.51 to 0.79 if k=524, n=1024, t=50, and q=0; from 0.51 to 0.73 if k=524, n=1024, t=50, and q=64; from 0.63 to 0.87 if k=654, n=1024, t=37, and q=0; and from 0.63 to 0.8 if k=654, n=1024, t=37, and q=64.

Security: We discuss the security of this variant with parameter q=0 and q=64 respectively.

Parameter q=0:
In Table 2, we show the possible weaknesses in Variant IV with parameter q=0. Some explanations for these cases are given in the following.

CaseIV.A: Assume m_{1a} is known previously. $(m_{1a}+h(g(m_{1b}))G'$ cannot be removed from c_1 because $h(g(m_{1b}))$ is unknown.

CaseIV.B: If m_{1b} is known previously, then we know $(m_{1a}+h(g(m_{1b}))G' = c_1 + g(m_{1b})$. Similar to Case III.B, it is easy to compute $m_{1a}+h(g(m_{1b})$ and hence m_{1a}.

CaseIV.C: Similar to Case III.C.

CaseIV.D: If $m_{1a}=m_{2a}$ and $m_{1b}\neq m_{2b}$ are known previously, then $c_1+c_2 = (h(g(m_{1b})+h(g(m_{2b}))G'+e_1+e_2$. We cannot remove $(h(g(m_{1b})+h(g(m_{2b}))G'$ from c_1+c_2. Therefore, we cannot obtain any information about the positions in which the errors occur.

CaseIV.E: Similar to Case III.E.

CaseIV.F: Similar to Case III.F.

CaseIV.G: If the value $m_{1a}+m_{2a}$ and $m_{1b}\neq m_{2b}$ are known previously, then $c_1+c_2 = (m_{1a}+m_{2a}+h(g(m_{1b})+h(g(m_{2b}))G' + e_1+e_2$. Because the value $m_{1a}+m_{2a}$ is known, $(m_{1a}+m_{2a})G'$ can be computed. Hence $c_1+c_2+(m_{1a}+m_{2a})G' = h(g(m_{1b})+h(g(m_{2b}))G'+e_1+e_2$. However, we cannot remove $(h(g(m_{1b})+h(g(m_{2b}))G'$ from $c_1+c_2+(m_{1a}+m_{2a})G'$.

Table 2. The possible weaknesses in Variant IV with parameter q=0

	Information Known Previously	Information Leaked
Case IV.A	m_{1a} (or m_{2a})	None
Case IV.B	m_{1b} (or m_{2b})	m_{1a} (or m_{2a})
Case IV.C	$m_{1a}=m_{2a}$, $m_{1b}=m_{2b}$	None
Case IV.D	$m_{1a}=m_{2a}$, $m_{1b}\neq m_{2b}$	None
Case IV.E	$m_{1a}\neq m_{2a}$, $m_{1b}=m_{2b}$	$m_{1a}+m_{1b}$
Case IV.F	$m_{1a}+m_{2a}$, $m_{1b}=m_{2b}$	None
Case IV.G	$m_{1a}+m_{2a}$, $m_{1b}\neq m_{2b}$	None

Parameter q=64:
In Table 3, we show the possible weaknesses in Variant IV with parameter q=64. Some explanations for these cases are given in the following.

CaseIV.R.A: Similar to Case IV.A.

CaseIV.R.B: Assume m_{1b} is known previously. Because r_1 is an unknown 64-bit random vector, the probability that we get a correct guess of the value $g(r_1 \| m_{1b})$ is only $\frac{1}{2^{64}}$. Therefore, we cannot remove $g(r_1 \| m_{1b})$ from c_1. Another possible attack is to guess k positions of c that are not distorted by e. Because $E_{m_{1b}}$ is uniformly distributed in E, a cryptanalyst cannot identify which positions have better chances.

CaseIV.R.C: If $m_{1a} = m_{2a}$ and $m_{1b} = m_{2b}$ are known previously, then $c_1 + c_2 = (h(g(r_1 \| m_{1b}) + h(g(r_2 \| m_{2b}))G' + g(r_1 \| m_{1b}) + g(r_2 \| m_{2b})$. We cannot remove $(h(g(r_1 \| m_{1b}) + h(g(r_2 \| m_{2b}))G'$ from $c_1 + c_2$ because m_{1a}, m_{2a}, m_{1b}, and m_{2b} are unknown.

CaseIV.R.D: Similar to Case IV.D.

CaseIV.R.E: If $m_{1a} \neq m_{2a}$ and $m_{1b} = m_{2b}$ are known previously, then $c_1 + c_2 = (m_{1a} + m_{2a} + h(g(r_1 \| m_{1b}) + h(g(r_2 \| m_{2b}))G' + g(r_1 \| m_{1b}) + g(r_2 \| m_{2b})$. Because r_1 is a 64-bit random vector, the probability that $r_1 = r_2$ (hence $g(r_1 \| m_{1b}) = g(r_2 \| m_{2b})$) is equal to $1/2^{64}$ which is significantly small. Therefore, neither $(m_{1a} + m_{2a} + h(g(r_1 \| m_{1b}) + h(g(r_2 \| m_{2b}))G'$ nor $g(r_1 \| m_{1b}) + g(r_2 \| m_{2b})$ can be removed from $c_1 + c_2$.

CaseIV.R.F: If the value $m_{1a} + m_{1b}$ and $m_{2a} = m_{2b}$ are known previously, then $c_1 + c_2 + (m_{1a} + m_{2a})G' = (h(g(r_1 \| m_{1b}) + h(g(r_2 \| m_{2b}))G' + g(r_1 \| m_{1b}) + g(r_2 \| m_{2b})$. Neither $(h(g(r_1 \| m_{1b}) + h(g(r_2 \| m_{2b}))G'$ nor $g(r_1 \| m_{1b}) + g(r_2 \| m_{2b})$ can be removed from $c_1 + c_2 + (m_{1a} + m_{2a})G'$.

CaseIV.R.G: Similar to Case IV.G.

Table 3. The possible weaknesses in Variant IV with parameter *q*=64

	Information Known Previously	Information Leaked
Case IV.R.A	m_{1a} (or m_{2a})	None
Case IV.R.B	m_{1b} (or m_{2b})	None
Case IV.R.C	$m_{1a} = m_{2a}$, $m_{1b} = m_{2b}$	None
Case IV.R.D	$m_{1a} = m_{2a}$, $m_{1b} \neq m_{2b}$	None
Case IV.R.E	$m_{1a} \neq m_{2a}$, $m_{1b} = m_{2b}$	None
Case IV.R.F	$m_{1a} + m_{2a}$, $m_{1b} = m_{2b}$	None
Case IV.R.G	$m_{1a} + m_{2a}$, $m_{1b} \neq m_{2b}$	None

Variant V:

Encryption: Let $m = (m_a, m_b)$ be the message. $c = f(m_a, e)\, G' + e$, where $e = g(r \| m_b)$, g is an invertible function which maps $r \| m_b$ into an n-bit

error vector of weight t, and f is a trapdoor one-way function with two inputs (m_a and e) and an output of a k-bit vector. Here the function f and the function g should have the same property as that in Variant II and that in Variant IV respectively.

Decryption: First $m_a' = f(m_a, e)$ and e can be obtained by using the decryption algorithm of the code G. Secondly the receiver computes $m_b = g^{-1}(e)$, where g^{-1} is the inverse of g. Finally, m_a can be computed by $m_a = f^{-1}(m_a', e)$, where f^{-1} is the inverse of f.

Information rate: the same as Variant IV.

Security: We discuss the security of this variant with parameter q=0 and q=64 respectively.

Parameter q=0:

In Table 4, we show the possible weaknesses in Variant V with parameter q=0. Some explanations for these cases are given in the following.

Case V.A: Similar to Case IV.A.

Case IV.B: If m_{1b} is known previously, we know $f(m_{1a}, g(m_{1b}))G' = c_1 + g(m_{1b})$. Similar to Case III.B, it is easy to compute $f(m_{1a}, g(m_{1b}))$ and hence $m_{1a} = f^{-1}(f(m_{1a}, g(m_{1b})), g(m_{1b}))$.

Case IV.C: Similar to Case III.C.

Case IV.D: If $m_{1a} = m_{2a}$ and $m_{1b} \neq m_{2b}$ are known previously, then $c_1 + c_2 = (f(m_{1a}, g(m_{1b})) + f(m_{2a}, g(m_{2b})))G' + e_1 + e_2$. We cannot erase $(f(m_{1a}, g(m_{1b})) + f(m_{2a}, g(m_{2b})))G'$ from $c_1 + c_2$.

Case IV.E: If $m_{1a} \neq m_{1b}$ and $m_{2a} = m_{2b}$ are known previously, then $c_1 + c_2 = (f(m_{1a}, g(m_{1b})) + f(m_{2a}, g(m_{2b})))G'$. We can only obtain the value $f(m_{1a}, g(m_{1b})) + f(m_{2a}, g(m_{2b}))$.

Case IV.F: Similar to Case IV.E.

Case IV.G: Similar to Case IV.D.

Table 4. The possible weaknesses in Variant V with parameter q=0

	Information Known Previously	Information Leaked
Case V.A	m_{1a} (or m_{2a})	None
Case V.B	m_{1b} (or m_{2b})	m_{1a} (or m_{2a})
Case V.C	$m_{1a} = m_{2a}$, $m_{1b} = m_{2b}$	None
Case V.D	$m_{1a} = m_{2a}$, $m_{1b} \neq m_{2b}$	None
Case V.E	$m_{1a} \neq m_{2a}$, $m_{1b} = m_{2b}$	None
Case V.F	$m_{1a} + m_{2a}$, $m_{1b} = m_{2b}$	None
Case V.G	$m_{1a} + m_{2a}$, $m_{1b} \neq m_{2b}$	None

Parameter q=64:
In Table 5, we show the possible weaknesses in Variant V with parameter q=64. Some explanations for these cases are given in the following.

Case V.R.A: Similar to Case IV.R.A.
Case V.R.B: Similar to Case IV.R.B.
Case V.R.C: If $m_{1a} = m_{2a}$ and $m_{1b} = m_{2b}$ are known previously, then $c_1 + c_2 = (f(m_{1a}, g(r_1 \| m_{1b})) + f(m_{2a}, g(r_2 \| m_{2b})))G' + g(r_1 \| m_{1b}) + g(r_2 \| m_{2b})$. We cannot remove neither $(f(m_{1a}, g(r_1 \| m_{1b})) + f(m_{2a}, g(r_2 \| m_{2b})))G'$ nor $g(r_1 \| m_{1b}) + g(r_2 \| m_{2b})$ from $c_1 + c_2$.
Case V.R.D: Similar to Case V.D.
Case V.R.E: Similar to Case IV.R.E.
Case V.R.F: Similar to Case V.R.C.
Case V.R.G: Similar to Case V.G.

Table 5. The possible weaknesses in Variant V with parameter q=64

	Information Known Previously	Information Leaked
Case V.R.A	m_{1a} (or m_{2a})	None
Case V.R.B	m_{1b} (or m_{2b})	None
Case V.R.C	$m_{1a} = m_{2a}$, $m_{1b} = m_{2b}$	None
Case V.R.D	$m_{1a} = m_{2a}$, $m_{1b} \neq m_{2b}$	None
Case V.R.E	$m_{1a} \neq m_{2a}$, $m_{1b} = m_{2b}$	None
Case V.R.F	$m_{1a} + m_{2a}$, $m_{1b} = m_{2b}$	None
Case V.R.G	$m_{1a} + m_{2a}$, $m_{1b} \neq m_{2b}$	None

5 Conclusions

In this paper, we first propose two variants, Variant I and Variant II, of the McEliece scheme, which can prevent from both the message-resend attack and the related-message attack. These two variants are probabilistic encryptions, and have the same information rate as that of the original McEliece scheme. To improve the information rate and to prevent from Berson-like attacks, we also propose two variants, Variant IV and Variant V, of the McEliece scheme. In these two variants, if the parameter q is equal to 0, then they are deterministic encryptions and can improve the information rate from 0.51 to 0.79 if k=524, n=1024, t=50, or from 0.63 to 0.87 if k=654, n=1024, t=37. If the parameter q is equal to 64, then they are probabilistic encryptions and can improve the information rate from 0.51 to 0.73 if k=524, n=1024, t=50, or from 0.63 to 0.8 if k=654, n=1024, t=37.

References

1. Adams, C., and Meijer, H., „Security-Related Comments Regarding McEliece's Public-Key Cryptosystem," *Advances in Cryptology-CRYPTO'87*, Lecture notes in computer science (Springer-Verlag), pp. 224-228, 1988.
2. Adams, C., and Meijer, H., „Security-Related Comments Regarding McEliece's Public-Key Cryptosystem," *IEEE Transactions on Information Theory,* Vol. 35, pp. 454-455, 1989.
3. Bellare, M., and Rogaway, P., „Optimal asymmetric encryption," *Advances in Cryptology-EUROCRYPT'94*, Lecture notes in computer science 950 (Springer-Verlag), pp. 232-249, 1994.
4. Berlekamp, E.R., McEliece, R.J., and van Tilborg, H.C.A., „On the Inherent Intractability of Certain Coding Problems," *IEEE Transactions on Information Theory,* Vol. 24, pp. 384-386, 1978.
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 (Springer-Verlag), pp. 213-220, 1997.
6. Blum, M., and Goldwasser, S., „An Efficient Probabilistic Public-Key Encryption Scheme Which Hides All Partial Information," *Advances in Cryptology-CRYPTO'84*, Lecture notes in computer science (Springer-Verlag), pp. 289-299, 1985..
7. Brickell, E.F., and Odlyzko, A., „Cryptanalysis: A Survey of Recent Results," *Proc. IEEE*, 76, (5), pp. 153-165, 1988.
8. ElGamal, T., „A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," *IEEE Trans*., IT-31, (4), pp. 469-472, 1985.
9. Goldwasser, S., and Micali, S., „Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information," *Proceedings of the 14th ACM Symposium on the Theory of Computing*, pp. 270-299, 1982.
10. Hamming, R.W., *Coding and Information Theory*, Prentice-Hall, 1986.
11. Hin, P.J.M., „Channel-Error-Correcting Privacy Cryptosystems," M.Sc. Thesis, Delft University of Technology, Delft, 1986.
12. Jorissen, F., „A Security Evaluation of the Public-Key Cipher System Proposed by McEliece, used as a combined scheme," Technical Report, Katholieke University Leuven, Dept. Elektrotechniek, Jan 1986.
13. 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), pp. 68-70, 1991.
14. Lee, P.J., and Brickell, E.F., „An observation on the security of McEliece's Public-Key Cryptosystem," *Advances in Cryptology-EUROCRYPT'88*, Lecture notes in computer science (Springer-Verlag), pp. 275-280, 1988.
15. Lin, M.C., and Fu, H.L., „Information Rate of McEliece's Public-Key Cryptosystem," *Electronics Letters*, Vol. 26, No. 1, pp. 16-18, 1990.
16. McEliece, R.J., „A Public-Key Cryptosystem Based on Algebraic Coding Theory," *DSN Progress Report*, 42-44, pp. 114-116, 1978.
17. National Bureau of Standards, NBS FIPS PUB 46, „Data Encryption Standard," *National Bureau of Standards*, U.S. Department of Commerce, Jan 1977.
18. Park, C.S., „Improving Code Rate of McEliece's public-Key Cryptosystem," *Electronics Letters*, Vol. 25, No. 21, pp. 1466-1467, 1989.
19. Rabin, M.O., „Digital Signatures and Public-Key Functions as Intractable as Factorization," MIT Lab. For Computer Science, Technical Report, MIT/LCS/TR-212, Jan 1979.
20. Rivest, R.L., „The MD5 Message Digest Algorithm," RFC 1321, Apr 1992.

21. 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), pp. 120-126, 1978.
22. Schneier, B., *Applied Cryptography*, John Wiley & Sons, 1996.
23. Sendrier, N., „Efficient Generation of Binary Words of Given Weight," *Cryptography and Coding: 5th IMA Conference*, (Springer-Verlag), pp. 184-187, 1995.
24. Sun, H.M., and Hwang, T., „Key Generation of Algebraic-Code Cryptosystems", *Computers and Mathematics with Applications*, 27, (2), pp. 99-106, 1994.
25. van Tilburg, J., „On the McEliece Public-Key Cryptosystem," *Advances in Cryptology-CRYPTO'88*, Lecture notes in computer science (Springer-Verlag), pp. 119-131, 1990.