Cryptanalysis of Enhanced MORE

Damian Vizár and Serge Vaudenay

EPFL
CH-1015 Lausanne, Switzerland
http://lasec.epfl.ch

Abstract. Fully homomorphic encryption (FHE) has been among the most popular research topics of the last decade. While the bootstrapping-based, public key cryptosystems that follow Gentry's original design are getting more and more efficient, their performance is still far from being practical. This lead to several attempts to construct *symmetric* FHE schemes that would not be as inefficient as their public key counterparts. Unfortunately, most such schemes were also based on (randomized) linear transformations, and shown completely insecure. One such broken scheme was the Matrix Operation for Randomization and Encryption (MORE). In a recent paper, Hariss, Noura and Samhat propose Enhanced MORE, which is supposed to improve over MORE's weaknesses. We analyze Enhanced MORE, discuss why it does not improve over MORE, and show that it is even less secure by presenting a highly efficient ciphertext-only decryption attack. We implement the attack and confirm its correctness.

Keywords: Homomorphic Encryption, Symmetric encryption, Cryptanalysis, Key-recovery

1 Introduction

First introduced in 1978 [9], the concept of *homomorphic encryption* refers to a cryptosystem, whose decryption algorithm is a homomorphism w.r.t. a binary operation over the ciphertexts and another one over the plaintexts. If the plaintext space is a suitable group (or a monoid), such as the group $(\mathbb{Z}_n, +)$ in the Pailler cryptosystem [8], the homomorphic property enables one to do computations over encrypted data. This is very useful in several modern applications, mainly in cloud computing, where the computationally powerful cloud needs to carry out computation over potentially sensitive data.

In 2009, Gentry demonstrated that it is possible to construct *fully homomorphic encryption* (FHE) [4], whose decryption algorithm is essentially a ring homomorphism. This means, that for any circuit of "reasonable size," whose gates are the two ring operations, evaluating the circuit over plaintexts would give the same result as evaluating it over ciphertexts and then decrypting.

While Gentry's result was a sensation, the cryptosystems [3,1,11] that follow his, so called bootstrapping design paradigm, suffer from similar efficiency issues: the ciphertext expansion is very large, and the homomorphic computations too slow for any practical application. There were several designs that tried to circumvent these issues by dispensing with the bootstrapping in the symmetric key setting [2,6,13]. To facilitate efficient FHE, they do however resort to using only linear transformations, a practice which is known to yield very insecure designs. Unsurprisingly, they were all completely broken [12].

One such symmetric FHE is the Matrix Operation for Randomization and Encryption (MORE) [6], whose encryption algorithm is essentially a conjugation by a random invertible matrix over an instance of the RSA ring. MORE came with a security proof, which showed it secure against ciphertext-only decryption and key recovery attacks, assuming the uniform distribution of the plaintexts. Such an assumption is of course unrealistic, and MORE was shown completely insecure in most of real-world applications. Recently, Hariss, Noura and Samhat have attempted to resurrect MORE by proposing Enhanced MORE (EMORE) [5], which adds more key-dependent transformations around the core of MORE. These are supposed to make the cryptosystem more secure, and more efficient.

In this work, we analyze EMORE, and we argue that the introduced modifications improve neither the security, nor the applicability over MORE. On the contrary, we demonstrate that they *decrease* the security, by mounting a ciphertext-only decryption attack.

2 Notation, Syntax and Security Model

In this section, we introduce the used notation, define the syntax of symmetric FHE, and define the attack model.

Notation. We let $\mathbb{Z}_N$ denote the ring of integers modulo an integer $N \in \mathbb{N}$ with the usual operations. For a matrix $A \in \mathbb{Z}_N^{m \times n}$, we let $A(i,j)$ denote the entry in row i and column j, and let $A(i,:)$ and $A(:,j)$ denote the i-th row and j-th column of A, respectively. We also let $A(j)$ denote the j-th column of A. Given an implicit dimension n, we let $\mathrm{DiagMat}(d_1, \ldots, d_n)$ denote the diagonal matrix with $d_1, \ldots, d_n$ placed on the diagonal. Given a square matrix A of dimension n and an integer ℓ that divides n, we let $\mathrm{SubM}(A, \ell, i, j)$ denote the submatrix $A_{i,j}$ of A, such that

$$A = \begin{pmatrix} A_{1,1} & \cdots & A_{1,\ell} \\ \vdots & \ddots & \vdots \\ A_{\ell,1} & \cdots & A_{\ell,\ell} \end{pmatrix}.$$

Given an implicit dimension n, we let I denote the identity matrix of dimension n and we let e_i denote the i-th canonical unit vector, i.e. a column vector that has a 1 in i-th entry and zeros everywhere else. Given a square matrix A of dimension n, we let $\mathrm{CharPoly}(M)(x)$ denote the characteristic polynomial of M in the indeterminate x. Given a polynomial $p(x)$ of degree d, we let p_i denote the coefficient of the i-th power of x, i.e. $p(x) = \sum_{i=0}^{d} = p_i \cdot x^i$.

Syntax. A symmetric encryption scheme is a triplet $\Pi = (\mathrm{KeyGen}, \mathcal{E}, \mathcal{D})$. The key generator $\mathrm{KeyGen} : \mathbb{N} \to \mathcal{K} \times \mathcal{P}$ is a randomized algorithm that takes a security parameter λ, and outputs a secret key K and a public parameter P. The possibly randomized encryption algorithm $\mathcal{E} : \mathcal{K} \times \mathcal{P} \times \mathcal{M} \to \mathcal{C}$ maps a secret key K, a public parameter P and a plaintext M to a ciphertext C, and the deterministic decryption algorithm $\mathcal{D} : \mathcal{K} \times \mathcal{P} \times \mathcal{C} \to \mathcal{M}$ takes a secret key K, a public parameter P and a ciphertext C, and gives a plaintext M. We silently allow all the domains to depend on the security parameter, and the plaintext space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ can additionally depend on the secret key and the public parameter.

We require perfect correctness, i.e. for any security parameter $\lambda \in \mathbb{N}$, and for all $K, P, M \in \mathcal{K} \times \mathcal{P} \times \mathcal{M}$, we require that $\Pr[\mathcal{D}(K, P, \mathcal{E}(K, P, M)) = M] = 1$.

Homomorphic property. For the homomorphic property, we use the notion of circuits. We assume that $(\mathcal{M}, \oplus, \otimes)$ and $(\mathcal{C}, \boxplus, \boxtimes)$ are both commutative rings. A circuit $\mathsf{c} : \mathcal{M}^\mu \to \mathcal{M}$ maps a μ-tuple of plaintexts to a single plaintext by applying the binary operations $\oplus, \otimes$ to the inputs and intermediate results (we can think of the operations as gates).

A transformed circuit $\mathrm{Transf}(\mathsf{c}) : \mathcal{P} \times \mathcal{C}^\mu \to \mathcal{C}$ is a circuit over the ciphertext space that is obtained by replacing each application of $\oplus$ in c by $\boxplus$, and doing similarly for $\otimes$ and $\boxtimes$. The transformed circuit additionally takes the public parameter.

We say that a symmetric encryption scheme Π is fully homorphic, if for every $\lambda \in \mathbb{N}$, for every circuit $\mathsf{c} : \mathcal{M}^\mu \to \mathcal{M}$ and for all $K, P, (M_1, \ldots, M_\mu) \in \mathcal{K} \times \mathcal{P} \times \mathcal{M}^\mu$, we have that

$$\Pr[\mathcal{D}(K, P, \mathrm{Transf}(\mathsf{c})(P, \mathcal{E}(K, P, M_1), \ldots, \mathcal{E}(K, P, M_\mu))) \neq \mathsf{c}(M_1, \ldots, M_\mu)] = \mathsf{negl}(\lambda).$$

Security Model. Since the scheme we study in this paper is based on linear transformations, it obviously does not resist to any standard attacks, e.g. chosen plaintext decryption attacks. We therefore work with a very weak security model (in the sense that the task of the adversary is

very hard). We mount ciphertext-only decryption attacks with the plaintexts being sampled with a uniform distribution.

More precisely, we consider a security experiment parameterized by λ, which first runs $K, P \leftarrow \text{KeyGen}(\lambda)$, and then gives P to an adversary, and lets him/her to query $r \in \mathbb{N}$. Then, the experiment samples r plaintexts uniformly from $\mathcal{M}$, encrypts each of them, and gives the corresponding ciphertexts to the adversary. The adversary then wins if it finds all r plaintexts.

3 The MORE and EMORE Constructions

In this Section we first recall MORE, and then describe the scheme EMORE. We discuss what we see as flaws in the design of EMORE, and then we introduce a simplified version of EMORE, whose security is equivalent to the original version presented by the authors.

3.1 MORE

The scheme MORE [6] works over an RSA ring, i.e. over $\mathbb{Z}_N$ with N a product of two large primes (whose bit-length is determined by the security parameter). The plaintext space of MORE is $\mathcal{M} = \mathbb{Z}_N$, the ciphertext space is the matrix space $\mathcal{C} = \mathbb{Z}_N^{2\times 2}$, and the key space $\mathcal{K}$ is the set of all invertible matrices in $\mathbb{Z}_N^{2\times 2}$. The encryption algorithm creates a diagonal matrix from the plaintext and a uniform element of $\mathbb{Z}_N$, and conjugates it with the secret key. The result is the ciphertext. The algorithms of MORE are described in Figure 1.

Algorithm $\textsc{KeyGen}(\lambda)$
  Pick primes $p(\lambda), q(\lambda)$
  $N \leftarrow p \cdot q$
  $K \leftarrow\$ \{A \in \mathbb{Z}_N^{2\times 2} \mid A^{-1} \text{ exists}\}$
  return N, K
end Algorithm

Algorithm $\mathcal{E}(K, N, m)$
  $u \leftarrow\$ \mathbb{Z}_N$
  $M \leftarrow \text{DiagMat}(m, u)$
  return $K \cdot M \cdot K^{-1}$
end Algorithm

Algorithm $\mathcal{D}(K, N, C)$
  $M \leftarrow K^{-1} \cdot M \cdot K$
  return $M_{1,1}$
end Algorithm

Fig. 1. The symmetric FHE scheme MORE.

With N as the public parameter, MORE is easily seen to have perfect homomorphic properties. The authors of MORE proved that when the adversary is restricted to ciphertext-only attacks and MORE is used with plaintexts that are uniform and independent, then a decryption attack is equivalent with finding square roots in $\mathbb{Z}_N$, which is believed to be hard. Such an assumption on plaintexts is of course unrealistic, and it was shown that in most realistic settings, even side channel information about the plaintexts is enough to break MORE.

3.2 Enhanced MORE

The scheme EMORE [5] is supposed to overcome the drawbacks of MORE, according to the authors. We first give a brief, informal description of the original EMORE, as proposed by its authors.

Original EMORE. The original description of EMORE is not very precise. Here we give our interpretation of this description. EMORE works over $\mathbb{Z}_N$ with N a product of two large primes. It is parameterized by three integers ℓ, n and w, with n even. The plaintexts live in $\mathbb{Z}_N^\ell$. The key

space is a set of bit strings of an unspecified length (possibly λ would work). The key generation is thus a dummy algorithm that just samples a uniform bit string.

The encryption algorithm of EMORE takes a secret key K and a plaintext vector $(m_1, \ldots, m_\ell)$. It consists of the following stages:

1. *Key derivation.* First, the key K is used to derive three bit strings DK_p, DK_d and DK_s of $23 \cdot 8$, $16 \cdot 8$ and $23 \cdot 8$ bits, respectively. Then, we further derive a pseudorandom permutation π of the set $\{1, \ldots, \ell\}$ using DK_p, a collection of w psudorandom $n \times n$ matrices $K^1, \ldots, K^w$ from a special set $\mathcal{S}_{N,n}$ (defined below) using DK_d, and a pseudorandom permutation Δ of the set $\{1, \ldots, w\}$ using DK_s.

2. *Partitioning and permutation.* The plaintext vector $(m_1, \ldots, m_\ell)$ is permuted by π to obtain $\bar{m} = (m_{\pi(1)}, \ldots, m_{\pi(\ell)})$. Then, $\bar{m}$ is partitioned into $H = \lceil \ell / n \rceil$ smaller vectors of dimension n $(\bar{m}_1^i, \ldots, \bar{m}_n^i)$ for $1 \leq i \leq H$ (the last one possibly padded with zero-elements).

3. *MORE-like encryption.* We construct matrices $M^i = \mathrm{DiagMat}(\bar{m}_1^i, \ldots, \bar{m}_n^i)$ for $1 \leq i \leq H$. Then we compute the conjugate matrices $C^i = (K^{\Delta(i)})^{-1} \cdot M^i \cdot K^{\Delta(i)}$ for $1 \leq i \leq H$. We note that the secret matrix $K^{\Delta(i)}$ used to encrypt the i-th block of plaintext is selected with the secret permutation Δ. The final ciphertext is the list of matrices $C^1, \ldots, C^H$.

The decryption algorithm of the original EMORE is easy to derive. We first go through the same key derivation stage as in the encryption. Then, we reverse the MORE-like encryption, concatenate the obtained vectors of dimension n and finally apply the inverse of π. The encryption algorithm of EMORE is illustrated in Figure 2.

To derive DK_p, DK_d and DK_s, the authors suggests to use an unspecified hash function. The pseudorandom permutations are supposed to be generated using a technique proposed by Noura and Courrousé [7], and the collection of special matrices is generated using the streamcipher RC4. The matrices $K^1, \ldots, K^w$ are constructed in a way that facilitates their easy inversion. Each K^i and its inverse $(K^i)^{-1}$ have the following structure:

$$K^i = \begin{pmatrix} A & A+I \\ A-I & A \end{pmatrix} \qquad (K^i)^{-1} = \begin{pmatrix} A & -A-I \\ -A+I & A \end{pmatrix}$$

where A is an invertible matrix from $\mathbb{Z}_N^{\frac{n}{2} \times \frac{n}{2}}$. This is why n is required to be even. It is easy to verify that if A is invertible, then $\det(K^i) = 1$, and $(K^i)^{-1} \cdot K = I$. Generating such a matrix randomly is equivalent with sampling A from the domain of these matrices defined as

$$\mathcal{S}_{N,n} = \left\{ \begin{pmatrix} A & A+I \\ A-I & A \end{pmatrix} \middle| A \in \mathbb{Z}_N^{\frac{n}{2} \times \frac{n}{2}} \text{ and invertible} \right\}.$$

Complaints about EMORE. The design of Enhanced MORE contains several flaws and redundancies which limit security and applicability. We enumerate them here.

- *Bad key derivation.* The key derivation stage of EMORE contains some design choices that seem arbitrary. For instance, the authors choose to use an unspecified hash function to derive DK_p, DK_d and DK_s, but opt for RC4 when generating the matrices $K^1, \ldots, K^w$. We find it preferable to keep both components unspecified and treat them as parameters of EMORE (especially since RC4 is known to be biased). Beside this, the sizes of DK_p, DK_d and DK_s are inexplicably fixed, while the number of keys w and the dimension n are left as parameters. This limits the effective security of EMORE, regardless of the security parameter.
- *Limited usefulness of the homomorphic properties.* The homomorphic properties of MORE encryption scheme are based on the fact, that all ciphertexts are a diagonal matrices conjugated with *the same* key matrix; this is why the multiplication and addition of ciphertexts is preserved by the decryption.
  However in EMORE, any two plaintext sub-vectors $\bar{m}^i, \bar{m}^j$ with $i \neq j$ get encrypted with two independent matrices $K^{\Delta(i)}, K^{\Delta(j)}$. Thus it will be impossible to do any homomorphic

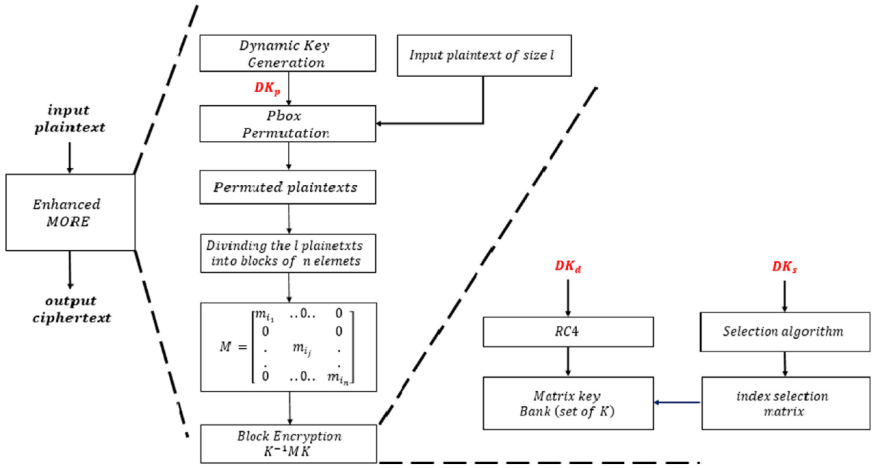

Fig. 2. The encryption algorithm of the Enhance MORE (the original version). Picture taken from the original paper [5].

computations that involve both $\bar{m}^i$ and $\bar{m}^j$, unless $\bar{m}^j$ is included in *another* plaintext vector on the position i or vice-versa. With $\ell > n$, EMORE is thus something like $H = \lceil \ell/n \rceil$ parallel instances of a smaller symmetric FHE. Moreover, *all* of them can only execute the same circuit when doing the homomorphic computations.

– *Useless key components.* The key selection by the permutation Δ is completely useless. Assuming the matrices $K^1, \ldots, K^w$ are uniformly sampled from $\mathcal{S}_{N,n}$, it is equivalent to simply sample H key matrices, and use them without permutation.
The initial permutation π is also of limited use. We demonstrate that given a matrix C^i and N, it is possible to extract the underlying plaintext sub-vector $(\bar{m}^i_1, \ldots, \bar{m}^i_n)$, regardless of π, It is debatable whether the fact that they are shuffled has any security benefits.

Simplified EMORE. Based on our critique of EMORE, we define simplified EMORE (sEMORE). In sEMORE, we work with plaintext vectors from $\mathbb{Z}_N^n$, so there is no partitioning of plaintexts. We also dispense with the permutation π. Because the plaintexts are now treated as monolythic vectors, we also use just a single key matrix. We treat the dimension n as an implicit parameter of the scheme. The algorithms of sEMORE are defined in Figure 3.

Algorithm KeyGen(λ,n)
  Pick primes $p(\lambda), q(\lambda)$
  $N \leftarrow p \cdot q$
  $K \leftarrow\!\!\$\ \mathcal{S}_{N,n}$
  return N, K
end Algorithm

Algorithm $\mathcal{E}(K, N, (m_1, \ldots, m_n))$
  $M \leftarrow \text{DiagMat}(m_1, \ldots, m_n)$
  return $K^{-1} \cdot M \cdot K$
end Algorithm

Algorithm $\mathcal{D}(K, N, C)$
  $M \leftarrow K \cdot M \cdot K^{-1}$
  return $(M_{1,1}, \ldots, M_{n,n}$
end Algorithm

Fig. 3. The symmetric FHE scheme simplified EMORE.

A decryption attack on sEMORE will easily translate to EMORE. We describe such an attack in the next section.

4 Decryption Attack on sEMORE

In this section, we analyze the security of sEMORE. Contrary to their intentions, the authors of EMORE have not improved, but decreased security of their construction compared to MORE. In MORE, a ciphertext-only attack in a setting with uniform plaintexts was not possible. Here, we show that sEMORE (and thus also EMORE) succumb to a single-ciphertext decryption attack in just such a setting.

Formally, we mount an attack in the model described in Section 2, using a single ciphertext C.

Redundant structure. We first exploit the known structure of the key matrix K to obtain some information about the underlying plaintext vector $(m_1, \ldots, m_n)$. We note that $(m_1, \ldots, m_n)$ are eigenvalues of C.

We do not know A, but we know that the eigenvector of C associated to m_i is of the form

$$\begin{pmatrix} A(i) \\ -A(i) + e_i \end{pmatrix} \text{ for } 1 \leq i \leq \frac{n}{2} \text{ and } \begin{pmatrix} -A(i - n/2) - e_{i-n/2} \\ A(i - n/2) \end{pmatrix} \text{ for } \frac{n}{2} < i \leq n.$$

Setting $C_{i,j} = \mathrm{SubM}(C, 2, i, j)$ for $i, j \in \{1, 2\}$, we then deduce the following equalities for $1 \leq i \leq n/2$ and $n/2 < j \leq n$:

$$\begin{pmatrix} C_{1,1}\ C_{1,2} \\ C_{2,1}\ C_{2,2} \end{pmatrix} \cdot \begin{pmatrix} A(i) \\ -A(i) + e_i \end{pmatrix} = m_i \cdot \begin{pmatrix} A(i) \\ -A(i) + e_i \end{pmatrix} \tag{4.1}$$

$$\begin{pmatrix} C_{1,1}\ C_{1,2} \\ C_{2,1}\ C_{2,2} \end{pmatrix} \cdot \begin{pmatrix} -A(j - n/2) - e_{j-n/2} \\ A(j - n/2) \end{pmatrix} = m_j \cdot \begin{pmatrix} -A(j - n/2) - e_{j-n/2} \\ A(j - n/2) \end{pmatrix}. \tag{4.2}$$

After evaluating the matrix multiplication on the left side, the equality (4.1) will yield (4.3) and (4.4), and the equality (4.2) will yield (4.5) and (4.6):

$$(C_{1,1} - C_{1,2}) \cdot A(i) + C_{1,2} \cdot e_i = m_i \cdot A(i) \tag{4.3}$$

$$(C_{2,1} - C_{2,2}) \cdot A(i) + C_{2,2} \cdot e_i = -m_i \cdot A(i) + m_i \cdot e_i \tag{4.4}$$

$$(-C_{1,1} + C_{1,2}) \cdot A(j - n/2) - C_{1,1} \cdot e_{j-n/2} = -m_j \cdot A(j - n/2) - m_j \cdot e_{j-n/2} \tag{4.5}$$

$$(-C_{2,1} + C_{2,2}) \cdot A(j - n/2) - C_{2,1} \cdot e_{j-n/2} = m_j \cdot A(j - n/2). \tag{4.6}$$

We next compute a sum of (4.3) and (4.4), obtaining (4.7), and the sum of (4.5) and (4.6) that will yield (4.8):

$$(C_{1,1} - C_{1,2} + C_{2,1} - C_{2,2}) \cdot A(i) + (C_{1,2} + C_{2,2}) \cdot e_i = m_i \cdot e_i \tag{4.7}$$

$$(-C_{1,1} + C_{1,2} - C_{2,1} + C_{2,2}) \cdot A(j - n/2) - (C_{1,1} + C_{2,1}) \cdot e_{j-n/2} = -m_j \cdot e_{j-n/2}. \tag{4.8}$$

We now set $j = i + n/2$ and sum the equalities (4.7) and (4.8):

$$(-C_{1,1} + C_{1,2} - C_{2,1} + C_{2,2}) \cdot e_i = (m_i - m_{i+n/2}) \cdot e_i. \tag{4.9}$$

Let $D = (-C_{1,1} + C_{1,2} - C_{2,1} + C_{2,2})$. We see that for every $1 \leq i \leq n/2$, the i-th column of D is the i-th canonical unit vector, multiplied by a difference of two eigenvalues of C. Thus the matrix D, which can be computed using only the ciphertext, is a diagonal matrix, whose entries on the diagonal leak differences of eigenvalues of C. This can be used to circumvent the computation of roots in $\mathbb{Z}_N$, and to extract the eigenvalues easily.

Extracting the plaintext vector. Let $\delta_i = D(i, i)$ denote the i-th entry on the diagonal of D. We consider the characteristic polynomial $p(x) = \mathrm{CharPoly}(C)(x)$ of the ciphertext C. Note that the roots of $p(x)$ are the elements of the plaintext vector m. We define n new polynomials $p_{-i}(x)$ and $p_i(x)$ for $1 \leq i \leq n/2$ as $p_{-i}(x) = p(x - \delta_i)$ and $p_i(x) = p(x + \delta_i)$.

For any $1 \leq i \leq n/2$, m_i must be a root of $p_{-i}(x)$. Thus m_i will also be a root of $\gcd(p(x), p_{-i}(x))$. If there are no $(j, k) \neq (i, i + n/2)$ for which $m_j - m_k = \delta_i$, then $p(x)$ and $p_{-i}(x)$ can only have a single root in common, and $\gcd(p(x), p_{-i}(x))$ will be of degree one, allowing to compute m_i easily. Similar applies with $p_i(x)$ and $m_{i+n/2}$.

```
Algorithm DecAttack(N, C)
    D ← (−C_{1,1} + C_{1,2} − C_{2,1} + C_{2,2})
    p(x) = CharPoly(C)
    for i ← 1 to n/2 do
        p_{−i}(x) = p(x − D(i, i));   p_i(x) = p(x + D(i, i))
        q(x) ← gcd(p(x), p_{−i}(x));   q′(x) ← gcd(p(x), p_i(x))
        if gcd computation failed due to bad division then
            return factors of N
        else if deg(q(x)) > 1 or deg(q′(x)) > 1 then
            abort
        end if
        m_i ← −q_0/q_1;   m_{i+n/2} ← −q′_0/q′_1
        if division failed then
            return factors of N
        end if
    end for
    return (m_1, ..., m_n)
end Algorithm
```

Fig. 4. Decryption attack on the symmetric FHE scheme sEMORE.

This allows us to mount the decryption attack described in Figure 4. We give a toy example of the attack in Section A.

Complexity and success probability. The complexity of the attack is dominated by the iterated computation of the gcd of two polynomials over $\mathbb{Z}_N$ of degree n, so the complexity is bounded by $O(n^3 \cdot \log(N)^2)$ binary operations.

The attack can abort due to two reasons. Either there is a division by an integer that is not coprime with N, or one of the computed polynomials $q(x)$ or $q'(x)$ has a degree higher than one. In the former case, we succeed, because after factoring N, computing the eigenvalues of any ciphertext will become easy thanks to the Chinese remainder theorem.

If the plaintexts are indeed distributed uniformly (or else with a high min-entropy), the latter reason for an abort is highly unlikely. More precisely, the probability that the attack fails is at most the probability that there exist $1 \leq i, j, \ell \leq n$ such that $(i, i + n/2) \neq (j, \ell)$ and $(m_i - m_{i+n/2}) = (m_j - m_\ell)$. Under the mild assumption that all plaintext components m_i are distributed identically and independently, the probability of such a collision with fixed i, j, ℓ is at most $p_{\max}$, Thus, the probability that any such i, j, k exist is (rather loosely) upper bounded by $n^3 \cdot p_{\max}$, where $p_{\max} = \max_\mu \Pr[m_i = \mu]$. The probability of success of this attack is then at least $1 - n^3 \cdot p_{\max}$.

We can generalize the attack by requesting several ciphertexts $C^1, \ldots, C^r$ at the beginning of the experiment, and applying the attack from Figure 4 to each C^i independently. If we succeed for at least one of the ciphertexts C^i, the knowledge of the eigenvalues of C^i lets us compute K, which in turn allows us to decrypt all remaining ciphertexts.

The success probability of the generalized attack is at least $1 - (n^3 \cdot p_{\max})^r$. In order for this strategy to succeed with a probability of at least $1/2$, we must have $|\log(p_{\max})| > 3\log(n)$ and we would have to use

$$r \geq \frac{-\log(2)}{3\log(n) + \log(p_{\max})}$$

If the distribution of plaintexts is such that the probability p_{max} is high, this estimation becomes meaningless.

4.1 Attacking EMORE with High $p_{\max}$.

If $p_{\max}$ is high, EMORE will not offer any meaningful security, as we will be able to easily verify whether a ciphertext C's underlying plaintext matrix contains any plaintext element μ that is very

likely to occur by checking if $p(\mu) = 0$ for $p(x) = \text{CharPoly}(C)$. This observation can be used to extend the attack from Figure 4 and increase its success probability if $p_{\max}$ is high, at the cost of increased computational complexity.

More precisely, we select a positive constant k for which the set $\mathcal{M}_{\mathsf{hi}} = \{\mu \in \mathbb{Z}_N | \Pr[m_i = \mu] \geq 1/k \cdot n^3\}$ will be non-empty. Instead of simply aborting if the condition on line 9 of Figure 4 evaluates as true, we iterate through all $\mu \in \mathcal{M}_{\mathsf{hi}}$, and for each μ check if it is a root of either the current $q(x)$ or the current $q'(x)$, which would imply that μ is one of the plaintext elements. For each such μ we further check whether $p(\mu - D(i,i)) = 0$ (respectively $p(\mu + D(i,i)) = 0$), which would further imply that $\mu - D(i,i)$ (respectively $\mu + D(i,i)$) is also among the plaintext-vector elements. We store all thus identified *distinct* plaintext elements. At the end of the main loop, we keep dividing the characteristic polynomial $p(x)$ by monomials constructed using the identified plaintext elements, and abort only if we cannot completely decompose $p(x)$.

Complexity and success probability. The complexity of the attack is now increased. In the worst case, each of the $n/2$ iterations of the main loop consists of computing the polynomial GCDs, and iterating over all elements of $\mathcal{M}_{\mathsf{hi}}$. Each iteration of the inner loop is dominated by the evaluation of two polynomials of degree no greater than n in some value. As we have $|\mathcal{M}_{\mathsf{hi}}| \leq k \cdot n^3$ by definition, the worst case complexity of the extended attack is

$$n/2 \cdot (n^2 \cdot O(\log(N)^2) + k \cdot n^3 \cdot n \cdot O(\log(N)^2))) = O(k \cdot n^5 \cdot \log(N)^2).$$

To analyze the probability of success of the extended attack, we first note that we surely succeed if, for every $i = 1, \ldots, n/2$, in the i^{th} iteration of the main loop, either $\deg(\gcd(q(x), q'(x))) = 1$, or else if we have $m_i = \mu$ and $m_{i+n/2} = \mu - D(i,i)$ (respectively $m_{i+n/2} = \mu$ and $m_i = \mu + D(i,i)$) for one of the values $\mu \in \mathcal{M}_{\mathsf{hi}}$ (then we will surely have $q(\mu) = 0$ and $p(\mu - D(i,i)) = 0$ or $q'(\mu) = 0$ and $p(\mu + D(i,i)) = 0$).

We let $\mathsf{E}(i,j,\ell)$ denote the event $\exists (j,\ell) \neq (i, i+n/2) : D(i,i) = m_j - m_\ell$, and let $\mathsf{F}(i)$ denote the event $(m_i \notin \mathcal{M}_{\mathsf{hi}}) \wedge (m_{i+n/2} \notin \mathcal{M}_{\mathsf{hi}})$. By using the union bound on the iterations of the main loop, we get

$$\begin{aligned}
\Pr[\mathsf{success}] \geq & 1 - n \cdot \Pr[\mathsf{E}(i,j,\ell) \wedge \mathsf{F}(i)] \\
\geq & 1 - n \cdot \Pr[\mathsf{E}(i,j,\ell) | \mathsf{F}(i)] \cdot \Pr[\mathsf{F}(i)] \\
\geq & 1 - n \cdot \left(\sum_{(j,\ell) \neq (i,i+n/2)} \sum_{\mu \in \mathbb{Z}_N} \Pr[m_i = m_j - m_\ell + \mu | \mathsf{F}(i)] \cdot \Pr[m_{i+n/2} = \mu | \mathsf{F}(i)] \right) \cdot \Pr[\mathsf{F}(i)] \\
\geq & 1 - n \cdot \left(n^2 \cdot \frac{1}{k \cdot n^3} \cdot \sum_{\mu \in \mathbb{Z}_N} \Pr[m_{i+n/2} = \mu | \mathsf{F}(i)] \right) \\
\geq & 1 - \frac{1}{k}.
\end{aligned}$$

The extended attack works with any k for which $\mathcal{M}_{\mathsf{hi}}$ is non-empty, and we can use the parameter k to fine-tune the trade-off between the complexity and success probability.

We further see that the worst-case complexity of the extended attack becomes practical when $p_{\max} \approx 1/n^3$ or higher, which is exactly where the loose estimation of the attack described in Figure 4 falls off Thus sEMORE yields to a ciphertext-only decryption attack under *any* possible circumstances.

4.2 Experimental Verification

We implemented the attack in the SAGE mathematical software [10], to verify its correctness. We conducted all experiments using a sEMORE instance with a random 2048-bit modulus and $n = 16$.

When the messages are uniform in $\mathbb{Z}_M$, the attack practically always succeeds, as expected. We tested this with 2^7 independent iterations of the attack, and all of them succeeded.

We further investigated the success rate of the simple attack with low-entropy distributions, for which the loose lower bound on the success probability is meaningless. We carried out two sets of experimental measurements, each set with a different type of distribution of the plaintext. In the first set, we sampled each plaintext element using a uniform distribution over a subset of $\mathbb{Z}_N$ of varying size. In the second set, we used a rounded Gaussian distribution with varying parameter σ. The results of the measurements are depicted in Figure 5. We can see that the lower bound of the success probability is indeed too restrictive, as the basic attack does succeed with non-zero probability initially, but the success probability quickly drops as $p_{\max}$ increases.

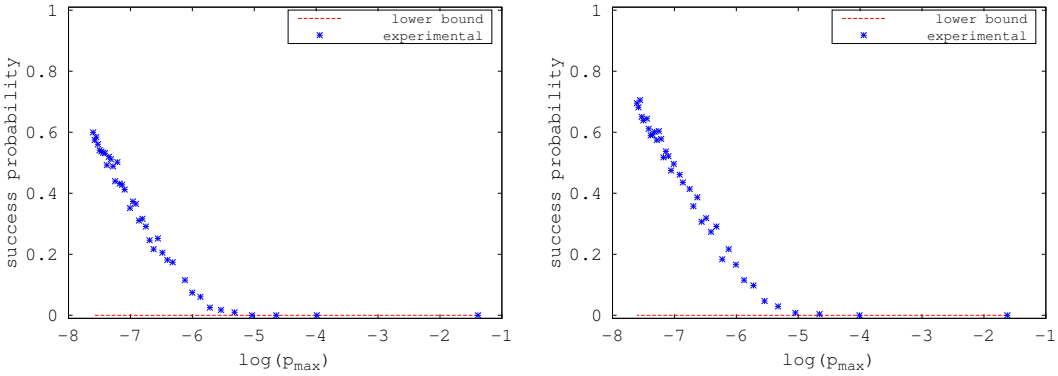

Fig. 5. The empirically measured success probability of the attack from Figure 4, compared to the predicted lower bound for an instance of sEMORE with a 2048-bit modulus, $n = 16$, and plaintext elements distributed uniformly in a subset of $\mathbb{Z}_N$ (**left**) or according to a rounded Gaussian distribution (**right**). Here $p_{\max} = \max_{\mu \in \mathbb{Z}_N}(\Pr[m_i = \mu])$.

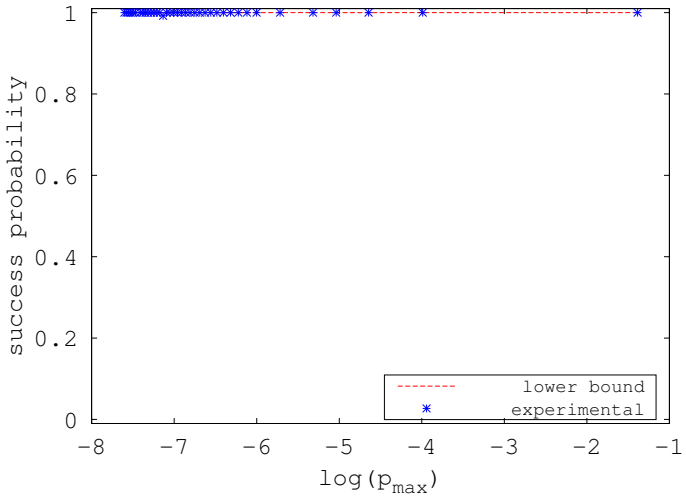

Fig. 6. The empirically measured success probability of the extended attack described in Section 4.1, compared to the predicted lower bound for an instance of sEMORE with a 2048-bit modulus, $n = 16$, and plaintext elements distributed uniformly in a subset of $\mathbb{Z}_N$. Here $p_{\max} = \max_{\mu \in \mathbb{Z}_N}(\Pr[m_i = \mu])$.

We also implemented the extended attack, and empirically measured it success probability on an instance of sEMORE with 2048-bit modulus, $n = 16$, and plaintexts distributed uniformly in subsets of $\mathbb{Z}_{\mathbb{N}}$ of varying size. In the attack, we set $k = 1$, so the set $\mathcal{M}_{\mathsf{hi}}$ always covered all elements of $\mathbb{Z}_N$ that occurred with non-zero probability. The results of the measurements can be found in Figure 6.

We see, that in the interval of $p_{\max}$, in which the basic attack stops being efficient, the extended attack already works flawlessly. This further confirms our assertion, that sEMORE and EMORE are completely broken.

5 Conclusion

In this paper, we analyze the security of Enhanced MORE, a reincarnation of the MORE symmetric FHE scheme. Even though MORE was badly broken, the authors of EMORE not only reused the linear encryption method, but introduced additional vulnerabilities. We make this explicit, by mounting a decryption attack. We believe it important to demonstrate concrete, and precisely described attacks on weak schemes like EMORE, so that their propagation in the literature, and their potential use in practice is limited.

References

1. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Proceedings of the 31st annual conference on Advances in cryptology*, CRYPTO'11, pages 505–524, Berlin, Heidelberg, 2011. Springer-Verlag.
2. Aldar C-F. Chan. Symmetric-key homomorphic encryption for encrypted data processing. In *Proceedings of the 2009 IEEE international conference on Communications*, ICC'09, pages 774–778, Piscataway, NJ, USA, 2009. IEEE Press.
3. Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.
4. Craig Gentry. *A fully homomorphic encryption scheme.* PhD thesis, Stanford, CA, USA, 2009. AAI3382729.
5. Khalil Hariss, Hassan Noura, and Abed Ellatif Samhat. Fully enhanced homomorphic encryption algorithm of MORE approach for real world applications. *J. Inf. Sec. Appl.*, 34:233–242, 2017.
6. Aviad Kipnis and Eliphaz Hibshoosh. Efficient methods for practical fully homomorphic symmetric-key encrypton, randomization and verification. *IACR Cryptology ePrint Archive*, 2012:637, 2012.
7. Hassan Noura and Damien Couroussé. HLDCA-WSN: homomorphic lightweight data confidentiality algorithm for wireless sensor network. *IACR Cryptology ePrint Archive*, 2015:928, 2015.
8. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.
9. R L Rivest, L Adleman, and M L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation, Academia Press*, pages 169–179, 1978.
10. W. A. Stein et al. *Sage Mathematics Software (Version 6.8).* The Sage Development Team, 2015. `http://www.sagemath.org`.
11. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43, 2010.
12. Damian Vizár and Serge Vaudenay. Cryptanalysis of chosen symmetric homomorphic schemes. *Studia Scientiarum Mathematicarum Hungarica*, 52(2):288–306, 2015.
13. Liangliang Xiao, Osbert Bastani, and I-Ling Yen. An efficient homomorphic encryption protocol for multi-user systems. *IACR Cryptology ePrint Archive*, 2012:193, 2012. informal publication.

A Toy Example of the Attack

In this section, we define a toy instance of the simplified Enhanced MORE encryption scheme, and mount the attack described in Figure 4.

We let $p = 11$ and $q = 13$, so we have $N = 143$. We set the dimension of plaintext space to $n = 4$. We next run the key generation algorithm, sample a random invertible matrix A, and compute K, obtaining:

$$A = \begin{pmatrix} 92 & 45 \\ 92 & 62 \end{pmatrix} \qquad \text{and} \qquad K = \begin{pmatrix} A & A+I \\ A-I & A \end{pmatrix} = \begin{pmatrix} 92 & 45 & 93 & 45 \\ 92 & 62 & 92 & 63 \\ 91 & 45 & 92 & 45 \\ 92 & 61 & 92 & 62 \end{pmatrix}.$$

Then we sample a uniform plaintext vector from $\mathbb{Z}_{143}$, and place it on the diagonal of a matrix M. We run the encryption $C = \mathcal{E}(K, N, M)$. We have:

$$M = \begin{pmatrix} 32 & 0 & 0 & 0 \\ 0 & 52 & 0 & 0 \\ 0 & 0 & 36 & 0 \\ 0 & 0 & 0 & 25 \end{pmatrix} \qquad \text{and} \qquad C = \begin{pmatrix} 25 & 74 & 50 & 2 \\ 43 & 23 & 104 & 99 \\ 72 & 32 & 43 & 104 \\ 10 & 103 & 92 & 54 \end{pmatrix}.$$

We now compute the matrix D that will leak the differences of the plaintext values. For this, we first partition the ciphertext C into four submatrices:

$$C_{1,1} = \begin{pmatrix} 25 & 74 \\ 43 & 23 \end{pmatrix} \qquad C_{1,2} = \begin{pmatrix} 50 & 2 \\ 104 & 99 \end{pmatrix}$$
$$C_{2,1} = \begin{pmatrix} 72 & 32 \\ 10 & 103 \end{pmatrix} \qquad C_{2,2} = \begin{pmatrix} 43 & 104 \\ 92 & 54 \end{pmatrix}.$$

Using these, we compute the matrix D, using the formula $D = (-C_{1,1} + C_{1,2} - C_{2,1} + C_{2,2})$:

$$D = \begin{pmatrix} 139 & 0 \\ 0 & 27 \end{pmatrix}.$$

Next, we compute the characteristic polynomial $p(x)$ of C:

$$p(x) = x^4 + 141 \cdot x^3 + 109 \cdot x^2 + 73 \cdot x + 104$$

Using the diagonal entries of D, we compute the polynomials $p_{-i}(x) = p(x - D(i,i))$ and $p_i(x) = p(x + D(i,i))$ for $i \in \{1, 2\}$. For each i, we then compute $q(x) = \gcd(p_{-i}(x), p(x))$ and $q'(x) = \gcd(p_i(x), p(x))$. These will be polynomials of degree one, that will allow us to recover m_i as $-q_0 \cdot q_1^{-1}$ and m_{i+2} as $-q_0' \cdot {q_1'}^{-1}$. We follow through the computation for $i = 1$:

$$p_{-1}(x) = x^4 + 14 \cdot x^3 + 38 \cdot x^2 + 104 \cdot x + 123, \quad \text{and we have} \quad q(x) = 14 \cdot x + 124$$

and

$$p_1(x) = x^4 + 125 \cdot x^3 + 86 \cdot x^2 + 136 \cdot x + 81, \quad \text{and we have} \quad q'(x) = 129 \cdot x + 75.$$

We can verify that

$$M(1,1) = 32 \equiv -124 \cdot 14^{-1} \pmod{143}$$

and

$$M(3,3) = 36 \equiv -75 \cdot 129^{-1} \pmod{143}.$$

For $i = 2$, we have

$$p_{-2}(x) = x^4 + 33 \cdot x^3 + 69 \cdot x^2 + 27 \cdot x + 39, \quad \text{and we have} \quad q(x) = 36 \cdot x + 130$$

and

$$p_2(x) = x^4 + 106 \cdot x^3 + 31 \cdot x^2 + 94 \cdot x + 38, \quad \text{and we have} \quad q'(x) = 107 \cdot x + 42.$$

We can verify that

$$M(2,2) = 52 \equiv -130 \cdot 36^{-1} \pmod{143}$$

and

$$M(4,4) = 25 \equiv -42 \cdot 107^{-1} \pmod{143}.$$

Thus the attack successfully recovered all entries of the plaintext vector.