On Some Weaknesses in the Disk Encryption Schemes EME and EME2

Cuauhtemoc Mancillas-López, Debrup Chakraborty and Francisco Rodríguez-Henríquez

Department of Computer Science, CINVESTAV-IPN, Av. IPN 2508, Col: San Pedro Zacatenco, Mexico City 07360, Mexico
mancilla@computacion.cs.cinvestav.mx, debrup@cs.cinvestav.mx, francisco@cs.cinvestav.mx

Abstract. Tweakable enciphering schemes are a certain type of blockcipher mode of operation which provide security in the sense of a strong pseudo-random permutation. It has been proposed that these types of modes are suitable for in-place disk encryption. Currently there are many proposals available for these schemes. EME is one of the efficient candidate of this category. EME2 is a derivative of EME which is currently one of the candidates of a draft standard for wide block modes by the IEEE working group on storage security. We show some weakness of these two modes assuming that some side channel information is available.

Key words: Tweakable Enciphering Schemes, Disc Encryption, Modes of operation, EME

1 Introduction

A mode of operation is a specific way to use a block-cipher to encrypt arbitrary long messages. Several classes of modes have been proposed in the last few years to provide different security services for various kind of scenarios. In particular, a Tweakable Enciphering Scheme (TES), is a length preserving enciphering scheme that provides security in the sense of a strong pseudo-random permutation. This means that an efficient adversary given access to the outputs of the TES encryption and decryption algorithms, will not be able to distinguish them from a random permutation and its inverse with non-negligible probability. For length preserving encryption this is the highest form of security that can be guaranteed. Some of the most important TES proposals published to date can be found in [9, 10, 7, 19, 6, 15, 5, 8, 18]. In [9], it was suggested that TES is the natural choice for the application of in place disk encryption.

TES can be used in low-level disk encryption by placing a hardware/software realization of this scheme in the disk controller, where it will be in charge of performing both, run-time encryption of the sectors to be written and run-time decryption of the sectors to be read. Thus, the data contained in the disk sectors remains at all times in encrypted form. Note that under this model, a TES has no knowledge of the high level partitions of the disk, such as files and directories. Moreover, due to the nature of the application, a secure length preserving encryption scheme (such as a TES) is required.

Since nowadays encrypting data recorded in bulk storage devices like hard-disks and flash memory portable devices has become a strategic problem, the design of secure TES has attracted considerable attention. Currently, there is also an on-going standardization effort carried out by the IEEE security in storage working group, that has produced a draft standard for wide-block encryption making it available for public scrutiny at [1].

Among the different TES found in the literature EME [10] is one of the efficient candidates, whereas EME2 [1], a derivative of EME, is currently a candidate of the draft standard proposed by [1]. Both, EME and EME2 work on n bit binary strings which are viewed as elements of the finite field $\mathrm{GF}(2^n)$. Hence, the n-bit strings can be treated as polynomials of degree less than n with coefficients in $\mathrm{GF}(2)$. The addition of the elements of such a field is defined as the regular polynomial addition and the multiplication is done as multiplication of two polynomials modulo a fixed irreducible polynomial $\tau(x)$ of degree n. Let $L \in \{0,1\}^n$. One operation of particular interest is the product xL, which means multiplication of the polynomial L by the monomial x modulo the irreducible polynomial representing the field. We shall further call this operation as *xtimes*. If the irreducible polynomial $\tau(x)$ representing the field is primitive then the monomial x would be a generator of the cyclic group formed by the non-zero elements of $GF(2^n)$. This means that $x, x^2, x^3, \ldots, x^{2^n-1}$ will all be distinct. This property has found many cryptographic applications. In particular in the modes EME and EME2 a sequence of the form x^iL, $1 \leq i < 2^{n-1}$, for some unknown L is used to mask certain data streams. These masks can be efficiently generated by repeatedly applying *xtimes* on L. The operation xL can be performed quite efficiently, but as we shall discuss in the rest of this paper, if not carefully crafted, that operation may leak information regarding L.

Our Contribution: We analyze the mode of operations EME and EME2 and point out certain weaknesses in the modes if side channel information is available. We use a previously reported observation that *xtimes* leaks much information in terms of timings and power utilization. Assuming the insecurity of the *xtimes* operation we systematically develop attacks on EME and EME2. Specifically we show that for both EME and EME2, given side channel information and access to the encryption algorithm, an adversary can efficiently mount distinguishing attacks. For EME we show a stronger attack, where an adversary can decrypt any ciphertext C without querying the decryption algorithm at C with very high probability. Also for EME, given access to the encryption algorithm the adversary can produce ciphertext of any given plaintext P, without querying the encryption algorithm at P.

The possible side channel weakness of EME and two of its derivatives EME$^+$ and EME* were pointed out in an appendix of [16], but the true vulnerabilities were not analyzed in details. We claim that our analysis is substantially different from that in [16]. In [11], Antoine Joux presented a cryptanalysis of EME mode. Joux's cryptanalysis operates on a flawed version of EME (this version can be found in [17], which is significantly different from the final mode as reported in [10]). Right after Joux attack, the EME mode was repaired and the attacks by Joux are not applicable to the corrected version of EME which has been reported in [10]. We note that, the attacks in [11] does not require any access to side channel information and that our attacks are un-related to those reported in [11].

2 Adversaries with Access to Side Channel Information

Modern security guarantees provided by security proofs, are based on complexity theoretic arguments and assumptions on abstract notions of computations. For security definitions, one models the adversary as a probabilistic polynomial time algorithm with access to input/outputs of a protocol. The adversary knows the algorithm which produces the outputs for his chosen inputs, but has no knowledge of a secret quantity which is called the key. Moreover, as the computation model is assumed to be an abstract one, hence the state of the algorithm is also assumed to be invisible to an adversary, who cannot know the branches taken, subroutines called, etc. by the algorithm during its execution.

However, this is not a realistic scenario, since computations must be done on a physical device, and that device can leak various kinds of information. This can motivate attacks from the adversary who by applying measurements on the leaked information may be able to gain access to sensitive data related to the computation being performed. In the past few years there have been numerous studies which point out insecurity of many established cryptographic algorithms if certain side channel information is available to the adversary.

Researchers have considered the possibilities of using different types of side channel information for breaking crypto-systems. Some of the categories of side channel attacks are timing attacks [13], power analysis attacks, differential power analysis attacks [14], electromagnetic radiation attacks[2], fault attacks [3] etc. These attacks utilize the leakages that are associated with any type of computation that takes place in a physical device.

3 Notations

In what follows we shall mean by $\mathsf{msb}(L)$ the most significant bit of a binary string L. For two strings L and M, $L||M$ will denote the concatenation of L and M. By $L << k$ we shall mean left-shift of L by k bits. $\mathsf{take}_k(L)$ will mean the k most significant bits of L. If b is a bit, then $\bar{b}$ will mean the complement of b. We shall treat n bit strings as polynomials with coefficients in $GF(2)$. Thus, an n bit string can be treated as an element in $GF(2^n)$. For two n bit strings X and Y, $X \oplus Y$ will denote the addition in the field, which can be realized by a bitwise xor of the strings X and Y. Multiplication of two n bit strings would be represented as XY which will mean the multiplication of the two corresponding $(n-1)$ degree polynomials modulo an n degree irreducible polynomial $\tau(x)$. Particularly, for $X \in \{0,1\}^n$, xX would mean the multiplication of the polynomials x and X modulo $\tau(x)$. By an n-bit block cipher we shall mean a function $E : \mathcal{K} \times$

$\{0,1\}^n \rightarrow \{0,1\}^n$, where $\mathcal{K} \neq \emptyset$ is the key space and for any $K \in \mathcal{K}$, $E(K,.)$ is a permutation. We shall often write $E_K(.)$ instead of $E(K,.)$.

4 Side Channel Weakness in the *xtimes* operation

The implementation of the *xtimes* operation is very efficient. Let $\tau(x)$ denote the n degree irreducible polynomial representing the field $GF(2^n)$, let Q be the n bit representation of the polynomial $\tau(x) \oplus x^n$. Then xL can be realized by the algorithm in Fig. 1.

```
Algorithm xtimes(L)
1. $b \leftarrow \mathsf{msb}(L)$
2. $L \leftarrow L << 1$
3. if $b = 1$,
4.     $L \leftarrow L \oplus Q$
5. return $L$
```

Fig. 1. The algorithm *xtimes*

This is the most efficient (and the usual) way of implementing *xtimes* where the basic operations involved are a left shift and a conditional xor. As it is obvious from the algorithm that line 4 gets executed only when the most significant bit (MSB) of L is a one. The power utilization in this algorithm would be different in the cases where $\mathsf{msb}(L) = 0$ and $\mathsf{msb}(L) = 1$. This difference of power consumption if measured can give information regarding the MSB of L.

Above described weakness of the *xtimes* operation is widely known. The *xtimes* operation on an n bit string can also be implemented by linear feedback shift registers (LFSR). The vulnerability of LFSRs to side channel attacks has been extensively studied and also there are experimental evidences that such systems leak a lot of information [12][4]. Thus with the support of the evidence as found in the literature and without going into technical/experimental details of side channel attacks in the rest of this paper we shall make the following assumption:

Assumption 1 *If the operation xL is implemented according to the algorithm xtimes as shown in Fig. 1 then the MSB of L can be obtained as a side channel information.*

Repeated application of *xtimes* on L can reveal much more information about L. In particular, based on Assumption 1 we can state the following proposition.

Proposition 1 *If xtimes is applied k ($k \leq n$) times successively on L then the k most significant bits of L can be recovered.*

```
Algorithm Recover($Q, k$)
1.  $D \leftarrow < d_{n-1}, d_{n-2}, \ldots, d_0 > \leftarrow 0^n$;
2.  $B \leftarrow$ Empty String;
3.  for $i = 1$ to $k$,
4.      $b \leftarrow \mathsf{SC}(L_1 \leftarrow xL)$ ;($\mathsf{SC}(L_1 \leftarrow xL)$ gives the MSB of $L$)
5.      if $d_{n-1} = 0$,
6.          $B \leftarrow B||b$;
7.      else,
8.          $B \leftarrow B||\bar{b}$
9.      end if
10.     $D \leftarrow D << 1$;
11.     if $b = 1$,
12.         $D \leftarrow D \oplus Q$
13.     end if
14.     $L \leftarrow L_1$
15. end for
16. return B
```

Fig. 2. The algorithm to recover k bits of L

In lieu of proof of the above proposition we present the procedure to recover the k bits of L in the algorithm in Figure 2.

The algorithm *Recover* as shown in Fig. 2 takes in as input the number of bits to be recovered k along with another n bit string Q which encodes the polynomial $x^n \oplus \tau(x)$. The algorithm also have access to some side-channel information, which gives it the information of the MSB of L, this is shown as $\mathsf{SC}(L_1 \leftarrow xL)$ in line 4. The algorithm, initializes an n bit string D with all zeros (d_i represents the i-th bit of D in the algorithm) and initializes B by an empty string. The output of the algorithm is a k bit string B whose bits would be the same as the k most significant bits of L. It is not difficult to see the correctness of the algorithm. Note that we simulate in D the same changes that takes place in L. After executing the i-th iteration of the for loop we obtain in line 4 the most significant bit of $x^{i-1}L$. This bit would be equal to the i-th bit of L if the MSB of D is zero, otherwise the i-th bit of L would be the complement of the most significant bit of $x^{i-1}L$.

5 Security of Tweakable Enciphering Schemes

In this section we shall review the security requirements of tweakable enciphering schemes. The material in this section is based on [9, 10].

A tweakable enciphering scheme is a function $\mathbf{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$, where $\mathcal{K} \neq \emptyset$ and $\mathcal{T} \neq \emptyset$ are the key space and the tweak space respectively. The message and the cipher spaces are $\mathcal{M}$. We shall write $\mathbf{E}_K^T(.)$ instead of $\mathbf{E}(K, T, .)$. The inverse of an enciphering scheme is $\mathbf{D} = \mathbf{E}^{-1}$ where $X = \mathbf{D}_K^T(Y)$ if and only if $\mathbf{E}_K^T(X) = Y$.

Let $\mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ denote the set of all functions $\boldsymbol{\pi} : \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$ where $\boldsymbol{\pi}(\mathcal{T}, .)$ is a length preserving permutation. Such a $\boldsymbol{\pi} \in \mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ is called a tweak indexed permutation.

An adversary A is a probabilistic algorithm which has access to some oracles and which outputs either 0 or 1. Oracles are written as superscripts. The notation $A^{\mathcal{O}_1, \mathcal{O}_2} \Rightarrow 1$ denotes the event that the adversary A, interacts with the oracles $\mathcal{O}_1, \mathcal{O}_2$, and finally outputs the bit 1. In what follows, the notation $X \overset{\$}{\leftarrow} \mathcal{S}$, will denote the event of choosing X uniformly at random from the finite set $\mathcal{S}$.

For a tweakable enciphering scheme $\mathbf{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$, we define the advantage an adversary A has in distinguishing $\mathbf{E}$ and its inverse from a random tweak indexed permutation and its inverse in the following manner.

$$\begin{aligned}\mathbf{Adv}_{\mathbf{E}}^{\pm\widetilde{\mathrm{prp}}}(A) = \Big| &\Pr\left[K \overset{\$}{\leftarrow} \mathcal{K} : A^{\mathbf{E}_K(.,.), \mathbf{E}_K^{-1}(.,.)} \Rightarrow 1\right] \\ &- \Pr\left[\boldsymbol{\pi} \overset{\$}{\leftarrow} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M}) : A^{\boldsymbol{\pi}(.,.), \boldsymbol{\pi}^{-1}(.,.)} \Rightarrow 1\right]\Big| . \qquad (1)\end{aligned}$$

Here, $\boldsymbol{\pi} \overset{\$}{\leftarrow} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ means that for each ℓ such that $\{0,1\}^{\ell} \subseteq \mathcal{M}$ and $T \in \mathcal{T}$ we choose a tweakable random permutation π^{T} from $\mathrm{Perm}(\ell)$ independently.

A TES is considered to be secure if $\mathbf{Adv}_{\mathbf{E}}^{\pm\widetilde{\mathrm{prp}}}(A)$ is small for all efficient adversaries. The definition of advantage of an adversary A above suggests the task of the adversary is to distinguish the output of an algorithm from random outputs. The adversary is given oracle access to both the encryption and decryption algorithms, which means that the adversary is free to see ciphertexts corresponding to the plaintexts of his choice and also decryptions of ciphertexts of his choice. Such an adversary is called a chosen-ciphertext (CCA) adversary Moreover, the adversary is at liberty to choose the tweaks.

The definition of advantage of an adversary A above does not take into account the information leaked by a physical implementation of the algorithm. It is practical to consider CCA adversaries, but in real life an adversary may have more information than only access to ciphertexts (plaintexts) of his/her chosen plaintexts (ciphertexts). As discussed in Section 2 given access to a physical implementation of a cryptographic algorithm, the adversary may perform other kinds of measurements and thus obtain more information. In the attacks that we construct we shall consider efficient adversaries with access to side channel information. In what follows by an oracle access to an algorithm we would mean that the adversary gets outputs of the algorithm along with certain side channel information.

6 The EME Mode of Operation

ECB-Mask-ECB (EME)[10] is an efficient tweakable enciphering scheme. EME consists of two electronic code-book layers with a masking layer in between. The encryption and decryption algorithm are given in Fig. 3.

EME takes in an m block message along with a tweak T. The algorithm is self explanatory, but an important feature to note is that in the masking layer, the mask M is dependent on all the plaintext blocks and the mask is distributed to all the blocks suitably. This makes each block of ciphertext dependent on all blocks of plain texts. This is a necessary criteria for any disk encryption mode.

Fig. 3. Encryption and Decryption using EME.

```
**Algorithm** EME.Encrypt$_K^T(P)$
1.  Partition $P$ into $P_1, P_2, \ldots, P_m$
2.  $L \leftarrow xE_K(0^n)$
3.  **for** $i \leftarrow 1$ to $m$ **do**
4.      $PP_i \leftarrow x^{i-1}L \oplus P_i$
5.      $PPP_i \leftarrow E_K(PP_i)$
6.  **end for**
7.  $SP \leftarrow PPP_2 \oplus PPP_3 \oplus \ldots PPP_m$
8.  $MP \leftarrow PPP_1 \oplus SP \oplus T$
9.  $MC \leftarrow E_K(MP)$
10. $M \leftarrow MP \oplus MC$
11. **for** $i \leftarrow 2$ to $m$ **do**
12.     $CCC_i \leftarrow PPP_i \oplus x^{i-1}M$
13. **end for**
14. $SC \leftarrow CCC_2 \oplus CCC_3 \oplus \ldots CCC_m$
15. $CCC_1 \leftarrow MC \oplus SC \oplus T$
16. **for** $i \leftarrow 1$ to $m$ **do**
17.     $CC_i \leftarrow E_K(CCC_i)$
18.     $C_i \leftarrow x^{i-1}L \oplus CC_i$
19. **end for**
20. **return** $C_1, C_2, \ldots, C_m$
```

```
**Algorithm** EME.Decrypt$_K^T(C)$
1.  Partition $C$ into $C_1, C_2, \ldots, C_m$
2.  $L \leftarrow xE_K(0^n)$
3.  **for** $i \leftarrow 1$ to $m$ **do**
4.      $CC_i \leftarrow x^{i-1}L \oplus C_i$
5.      $CCC_i \leftarrow E_K^{-1}(CC_i)$
6.  **end for**
7.  $SC \leftarrow CCC_2 \oplus CCC_3 \oplus \ldots \oplus CCC_m$
8.  $MC \leftarrow CCC_1 \oplus SC \oplus T$
9.  $MP \leftarrow E_K^{-1}(MC)$
10. $M \leftarrow MP \oplus MC$
11. **for** $i \leftarrow 2$ to $m$ **do**
12.     $PPP_i \leftarrow CCC_i \oplus x^{i-1}M$
13. **end for**
14. $SP \leftarrow PPP_2 \oplus PPP_3 \oplus \ldots PPP_m$
15. $PPP_1 \leftarrow MP \oplus SP \oplus T$
16. **for** $i \leftarrow 1$ to $m$ **do**
17.     $PP_i \leftarrow E_K^{-1}(PPP_i)$
18.     $P_i \leftarrow x^{i-1}L \oplus PP_i$
19. **end for**
20. **return** $P_1, P_2, \ldots, P_m$
```

EME has some message length restrictions. If the block length of the underlying block cipher is n then EME cannot encrypt more than n blocks of messages. Also the message length should always be a multiple of n. EME is proved to be a secure tweakable enciphering scheme.

7 The Attack on EME

Note that EME uses three layers of *xtimes* operation. These operations can leak information about the internal variables L and M. Utilizing this leaked information one can attack the mode. We show two attacks. One attack is a distinguishing attack, which shows that an adversary with oracle access to only the encryption oracle of the mode and the side channel information can distinguish with probability 1 between the real oracle from the one which produces only random strings. We also show a stronger attack, where the adversary can successfully decrypt any given ciphertext C by querying the decryption oracle with ciphertexts other than C. Similarly, without knowledge of the key an adversary can produce a valid cipher text from a given plaintext P and tweak T by querying the encryption oracle (but not at P).

The main observation that makes these attacks possible is Proposition 1. From the algorithm in Fig. 3 it is clear that on encryption or decryption of m plaintext or ciphertext blocks *xtimes* is applied m times on $E_K(0^n)$ and $m-1$ times on M. This information would be crucial in mounting the attacks. For an m block query the side channel information that we are interested in is the information regarding M and $E_K(0^n)$ further we shall say that the M-side-channel and L-side-channel gives the side channel information regarding M and $E_K(0^n)$. So after an m block encryption or decryption query the M-side-channel and the L-side-channel will give the $(m-1)$ significant bits of M and m significant bits of $E_K(0^n)$ respectively.

7.1 The Distinguishing Attack

First we note down the basic steps followed by the adversary:

1. Apply an arbitrary encryption query of n blocks with an arbitrary tweak.
   - Obtain the n bits of $E_K(0^n)$ from the L-side-channel .
   - Compute $L = xE_K(0^n)$.
2. Apply an encryption query with plaintext L and tweak $x^{-1}L$. Let C be the response of this query.
3. If C is equal to $(1 \oplus x)x^{-1}L$ output EME otherwise output random.

It is easy to see why this attack works. When applying the query in step 1, the adversary recovers the n bits of $E_K(0^n)$. From line 2 of the algorithm in Fig. 3 we can see that $L = xE_K(0^n)$, thus the adversary can compute L. Following the algorithm in Fig. 3 we see that for the second query (in step 2) the value of PP_1 would be 0^n hence the value of PPP_1 would be $E_K(0^n)$. As the query consists of only one block, so values of both SP and SC would be zero. So, MP would be computed as $PPP_1 \oplus T$. Note that $T = x^{-1}L = E_K(0^n)$. So MP would be 0^n and CCC_1 would also be 0^n. Thus, we would obtain the output as

$$\begin{aligned} C &= E_K(0^n) \oplus L = E_K(0^n) \oplus xE_K(0^n) \\ &= (1 \oplus x)x^{-1}(xE_K(0^n)) = (1 \oplus x)x^{-1}L \end{aligned}$$

7.2 The Stronger Attack

Here we describe a stronger attack, in which assuming that the adversary has access to the M-side-channel and L-side-channel can decrypt any given ciphertext by querying the decryption oracle with ciphertexts other than the ciphertext in question. Before we present the attack we note down an important but obvious characteristics of EME in the proposition below:

Proposition 2 *An oracle access to the blockcipher E_K is enough to encrypt any plaintext $P_1||P_2||\dots||P_m$ with arbitrary tweak T using the EME mode of operation which uses the key K. Similarly, with an oracle access to both E_K^{-1} and E_K one can decrypt any arbitrary ciphertext $C_1, C_2, \dots, C_m$ with an arbitrary tweak T which has been produced by the EME mode of operation with key K.*

The truth of the above proposition can be easily verified from the algorithm in Fig. 3. Following the algorithm in Fig. 3, if we write the dependence of the ciphertext (resp. plaintext) with the plaintext (resp. ciphertext), then the only unknown terms would be of the form $E_K(X)$, for some X, which can be obtained by querying the oracle E_K at X. Similar argument hold for the decryption oracle.

In the attack that follows we shall show that given access to the encryption algorithm of EME along with the L-side-channel and M-side-channel, an adversary can use it as an oracle for the blockcipher $E_K()$. Similarly given an access to the decryption algorithm of EME along with the M-side-channel and L-side-channel, the adversary can use it as an oracle for E_K^{-1}. So, by Proposition 2 the adversary can compute ciphertext (plaintext) corresponding to any plaintext (ciphertext) for EME. We describe the steps undertaken by the adversary to obtain $E_K(X)$ given an oracle access to the encryption algorithm of EME in Fig. 4. We assume that the procedure $\mathbf{AdvSCA}^{\mathrm{EME}_K(.,.)}(X)$ has an access to to the EME encryption algorithm, we also assume that n (the block length of the block cipher E_K) is even [1].

```
$\mathbf{AdvSCA}^{\mathrm{EME}_K(.,.)}(X)$

1. Apply an arbitrary encryption query of $n$ blocks with
   an arbitrary tweak.
     – Obtain the $n$ bits of $L$ using the L-side-channel .
2. Apply an encryption query with tweak $X$ and the
   following plaintext:
       $\underbrace{Y \oplus L, Y \oplus xL, \dots, Y \oplus x^{n-1}L}_{n \text{ blocks}}$.
   where $Y \in \{0,1\}^n$ is chosen arbitrarily.
     – Obtain $(n-1)$ significant bits of $M$ using the
       M-side-channel , call this as $M_1$.
3. Output $M_1 \oplus \mathsf{take}_{n-1}(X)$
```

Fig. 4. The side channel adversary with access to EME encryption algorithm producing $n-1$ bits of $E_K(X)$ for arbitrary $X \in \{0,1\}^n$

[1] This assumption is not strong, as we do not know of any blockcipher whose block length is odd.

Proposition 3 *The procedure* $\mathbf{AdvSCA}^{\mathrm{EME}_K(.,.)}(X)$ *as shown in Fig. 4 outputs* $\mathsf{take}_{n-1}(E_K(X))$.

Proof. In step 1 the information regarding L is obtained. For the query in step 2 according to the algorithm (see Fig. 3), we obtain $PP_i = Y$. Thus,

$$\begin{aligned} SP &= PPP_2 \oplus PPP_3 \oplus \cdots \oplus PPP_n \\ &= E_K(Y) \oplus \underbrace{(E_K(Y) \oplus \cdots \oplus E_K(Y))}_{n-2} = E_K(Y) \end{aligned}$$

So we have,

$$MP = PPP_1 \oplus SP \oplus X = E_K(Y) \oplus E_K(Y) \oplus X = X$$

Thus, $MC = E_K(X)$, and $M = MP \oplus MC = X \oplus E_K(X)$. Hence $M \oplus X = E_K(X)$. In step 2, only $n-1$ bits of M would be obtained, hence the procedure $\mathbf{AdvSCA}^{\mathrm{EME}_K(.,.)}(X)$ outputs $n-1$ bits of $E_K(X)$. □

Now we design another adversary which on given access to the EME encryption algorithm and X and $\mathsf{take}_{n-1}(E_K(X))$ can produce $E_K(X)$ with high probability. We call this procedure as $\mathbf{AdvSCB}^{\mathrm{EME}_K(.,.)}(X, \mathsf{take}_{n-1}(E_K(X)))$, which is shown in Fig. 5.

```
$\mathbf{AdvSCB}^{\mathrm{EME}_K(.,.)}(X, \mathsf{take}_{n-1}(E_K(X)))$

1. Apply an arbitrary encryption query of $n$ blocks with
   an arbitrary tweak.
     – Obtain the $n$ bits of $L$ using the L-side-channel .
2. $Z \leftarrow \mathsf{take}_{n-1}(E_K(X))$
3. $S \leftarrow \mathbf{AdvSCA}^{\mathrm{EME}_K(.,.)}(Z||1 \oplus x^{-1}L)$
4. Apply a query with tweak $Z||1$ and plaintext $X \oplus L$
     – Get the output as $C$
5. If $\mathsf{take}_{n-1}(C \oplus L) = S$ output $Z||1$
   else output $Z||0$
```

Fig. 5. The side channel adversary with access to EME encryption algorithm producing $E_K(X)$

Proposition 4 *Let the procedure* $\mathbf{AdvSCB}^{\mathrm{EME}_K(.,.)}(X, \mathsf{take}_{n-1}(E_K(X)))$ *be as described in Fig. 5, then*

$$\Pr[\mathbf{AdvSCB}^{\mathrm{EME}_K(.,.)}(X, \mathsf{take}_{n-1}(E_K(X)) = E_K(X)] \geq 1 - \frac{1}{2^{n-1}}$$

Proof. Let us first see what is done in the procedure described in Fig. 5. The adversary knows X and the $(n-1)$ significant bits of $E_K(X)$. He wants to predict the missing bit of $E_K(X)$. We call the $(n-1)$ bits of $E_K(X)$ as Z. He guesses that the missing bit is 1 and tries to verify if his guess is correct. First let us concentrate on the query made at step 4. The query is made with a tweak $Z||1$ and a single block of plaintext $X \oplus L$. If his guess regarding the last bit of $E_K(X)$ is correct, then $Z||1$ would be $E_K(X)$, and in such a case the response C obtained would be

$$C = L \oplus E_K(E_K(X) \oplus E_K(0^n)) = L \oplus E_K(E_K(X) \oplus x^{-1}L). \qquad (2)$$

This can be easily verified from the algorithm in Fig. 3. In the third step of the procedure, S is the output of $\mathbf{AdvSCA}^{\mathrm{EME}_K(.,.)}$ on $Z||1 \oplus x^{-1}L$. So according to Proposition 3

$$S = \mathsf{take}_{n-1}(E_K(Z||1 \oplus x^{-1}L)). \qquad (3)$$

So if the guess is correct then from eq. (2) and eq. (3) we get that

$$\mathsf{take}_{n-1}(C \oplus L) = S.$$

Thus, if the last bit of $E_K(X)$ is 1, then the procedure will always output the correct value of $E_K(X)$. On the other hand if the guess is wrong, i.e., the last bit of $E_K(X)$ is zero, then the response to the query in step 4 would be as

$$C = L \oplus E_K(E_K(X) \oplus E_K(0^{n-1}1))$$

And in this case the check in step 5 will pass with a probability less than $\frac{1}{2^{n-1}}$. Thus, the probability with which a correct guess can be made is greater than $(1 - \frac{1}{2^{n-1}})$. □

So using the procedures described in Fig 4 and Fig 5 the side channel adversary can compute $E_K(X)$ for a X of his choice with very high probability. Using the same technique the adversary can compute $E_K^{-1}(X)$ for any X given access to the decryption algorithm of EME.

Thus, we can conclude that given access to the encryption and decryption algorithms of EME and the relevant side channel information, an adversary can compute the encryption of a plain-text P of his choice without querying the encryption algorithm at P. Similarly, (s)he can decrypt any ciphertext C without querying the decryption algorithm at C.

8 EME2 Mode of Operation

EME2 adds certain functionalities which are not present in EME, for example, EME2 can handle arbitrary long messages (recall that EME cannot securely encrypt messages larger than n blocks long). Additionally EME2 can handle arbitrarily long tweaks (EME can only handle n bit tweaks, where n is the block length of the block cipher). The description of EME2 is a bit different from that of EME, the primary difference being that it encrypts the tweak. The description of EME2 is given in Fig. 6. The description given in Fig 6 is not the full description, if we assume that both the tweak length and the block length are multiples of the block length of the block cipher and the number of blocks are less less or equal to the block length of the block cipher, then the original description of EME2 translates to the description given in Fig. 6. But the above stated restrictions are not valid for the EME2 mode, the full description of the mode can handle plaintexts which do not satisfy these restrictions. For the full description of the mode see [1]. The main difference of the restricted description

```
Algorithm EME2.Encrypt$^T_{K_1,K_2,K_3}(P)$
1.  Partition $P$ into $P_1, P_2, \ldots, P_m$
2.  if len(T)= 0 then $T^* = E_{K_1}(K_3)$
3.  else
4.      Partition the tweak $T$ to $T_1, T_2, \ldots, T_r$
5.      for $i = 1$ to $r$
6.          $K_3 \leftarrow xK_3$
7.          $TT_i \leftarrow E_{K_1}(K_3 \oplus T_i) \oplus K_3$
8.      $T^* = TT_1 \oplus TT_2 \oplus \cdots \oplus TT_r$
9.  end if
10. $L \leftarrow K_2$
11. for $i = 1$ to $m$
12.     $PPP_i \leftarrow E_{K_1}(L \oplus P_i)$
13.     $L \leftarrow xL$
14. end for
15. $MP \leftarrow PPP_1 \oplus PPP_2 \oplus \cdots \oplus PPP_m \oplus T^*$
16. $MC \leftarrow E_{K_1}(MP)$
17. $M \leftarrow MP \oplus MC$
18. for $i \leftarrow 2$ to $m$
19.     $M \leftarrow xM$
20.     $CCC_i \leftarrow PPP_i \oplus M$
21. end for
22. $CCC_1 \leftarrow MC \oplus CCC_2 \oplus \cdots \oplus CCC_m \oplus T^*$
23  $L \leftarrow K_2$
24. for $i \leftarrow 1$ to $m$
25.     $C_i \leftarrow E_{K_1}(CCC_i) \oplus L$
26.     $L \leftarrow xL$
27. end for
28. return $C_1, C_2, \ldots, C_m$
```

Fig. 6. Encryption using EME2.

of EME2 compared to EME is in the handling of the tweak, as it can handle arbitrarily long tweaks and converts an arbitrary long tweak to a n bit value which is used in the mode. Also, EME2 uses three n bit keys, for processing the tweak it uses the key K_3, and the value of L which is used to mask the plain-texts and the ultimate outputs is first set to the value of K_2 and the bulk encryption is done by the key K_1.

9 A Distinguishing Attack on EME2

As evident from the algorithm in Fig. 6 there are four layers of *xtimes* operations performed in the algorithm. Thus, based on Assumption 1 and Proposition 1 one can get information about K_2, K_3 and M from the algorithm. We will call them as the K2-side-channel, K3-side-channel and M-side-channel respectively. Using these side-channel information one can mount a distinguishing attack on EME2. The adversary performs the following steps:

1. Apply an n block encryption query with no tweak.
   – Obtain K_2 using the K2-side-channel.
2. Apply an arbitrary encryption query with a n block tweak.
   – Obtain K_3 using the K3-side-channel.
3. Apply an encryption query with a one block tweak where $T = 0^n$, and a $n-1$ block message $P = P_1||P_2||\dots||P_{n-1}$, where $P_i = x^{i-1}K_2 \oplus xK_3$
   – Obtain $(n-2)$-bits of M using the M-side-channel . Call this as M_1.
4. Apply an encryption query with a one block tweak $T = 0^n$, and one block of message $P = P_1 = xK_3 \oplus K_2$
   – Obtain the corresponding ciphertext and call it C_1.
5. If the first $(n-2)$ bits of $C_1 \oplus xK_3 \oplus K_2$ are equal to M_1 output "EME2" otherwise "random".

To see why this attack works, note that according to the algorithm in Fig. 6, for query 3 we have the following:
Firstly, as there is a single block of tweak and the tweak is zero hence we get

$$T^* = E_{K_1}(xK_3) \oplus xK_3 \tag{4}$$

further, from line 12 of Fig. 6 we have, for all $i = 1, \dots, n-1$,

$$PPP_i = E_K(x^{i-1}K_2 \oplus P_i) = E_K(x^{i-1}K_2 \oplus x^{i-1}K_2 \oplus xK_3) = E_K(xK_3)$$

Now, according to line 15 of the algorithm in Fig. 6 we have

$$\begin{align} MP &= PPP_1 \oplus PPP_2 \oplus \dots \oplus PPP_{n-1} \oplus T^* \tag{5}\\ &= E_K(xK_3) \oplus T^* \tag{6}\\ &= E_K(xK_3) \oplus E_{K_1}(xK_3) \oplus xK_3 \tag{7}\\ &= xK_3 \tag{8}\end{align}$$

Equation (6) follows from eq. (5) because the PPP_is are all equal and we assume that n is even. Equation (7) follows from eq. (6) by substituting the value of T^* in eq. (4). Thus, the value of M gets computed as

$$M = MP \oplus E_{K_1}(MP) = xK_3 \oplus E_{K_1}(xK_3) \tag{9}$$

Thus the value M_1 obtained in step 3 consists of the first $(n-1)$ bits of M as in eq. (9).

In the query in step 4, the tweak is again a single block and its value is zero, thus the value of T^* would be same as in eq. (4), and $PPP_1 = E_{K_1}(xK_3)$. Thus we would have $M = xK_3 \oplus E_{K_1}(xK_3)$, which is same as the value of M obtained as side channel information from query 3 (eq. (9)). Continuing, according to the algorithm in Fig. 6 we would have the cipher text C_1 computed as $C_1 = E_{K_1}(xK_3) \oplus K_2$. So, we have

$$C_1 \oplus xK_3 \oplus K_2 = xK_3 \oplus E_{K_1}(xK_3) \qquad (10)$$

So comparing eq. (10) and eq. (9), we obtain $C_1 \oplus xK_3 \oplus K_2 = M$. The $(n-2)$ significant bits of M has been obtained from the side channel information in query 3. So if the check in step 5 is successful then with overwhelming probability the adversary can say that he is communicating with EME2.

The strong attack discussed in Section 7.2 for EME cannot be applied in the case of EME2. The strong attack for EME utilizes the fact that by obtaining the value of M one can obtain the block-cipher encryption of the tweak. As in EME the tweak can be freely chosen, hence one can get encryption of any string by suitably choosing the tweak. In EME2, the tweak is encrypted, this prevents one to apply the strong attack applicable to EME.

10 Conclusion

We presented some attacks on EME and EME2 assuming that *xtimes* leaks some information. These attacks does not contradict the claimed security of the modes, as the security definition and the security proofs for these modes does not assume any side-channel information being available to the adversary. Also the consequences of these attacks shown are not immediate. But, it points out that using *xtimes* indiscriminately may give rise to security weakness in true implementations.

Acknowledgements: The second author acknowledges the support from the CONACyT project 90775.

References

1. Draft standard architecture for wide-block encryption for shared storage media. https://siswg.net/index2.php?option=com_docman&task=doc_view&gid=84 &Itemid=41.
2. Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, and Pankaj Rohatgi. The EM side-channel(s). In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 29–45. Springer, 2002.
3. Ross J. Anderson and Markus G. Kuhn. Low cost attacks on tamper resistant devices. In Bruce Christianson, Bruno Crispo, T. Mark A. Lomas, and Michael Roe, editors, *Security Protocols Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 125–136. Springer, 1997.

4. Sanjay Burman, Debdeep Mukhopadhyay, and Kamakoti Veezhinathan. LFSR based stream ciphers are vulnerable to power attacks. In K. Srinathan, C. Pandu Rangan, and Moti Yung, editors, *INDOCRYPT*, volume 4859 of *Lecture Notes in Computer Science*, pages 384–392. Springer, 2007.
5. Debrup Chakraborty and Palash Sarkar. A new mode of encryption providing a tweakable strong pseudo-random permutation. In M. J. B. Robshaw, editor, *FSE*, volume 4047 of *Lecture Notes in Computer Science*, pages 293–309. Springer, 2006.
6. Debrup Chakraborty and Palash Sarkar. HCH: A new tweakable enciphering scheme using the hash-counter-hash approach. *IEEE Transactions on Information Theory*, 54(4):1683–1699, 2008.
7. Shai Halevi. EME*: Extending eme to handle arbitrary-length messages with associated data. In Anne Canteaut and Kapalee Viswanathan, editors, *INDOCRYPT*, volume 3348 of *Lecture Notes in Computer Science*, pages 315–327. Springer, 2004.
8. Shai Halevi. Invertible universal hashing and the TET encryption mode. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 412–429. Springer, 2007.
9. Shai Halevi and Phillip Rogaway. A tweakable enciphering mode. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 482–499. Springer, 2003.
10. Shai Halevi and Phillip Rogaway. A parallelizable enciphering mode. In Tatsuaki Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 292–304. Springer, 2004.
11. Antoine Joux. Cryptanalysis of the EMD mode of operation. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2003.
12. Antoine Joux and Pascal Delaunay. Galois LFSR, embedded devices and side channel weaknesses. In Rana Barua and Tanja Lange, editors, *INDOCRYPT*, volume 4329 of *Lecture Notes in Computer Science*, pages 436–451. Springer, 2006.
13. Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In Neal Koblitz, editor, *CRYPTO*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.
14. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.
15. David A. McGrew and Scott R. Fluhrer. The security of the extended codebook (XCB) mode of operation. In Carlisle M. Adams, Ali Miri, and Michael J. Wiener, editors, *Selected Areas in Cryptography*, volume 4876 of *Lecture Notes in Computer Science*, pages 311–327. Springer, 2007.
16. Raphael Chung-Wei Phan and Bok-Min Goi. On the security bounds of CMC, EME, EME$^+$ and EME* modes of operation. In Sihan Qing, Wenbo Mao, Javier Lopez, and Guilin Wang, editors, *ICICS*, volume 3783 of *Lecture Notes in Computer Science*, pages 136–146. Springer, 2005.
17. Phillip Rogaway. The EMD mode of operation (a tweaked, wide-blocksize, strong PRP). Cryptology ePrint Archive, Report 2002/148, 2002. http://eprint.iacr.org/.
18. Palash Sarkar. Improving upon the TET mode of operation. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *ICISC*, volume 4817 of *Lecture Notes in Computer Science*, pages 180–192. Springer, 2007.
19. Peng Wang, Dengguo Feng, and Wenling Wu. HCTR: A variable-input-length enciphering mode. In Dengguo Feng, Dongdai Lin, and Moti Yung, editors, *CISC*, volume 3822 of *Lecture Notes in Computer Science*, pages 175–188. Springer, 2005.