# Customizing Cellular Message Encryption Algorithm

Debdeep Mukhopadhyay and Dipanwita RoyChowdhury
*(Corresponding author: Debdeep Mukhopadhyay)*

Department of Computer Science and Engineering, IIT Kharagpur
Kharagpur, West Bengal, 721302 India (Email: debdeep.mukhopadhyay@gmail.com)

*(Received Jan. 13, 2006; revised and accepted Apr. 12, 2006)*

## Abstract

This paper observes the cryptanalysis of the Telecommunications Industry Association's Cellular Message Encryption Algorithm (CMEA). The CMEA has been widely used for wireless security and the breaking of the scheme proves the requirement of alternatives. In the current paper, the properties of CMEA which have lead to the successful cryptanalysis, have been identified. Accordingly the algorithm has been modified to prevent the attacks. Finally the customized CMEA has been subjected to standard linear and differential cryptanalysis to evaluate its security margin. The endeavour demonstrates that with appropriate modifications the CMEA can be transformed into a strong cipher, which is essential for wireless security.

*Keywords: CMEA, cryptanalysis, wireless security*

## 1 Introduction

Cellular Message Encryption Algorithm (CMEA) [12] has been developed by the Telecommunications Industry Association (TIA) to encrypt digital cellular phone data. CMEA is one of the four cryptographic primitives specified for telecommunications and is designed to encrypt the control channel, rather than the voice data. It is a block cipher which uses a 64 bit key and operates on a variable block length. CMEA is used to encrypt the control channels of cellular phones. It is distinct from ORYX, an also insecure stream cipher that is used to encrypt data transmitted over digital cellular phones.

In March 1997, Counterpane Systems and UC Berkeley jointly [13] published attacks on the cipher showing it had several weaknesses. In the paper the authors have presented several attacks on CMEA which are of practical threat to the security of digital cellular systems. The authors describe an attack on CMEA which requires $40-80$ known plaintexts, has time complexity about $2^{24}-2^{32}$, and finishes in minutes or hours of computation on a standard workstation. The authors point out that the cryptanalysis of CMEA underscores the need for an open cryptographic review process. Thus having faith on new algorithms which are designed close door is always dangerous. The use of such algorithms can lead to a total collapse of the cellular telephonic industry. CMEA is used to protect sensitive control data, such as the digits dialed by the cellphone user. A successful break of CMEA might reveal user calling patterns. Finally compromise of the control channel contents could lead to the leaking of any confidential data (like credit card numbers, bank account numbers and voice mail PIN numbers) that the user types on the keypad.

Following the revelation of the weakness of CMEA, a patchup algorithm called ECMEA was standardised by TIA. A further enhancement of ECMEA, called SCEMA, is also developed [12]. However according to [13] the previous cryptanalysis of all the crypto-algorithms proposed by TIA clearly demonstrate that there is a need of explicitly stating security assumptions during every step of the design. Also security components should not be reused without thoroughly examining the implications of reuse. Although it has been proposed that the future generation cellular networks (CDMA 2000 1X Revision A) will use AES (Rijndael) [2], the implementation constraints of a wireless network might prove to be a concern. This motivates the design of special ciphers for wireless telephones (networks) but at the same time which are evaluated meticulously. The security margins of such algorithms must be stated so as to increase confidence in the ciphers. In other words, dedicated as well as standard block cipher security analysis should be presented for the ciphers which are used to prevent frauds in such important networks. In these lines, the present paper revisits the CMEA algorithm. The algorithm has been analysed to understand the reasons of its insecurity. Based upon the analysis the CMEA has been modified to CMEA-I. The new algorithm has been analysed and it has been shown that the original attacks does not work against the cipher. Also the diffusion and confusion properties of CMEA-I has been demonstrated by means of Avalanche analysis. The security of CMEA depends on the strength of the

T-box. Hence, security margins have been presented to establish that the T-box provides sufficient security margins against linear and differential cryptanalysis.

The paper is organised as follows. In Section 2 the preliminaries have been stated which details the original CMEA algorithm and the attacks against it. In Section 3 the CMEA algorithm has been analysed to understand why the algorithm breaks in the face of the attacks detailed in Section 2. Section 4 presents the customized CMEA with necessary modifications to plague the existing weaknesses of CMEA. Section 5 performs a security analysis of CMEA-I. The section shows how CMEA-I prevents the attack proposed in [13]. The diffusion and confusion properties of CMEA-I has also been analysed in the section using Avalanche criterion. Linear and differential cryptanalysis has been performed on the T-box in the section. The efficiency of the cipher has been discussed in Section 6. Finally Section 7 concludes the work.

# 2 Preliminaries

This section describes the CMEA algorithm and the existing cryptanalysis of CMEA. CMEA is a byte-oriented variable width block cipher with a 64 bit key. Block sizes may be any number of bytes. CMEA is optimized for 8-bit microprocessors with severe resource limitations.

## 2.1 The CMEA as It Is

CMEA has three layers. The first layer performs one non-linear pass on the block, affecting left-to-right diffusion. The second layer is a purely linear, unkeyed operation intended to make changes in the opposite direction. One can think of the second step as XORing the right half of the block onto the left half. The third layer performs a final non-linear pass on the block from left to right. In fact, it is the inverse of the first layer.

CMEA obtains its non-linearity in the first and third layer from an 8-bit keyed lookup table known as the T-box. The T-box calculates its 8-bit output as $T(x) = C(((C(((C(((C((x \oplus K_0) + K_1) + x) \oplus K_2) + K_3) + x) \oplus K_4) + K_5) + x) \oplus K_6) + K_7) + x$, $x$ is the input byte and $K_{0,\cdots,7}$ represents the 8 byte key. In this equation C is an unkeyed 8-bit lookup table known as the CaveTable. The operation $\oplus$ represents a bit-wise xor, while + denotes binary addition on the operands. All the operations are 8 bit operations. The algorithm encrypts an $n$-byte message $P_{0,\cdots,n-1}$ to a ciphertext $C_{0,\cdots,n-1}$ under the key $K_{0\cdots7}$ as follows:

**Algorithm 1.**

$y_0 = 0$
$for(i = 0; i < n; i++)$
{
  $P'_i = P_i + T(y_i \oplus i)$
  $y_{i+1} = y_i + P'_i$
}
$for(i = 0; i < \lfloor n/2 \rfloor; i++)$
  $P_i^{'} = P'_i \oplus (P'_{n-i-1} \bigvee 1)$
$z_0 = 0$
$for(i = 0; i < n; i++)$
{
  $z_{i+1} = z_i + P_i^{'}$
  $C_i = P_i^{'} - T(z_i \oplus i)$
}

Recovering the values of all the 256 T-box entries is equivalent to the breaking of CMEA even if the keys are not recovered. The values of $T(0)$ occupies a position of special importance. $T(0)$ is always used to obtain $C_0$ from $P_0$. Without $T(0)$ one cannot trivially predict where other T-box entries are likely to be used. Knowing $T(0)$ lets us learn the inputs to the T-box lookups that modify the second byte in the message. The CAVE Table has very skewed statistical distribution. 92 of the possible 256 eight bit values never appear.

## 2.2 Attacks on CMEA

The attacks against CMEA are briefed next. The attacks [13] can be categorised into two broad types:

### 2.2.1 A Chosen Plaintext Attack

CMEA is weak against chosen-plaintext attacks; one can recover all the T-box entries with about 338 chosen texts (on average) and very little work. The attacker does not have control over the block length. The attack has two steps.

1) Recovery of $T(0)$
   For each guess of $x$, where $x$ is a byte, the message $P = (1-x, 1-x, 1-x, \cdots, 1-x)$ is encrypted, where the sign $-$ denotes binary subtraction. Each byte has the value $(1-x)$. If the result is of the form $C = (-x, \cdots)$ then with very high probability $T(0) = x$. There are only 256-92=164 possible values of $T(0)$, thus the correct value is expected to be guessed using on the average $164/2 = 82$ trials.

2) Recovery of the remaining T-box entries
   For each byte $j$, to learn the value of $T(j)$ let $k = ((n-1) \oplus j) - (n-2)$, where the desired blocks are $n$ bytes long. The encryption of $P = (1 - T(0), 1 - T(0), \cdots, 1 - T(0), k - T(0), 0)$ is obtained. If the result is of the form $C = (t - T(0), \cdots)$ then with high probability $T(j) = t$, with a possible ambiguity in the LSB. The second phase requires 256 more chosen plaintexts, thus requiring 338 chosen plaintexts on the whole.

### 2.2.2 A Known Plaintext Attack on 3-byte Blocks

Because of the skewed distribution of the CAVE Table $T(0)$ can have 164 possibilities. For each guess at $T(0)$, a $256 \times 256$ array of $p_{i,j}$ is constructed which checks whether

$T(i) = j$ is possible for each $i, j$. All values for $T(i)$, $i > 0$, are initially listed as possible. Since, $T(i) - i$ is a CAVE Table output and the Cave Table has an uniform distribution, one can immediately rule out the 92 values for $T(i)$.

Using each known plaintext/ciphertext pair lets us establish implications of the form, $T(0) = t_0, T(i) = j \Rightarrow T(i') = j'$.

If we have eliminated $T(i') = j'$ as impossible, then we can conclude $T(i) = j$ is impossible. In this way $p_{i,j}$ is reduced. One either reaches a conclusion or moves to Phase 2.

The second phase recovers the CMEA key from the information previously stored in the $p_{i,j}$ array. The key recovery is based on pruned search. First one guesses $K_6$ and $K_7$. Then the effect of the last one-fourth of the T-box is peeled off and checked whether it is a valid T-box entry. Because of the skew in the CAVE Table incorrect key guesses are easily identified. The pruned key search is continued by guessing $K_4$ and $K_5$. Though the pruned search complexity grows very fast, the T-box can be subjected to a classic meet-in-the-middle attack. One can work halfway through the T-box given only $K_{0\cdots3}$, and one can work backwards up to the middle given just $K_{4\cdots7}$ and look for a match. The combination of the pruned search and the meet-in-the-middle attack cryptanalysis recovers the entire CMEA Key with 40-80 known plaintexts.

# 3 Why is CMEA Weak?

A detailed study of the CMEA algorithm shows why CMEA is susceptible to chosen plaintext and known plaintext attacks. In this section the properties of the algorithm which make the cipher weak have been identified. The CMEA algorithm is modified to a new algorithm named CMEA-I plaguing the weaknesses of the existing CMEA. The security of CMEA-I has been analyzed in the following section. Recovery of all values of the 256 T-box entries is equivalent to the breaking of the cipher, so the strength of the T-box requires special attention and hence has been treated subsequently in details.

- **Property 1:** If the plaintext is of the form $P = \{1 - x, 1 - x, \cdots, 1 - x\}$ and the ciphertext is of the form $C = \{-x, \cdots\}$ then with very high probability $T(0) = x$.

  **Analysis:** $P'_0 = P_0 + T(0) = 1 - x + T(0)$. If $T(0) = x$, we have $P'_0 = 1$. Thus, $y_1 = y_0 + P'_0 = 0 + 1 = 1$. Likewise, $P'_1 = P_1 + T(1 \oplus 1) = 1 - x + T(0)$. If $T(0) = x$, we have $P'_1 = 1$. Thus, $y_2 = y_1 + P'_1 = 1 + 1 = 2$. Thus continuing we have $P'_{n-1} = 1$.

  So, $P_0^{'} = P'_0 \oplus (P'_{n-1} \bigvee 1) = 1 \oplus 1 = 0$. Hence, $C_0 = P_0^{'} - T(0) = -T(0) = -x$.

  The probability when using the CaveTable is dependent on the fact that the initial guess for $T(0)$ is correct and the possible number of trials is thus only (256-92)/2 = 82 on the average.

- **Property 2:** If the plaintext is of the form $P = \{1 - T(0), 1 - T(0), \cdots, 1 - T(0), k - T(0), 0\}$ and the ciphertext is $C = \{t - T(0), \cdots\}$ where $k = ((n-1) \oplus j) - (n-2)$ then with very high probability $t = T(j)$.

  **Analysis:** It has been shown that $P'_i = 1$ and $y_{i+1} = (i+1)$, where $0 \leq i \leq (n-3)$. Now,

  $$\begin{aligned} P'_{n-2} &= P_{n-2} + T(y_{n-2} \oplus (n-2)) \\ &= P_{n-2} + T(0), \quad \text{since } y_{n-2} = n-2 \\ &= k - T(0) + T(0) = k. \end{aligned}$$

  Using this fact, $y_{n-1} = y_{n-2} + P'_{n-2} = (n-2) + k = (n-1) \oplus j$. Therefore, $P'_{n-1} = P_{n-1} + T(y_{n-1} \oplus (n-1)) = 0 + T(j)$. Thus, $C_0 = P_0^{'} - T(0)$ or $t - T(0) = P'_0 \oplus (P'_{n-1} \bigvee 1) - T(0)$ or $t = 1 \oplus (T(j) \bigvee 1) = T(j)$, with a very high probability, with some confusion with the LSB.

- **Property 3:** The CMEA algorithm uses a skewed CAVE Table [13]. The CAVE Table is not a permutation and 92 of the possible 256 values does not occur.

- **Property 4:** The CMEA algorithm uses a four round T-box which can be subjected to meet-in-the-middle attack [13].

Using the above properties one can explain why the CMEA algorithm is weak against the chosen plaintext and known plaintext attacks. The causes of the attacks are enlisted as follows:

1) Chosen Plaintext Attack: The CMEA algorithm is weak against chosen plaintext attack because of Properties 1 and 2.

2) Known Plaintext Attack: The known plaintext attack is powerful against the CMEA algorithm because of Properties 3 and 4.

# 4 Customized Cellular Message Encryption Algorithm : CMEA-I

Analyzing the above properties the CMEA algorithm has been modified. The resultant cipher is presented in this section.

- **Modification 1:** Clearly the update equation of $P_i$ needs to be changed so that Properties 1 and 2 work no more. The modified equation is of the form:

  $$P'_i = P_i + T(y_i \oplus f(i, n)),$$

such that as we vary $i$ from 0 to $(n-1)$ (where $n$ is the number of byte blocks in the plaintext) the T-box is not predictably accessed. In the original CMEA Property 1 exists because for a particular nature of the input plaintext and key the T-box was always referred at the point 0. So, the function $f(i,n)$ should be such that the T-box is accessed at different points. After considering several forms of the function $f(i,n)$ the proposed function is $f(i,n) = (2i)\%n$, where % represents the modulo operation. Hence the update equation is:

$$P'_i = P_i + T(y_i \oplus ((2i)\%n)).$$

Thus the algorithm is transformed into:

**Algorithm 2.**
$y_0 = 0$
$for(i = 0; i < n; i++)$
{
$P'_i = P_i + T(y_i \oplus ((2i)\%n))$
$y_{i+1} = y_i + P'_i$
}
$for(i = 0; i < \lfloor n/2 \rfloor; i++)$
$P_i^{'} = P'_i \oplus (P'_{n-i-1} \bigvee 1)$
$z_0 = 0$
$for(i = 0; i < n; i++)$
{
$z_{i+1} = z_i + P_i^{'}$
$C_i = P_i^{'} - T(z_i \oplus ((2i)\%n))$
}

- **Modification 2:** The CAVE Table is replaced with the AES S-box which can be efficiently implemented [9]. Thus the distribution is no more skewed and all the possible 256 values appear as a possibility.

- **Modification 3:** The T-box previously had 4 rounds. The number of rounds of the T-box has been increased to 8 rounds to prevent meet-in-the-middle attack. The output of the 4 round T-box is recycled again through the T-box.

# 5 Security Analysis of CMEA-I

In the present section the security of CMEA-I has been analyzed. The analysis shows that the scheme does not break under a chosen plaintext and known plaintext attack. Avalanche analysis has been performed on CMEA-I. The results show that the scheme provides the necessary diffusion and confusion necessary for a strong cryptographic scheme. The security of the T-box plays a vital role in the security of the cipher. So, the T-box has been also analysed using linear and differential cryptanalysis.

## 5.1 How CMEA-I Prevents Chosen-Plaintext and Known-Plaintext Attacks?

Due to the modifications incorporated in the cipher the original attack does not work for CMEA-I. For 50,000 variations of the key, plaintexts of the form $(1 - T(0), 1 - T(0), \cdots, 1 - T(0))$ gives ciphertext of the form $(-T(0), \cdots)$ only 0.766% of the time. However we present a modified attack in lines with the original attack and show that the cipher prevents the attack successfully.

Let the $P_0$ block of the plaintext be $(1 - x_0)$. Thus $P'_0 = P_0 + T(y_0 \oplus 0) = 1 - x_0 + T(0 \oplus 0) = 1 - x_0 + T(0)$. Let $x_0 = T(0)$. So $P'_0 = 1$ and $y_1 = y_0 + P'_0 = 1$. Similarly, $P'_1 = P_1 + T(1 \oplus 2) = P_1 + T(3)$. Hence if we have $P_1 = 1 - x_1$ and let $x_1 = T(3)$. So, $P'_1 = 1$ and $y_2 = y_1 + P'_1 = 1 + 1 = 2$. Likewise,

$$\begin{aligned} P'_2 &= P_2 + T(y_2 \oplus 4) \\ &= P_2 + T(2 \oplus 4) \\ &= 1 - x_2 + T(6), \quad \text{if } P_2 = 1 - x_2 \\ &= 1, \quad \text{using the guess } x_2 = T(6). \\ y_3 &= y_2 + P'_2 = 2 + 1 = 3. \end{aligned}$$

For the fourth block,

$$\begin{aligned} P'_3 &= P_3 + T(y_3 \oplus 6) \\ &= P_3 + T(3 \oplus 6) \\ &= 1 - x_3 + T(5), \quad \text{if } P_3 = 1 - x_3 \\ &= 1, \quad \text{if } x_3 = T(5). \end{aligned}$$

Thus if we have four blocks in the plaintext (without loss of generality) then $P_0^{'} = P'_0 \oplus (P'_3 \bigvee 1) = 0$. and hence, $C_0 = 0 - T(0) = -T(0)$.

Thus for 4 input blocks if one obtains chosen plaintexts of the form $P = (1 - T(0), 1 - T(3), 1 - T(6), 1 - T(5))$ then the ciphertext is of the form $C = (-T(0), \cdots)$. Then the number of trials on the average is $(256^4)/2$ which is equivalent to a brute force search on the entire plaintext space and is much larger than that required for original CMEA. (Note that as the CAVE Table has been replaced by the S-box of Rijndael-AES the number of possible values of each T-box access is 256).

The following proof shows that the attack is inefficient against CMEA-I.

*Proof.* During the attack we find that at each stage $y_i = i$ and $f(i,n) = (2i)\%n$, where % refers to the modulo operation. Let CMEA-I break in the face of the attack. For the attack to work the T-box must be accessed at the same point for at least a single case. In other words there should be repetition in the point at which the T-box is accessed.

Let us have two instances of $i$, namely $i_1$ and $i_2$ $(i_1 \neq i_2)$, for which the T-box is accessed at the same point. Thus, $i_1 \oplus ((2i_1)\%n) = i_2 \oplus ((2i_2)\%n)$ or $(i_1 \oplus i_2) = 2(i_1 \oplus i_2)\%n$. If, $2(i_1 \oplus i_2) < n$, then the equation is possible if $i_1 = i_2$, contradicting our initial assumption.

Also, if $2(i_1 \oplus i_2) = kn+r > n$ (where $k \geq 1$ and $r < n$), we have $(kn+r)/2 = (kn+r)\%n = r$ or $kn = r$, which is not possible as $r < n$. Thus we arrive at a contradiction, and hence the T-box is not accessed at the same point. Thus the attack does not work against CMEA-I.

Also the number of chosen plaintexts grows exponentially with the number of blocks. For an n byte block the number of chosen plaintexts is of the order of $256^n$. Thus the number of plaintexts to be investigated is equal to that in a brute force search on the entire plaintext space. Such a large number of plaintext requirement makes the attack ineffective against CMEA-I. □

As the CAVE Table has been replaced by the AES S-box the skewness of the CAVE Table does not exist. Also all the 256 values may appear. The T-box has been extended to eight rounds and thus a meet-in-the-middle attack does not work. The known plaintext attack against the original CMEA was found to be ineffective against the customized CMEA (CMEA-I).

## 5.2 Diffusion and Confusion in the CMEA-I Algorithm

Diffusion and confusion are two important properties necessary for the security of block ciphers [10]. The current section of the paper deals with diffusion and confusion in the CMEA-I algorithm. The CMEA-I algorithm has been subjected to Avalanche Attack to test the confusion and diffusion which the cipher provides. A function has a good avalanche effect when a change in one bit of the input results in a change of half of the outputs bits.

Diffusion criteria requires that a change in a single bit of the plain text should cause a change in several bits in the cipher text (the key is kept constant). In order to test the diffusion property the CMEA-I algorithm has been subjected on pairs of plaintext which differ by one bit. The number of output bits affected should have a mean of $n/2$ where $n$ is the number of bits of the cipher. In other words it is expected that for a good cipher approximately half of the output bits should be affected. The experiments have been performed on a block size of three-bytes (24 bits). In Figure 1 the frequency of the number of bits affected has been plotted versus the number of bits affected. The plot shows that around 12 bits are affected for a maximum number of cases. Also the computed average is around 11.98. The plot shows that the algorithm provides sufficient diffusion property.

Confusion criteria requires that a change in a single bit in the key should cause a change in several bits in the cipher text (the plaintext is kept constant). In order to test the confusion property the CMEA-I algorithm has been used to encrypt plaintexts with pairs of keys which differ by one bit. The number of output bits affected according to the Avalanche criterion should be around $n/2$ where $n$ is the number of bits of the cipher. The experiments have been performed again on a block size of three-bytes (24 bits). In Figure 2 the frequency of the number of bits affected has been plotted versus the number of bits affected. The plot shows that around 12 bits are affected for a maximum number of cases. Also the computed average is around 11.91. The plots show that the confusion property is satisfied by CMEA-I.

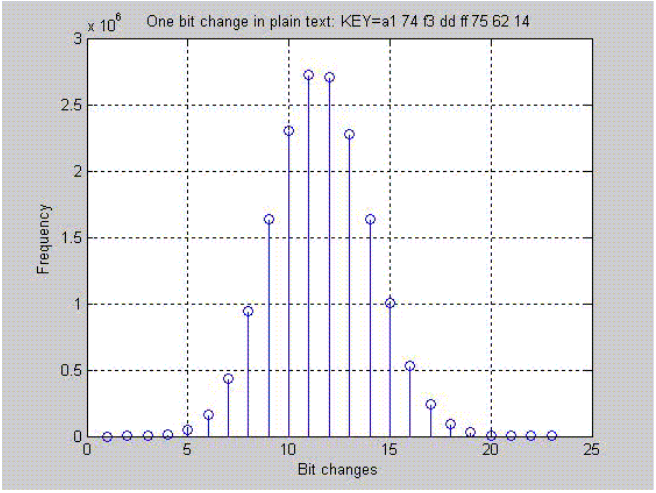



Figure 1: Avalanche effect to show diffusion

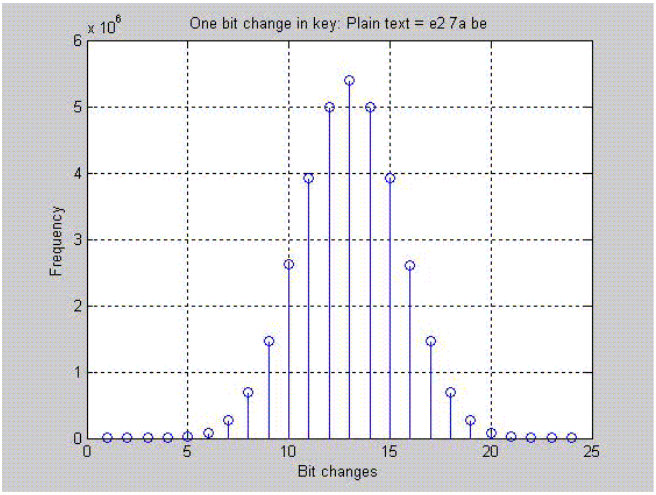



Figure 2: Avalanche effect to show confusion

## 5.3 Finer Issues of Security

The T-box plays a central role in the cipher structure of CMEA. One can gather information about the T-box entries from the known CMEA encryptions. Also if the T-box is compromised and all the T-box outputs can be

identified then the CMEA algorithm is also broken. So, the problem reduces to the cryptanalysis of the T-box algorithm, given information about the input and output of some of the elements. More formally in this section we shall inspect given the T-box input and outputs for some values is it possible to obtain the other T-box elements or to recover the key. We analyze the T-box algorithm under linear and differential attacks [1, 4, 5, 11].

### 5.3.1 Differential Analysis of the T-box

Differential Analysis on a block cipher observes that given a certain input difference if a particular output difference occurs with a high probability. In an ideal cipher for an $n$-bit block the probability should be of the order of $1/2^n$. Differential cryptanalysis seeks to exploit a scenario where a particular output difference $\delta Y$ occurs given a particular input difference $\delta X$ with a very high probability. The pair $(\delta X, \delta Y)$ is referred to as a differential.

We first observe the security which a single round of the T-box provides against a differential attack. The Figure 3 shows the single round T-box.

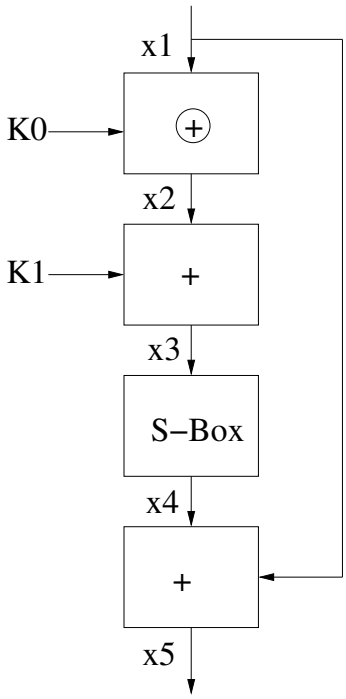



Figure 3: One round T-box

Given $x_1$ and $x_5$ (the input and output pair at any point) one can calculate $x_3$. Thus the problem reduces to the cryptanalysis of the portion in the T-box shown in Figure 4.

In Figure 4, $\delta x_1$ and $\delta x_2$ are same and does not depend upon the key. Once we know $\delta x_2$ and $\delta x_3$ and observe the differential property of the addition block to obtain information about $K_1$ we can also infer information about $K_0$.

From the differentials of the addition block we observe the following two facts:

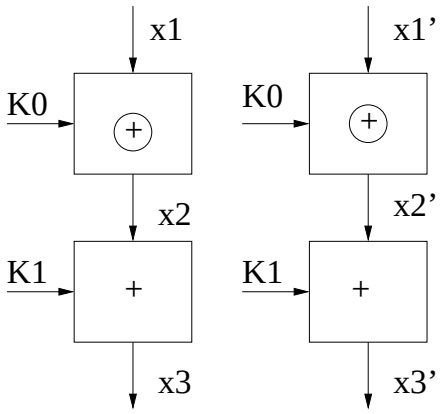



Figure 4: Differential analysis of T-box

1) For a fixed $(\delta x_2, \delta x_3)$ can certain keys be ruled out?

2) What is the worst case size of the reduced key space?

Analysis of one round of the T-box brings the following observations to the surface. We have created tables for the entire key space and noted how many keys are possible for each pair of $(\delta x_2, \delta x_3)$. The tables show that the distribution is very sparse and there are large number of cases where a $(\delta x_2, \delta x_3)$ pair is not possible for any key. There are instances for which certain keys can be immediately ruled out. The remaining set of possible keys varies in size and ranges from as low as 2 to 254 (except the $(0, 0)$ pair where all the keys are possible). Thus in such worst case scenario a random search over only 2 values will reveal $K_1$ and hence $K_0$. Hence, one round of the T-box shows weaknesses. So, we require to increase the number of rounds of T-box.

Let us calculate the maximum probability of a differential to pass through one round of the T-box. It was found that there exists weak keys for each possible $\delta x_1$. The weak key is defined to be a key for which there is a $\delta x_3$ which always occurs for the particular $\delta x_1$ and the key. Next the $\delta x_3$ which serves as an input to the S-box was considered. The corresponding output differential $\delta x_4$ with the highest probability was observed. Next for all these $\delta x_1$'s and $\delta x_4$'s the possible $\delta x_5$'s were found which had the highest probability.

The above steps were done for all the possible $\delta x_1$'s and their corresponding weak keys. The analysis results in the worst case maximum probability of obtaining a $\delta x_5$ for any given $\delta x_1$.

The probability worked to around 0.0078, so for 8 rounds of the T-box the probability is around $1.37 \times 10^{-17}$, which is negligible. If we do not use the weak keys then the worst case probability of the passing of differential reduces to around 0.0039. But this reduces the key space.

### 5.3.2 Linear Cryptanalysis of the T-box

The S-box of AES is known to be resistant against linear cryptanalysis. The current subsection works out the

security margin which the T-box provides against Linear Cryptanalysis (LC) and shows that the scheme is at least as secured as the AES S-box.

Linear Cryptanalysis tries to take advantage of high probability occurrence s of linear expressions involving plaintext bits, the ciphertext bits and subkey bits. The difference of the probability from the probability 1/2 is known as the bias of the linear equation. Linear Cryptanalysis exploits linear approximations with a large bias. It is a known plaintext attack, that is the attacker does not choose which plaintexts are available. The basic idea is to approximate the operation of a portion of the cipher with a linear expression where the linearity refers to a mod-2 bitwise operation ($\oplus$).

We obtain a linear expression relating the bits of $x_1$ and the keys $K_0$ and $K_1$, refer Figure 3. The expressions may be derived as follows:

$$\begin{aligned} x_5[0] &= x_4[0] \oplus x_1[0] \quad \text{with probability 1} \\ &= f(x_3[i_1], x_3[i_2], \cdots, x_3[i_k]) \oplus x_1[0], \end{aligned}$$

where $f$ is a linear approximation for the S-box with the largest bias $\epsilon_{RD}$.

It may be noted that such an expression will have the largest linear probability bias, which is the amount by which the linear probability differs from 1/2. All other linear expressions will have a smaller bias and hence we have considered this expression. The fact has been verified experimentally, however the fact can be established with the following logic.

The bias of other linear expressions can be estimated from the bias of the individual linear expressions by using the Piling-Up lemma [11, 4]. Thus if we combine $l$ linear equations of the form $x_5[i] = x_4[i] \oplus x_1[i]$ each with a bias $(1/2)^{i+1}$, (we prove this later in lemma1) the bias of the combined equation is

$$\begin{aligned} & 2^{l-1} \prod_{i=i_1}^{l} (1/2)^{i+1} \\ = \; & 2^{l-1}[(1/2)^{i_1+1}.(1/2)^{i_2+1} \cdots (1/2)^{i_l+1}] \\ = \; & 1/2^{(i_1+i_2+\cdots+i_l+1)} < 1/2. \end{aligned}$$

Note that 1/2 is the bias of the equation with which we have started with viz. $x_5[0] = x_4[0] \oplus x_1[0]$. So, all other linear approximations have a lesser bias. Hence, all the other linear equations which are developed from other starting equations have a lesser bias. Thus we compute the probability of the linear trail with the maximum bias, to give us the upper bound of linear probability bias for the T-box.

Now to obtain linear expressions where $x_3[i_1]$ is expressed in terms of $x_1[i_1]$ and $K_0[i_1]$ we have

$$\begin{aligned} x_3[i_1] &= x_2[i_1] \oplus K_1[i_1], \quad \text{with bias } \epsilon_{i_1}, \\ &= x_1[i_1] \oplus K_0[i_1] \oplus K_1[i_1], \quad \text{with bias } \epsilon_{i_1}. \end{aligned}$$

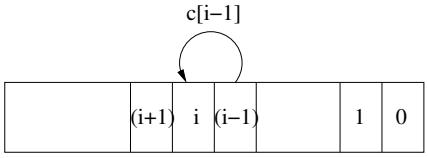
Similarly,

$$\begin{aligned} x_3[i_2] &= x_1[i_2] \oplus K_0[i_2] \oplus K_1[i_2], \quad \text{with bias } \epsilon_{i_2} \\ x_3[i_3] &= x_1[i_3] \oplus K_0[i_3] \oplus K_1[i_3], \quad \text{with bias } \epsilon_{i_3}. \\ &\vdots \\ x_3[i_k] &= x_1[i_k] \oplus K_0[i_k] \oplus K_1[i_k], \quad \text{with bias } \epsilon_{i_k}. \end{aligned}$$

The linear approximation for the S-box with maximum bias is: $x_5[0] = f(x_3[i_1], x_3[i_2], \cdots, x_3[i_k]) \oplus x_1[0]$, with bias $\epsilon_{RD}$. Thus, we combine the linear equations to obtain the linear trail, $x_5[0] = f(x_1[i_1], x_1[i_2], \cdots, x_1[i_k], K_0[i_1], K_0[i_2], \cdots, K_0[i_k], K_1[i_1], K_1[i_2], \cdots, K_1[i_k]) \oplus x_1[0]$. The bias of the linear trail, using the Piling-Up lemma, is $2^k(\epsilon_{i_1}\epsilon_{i_2}\cdots\epsilon_{i_k}\epsilon_{RD})$. Using the following result we compute the upper bound of the bias.

**Lemma 1.** *For a given n-bit input x and k the output is denoted by another n-bit number y=x+k. The probability that each output bit y[i] can be denoted by the linear function $x[i] \oplus k[i]$ is denoted by $p_i$, $0 \leq i < n$. Then $p_i = 1/2 + (1/2)^{i+1}$ and $1/2 < p_i \leq 1$.*

*Proof.* Let $c[i]$ denote the carry out from the addition of $x$ and $k$ after $i$ bits, refer Figure 5. Clearly, $y[0] = x[0] \oplus k[0]$, with probability 1. Thus $p_0 = 1$.




1. The Output Register y which stores the sum of two registers x and k
2. 0, 1, ... ,(i−1), i, (i+1), ... indicates the bit positions of y
3. c[i−1] indicates the carry out after the addition of (i−1) bits are complete

Figure 5: The output state of the sum

Now, $y[1] = x[1] \oplus k[1]$ when there is no carry in $c[0]$ which is the generated carry from the addition of the lowest bits. $c[0] = 0$, with probability 3/4 and hence $p_1 = 3/4$.

Let, the event that the $i_{th}$ bit of $y$ can be expressed as a linear expression $x[i]$ and $k[i]$ has a probability $p_i$. Similarly the $(i+1)^{th}$ can be linearly expressed with a probability $p_{i+1}$.

Now, we note the following fact. The $(i+1)^{th}$ bit cannot be linearly expressed if there is a carry from the $i^{th}$ bit, that is if $c[i]$=1.

This can be divided into two mutually exclusive cases. First the event say $A$, $c[i-1]$=0 and the addition of $x[i]$ and $y[i]$ generates a carry. Now, when $c[i-1] = 0$, then $y[i]$ must have been linearly expressed (using the above fact) and the probability by definition is $p_i$. Thus the probability that $A$ is true is $1/4p_i$.

The other event $B$ is the case where $c[i-1]$=1 and the addition of $x[i]$ and $y[i]$ propagates the carry. The probability that $B$ is true is $3/4(1-p_i)$.

Clearly if the event $(A \bigcup B)$ occurs then the $(i+1)^{th}$ bit cannot be linearly expressed and the probability is by definition $(1-p_{i+1})$.

Thus, $(1-p_{i+1}) = P(A \bigcup B) = P(A)+P(B)$ (because $A$ and $B$ are mutually exclusive) $= 1/4p_i + 3/4(1-p_i)$ or $p_{i+1}=1/4+p_i/2$.

Using the recurrence relation we have $p_{i+1}=1/4+p_i/2$ $= 1/4+1/2(1/4+p_{i-1}/2) = 1/4[1+1/2]+(1/2)^2 p_{i-1}$. Next, we have $p_{i+1}=1/4[1+(1/2)+(1/2)^2+\cdots+(1/2)^i]+(1/2)^{i+1}p_0 = 1/2[1+(1/2)^{i+1}]$, since $p_0=1$. Thus, $p_i = 1/2[1+(1/2)^i] = 1/2+(1/2)^{i+1}$.

Using the equation we have $p_0=1, p_1=3/4, p_2=5/8, p_3=9/16$ and so on. Clearly, $1/2 < p_i \leq 1$. □

The linear trail has a bias of: $2^k((1/2)^{i_1+1}(1/2)^{i_2+1} \cdots (1/2)^{i_k+1}\epsilon_{RD}) = 2^k((1/2)^{i_1+i_2+\cdots+i_k+k}\ \epsilon_{RD}) = \epsilon_{RD}/(2^{i_1+i_2+\cdots+i_k}) < \epsilon_{RD}$.

Thus the security margin provided by the modified T-box of CMEA-I against Linear Cryptanalysis is at least as much as the AES S-box.

# 6 Efficiency of CMEA-I

In this section we compare the efficiency of the design with respect to the original CMEA algorithm, which is known to be suited for the telecommunication industry. The CMEA algorithm is optimized for 8 bit micro-processors with severe resource limitations [13]. The CMEA algorithm has been modified in the following three places in order to prevent successful cryptanalysis of CMEA-I.

- The update equation of $P_i$ has been changed to $P_i'=P_i+T(y_i \oplus ((2i)\%n))$.
- The CAVE Table is replaced with the AES S-box.
- The number of rounds of T-box has been increased to eight rounds.

The first change is a minor functional change and does not require any extra computation with respect to the CMEA algorithm. The multiplication by 2 is a simple shift left operation and hence has no negative effect on the efficiency of the original CMEA algorithm.

The second change advocates the replacement of the CAVE Table with the AES S-box. The AES S-box, unlike the DES S-box and the CAVE Tables can be implemented through a compact algebraic equation [2]. The structured algorithm of AES S-box makes it amenable to efficient implementations both in hardware and software [3, 6, 9, 8]. Still the S-box of Rijndael is secured as it has withstood lot of cryptanalysis [2].

The third modification of CMEA-I is the increase of the number of rounds in T-box from four to eight. But this increase in number of rounds does not incur any penalty on the computational cost as the replacement of CAVE Table by AES S-box leads to extremely fast designs [7].

# 7 Conclusion

In the present paper the original CMEA algorithm has been modified into CMEA-I. The paper shows how the existing cryptanalysis of CMEA fails to break CMEA-I. It has been shown that the T-box provides sufficient security margin to the cipher CMEA-I in the face of linear and differential cryptanalysis. In short, the paper demonstrates that with suitable modifications the original CMEA algorithm can be made strong and hence can be suitable for wireless security.

# References


[1] E. Biham and A. Shamir, "Differential cryptanalysis of DES like cryptoSystems," *Journal of Cryptology*, vol. 4, pp. 3-72, 1991.

[2] J. Daemen and V. Rijmen, *The Design of Rijndael*, Springer-Verlag, 2002.

[3] B. Gladman, "Implementations of AES (Rijndael) in C/C++ and assembler," 2007. (http://fp.gladman.plus.com/ cryptography_technology/rijndael)

[4] H. M. Heys, "A tutorial on linear and differential cryptanalysis," 2007. (www.engr.mun.ca/ howard/PAPERS/ldc_tutorial.ps).

[5] M. Matsui, "Linear cryptanalysis method for DES cipher," in *Advances in Cryptology (Eurocrypt'93)*, LNCS 765, pp. 386-397, Springer-Verlag, 1993.

[6] S. Morioka and A. Satoh, "An optimized S-box circuit architecture for low power AES design," in *Proceedings of Cryptographic Hardware and Embedded Systems*, pp. 271-295, Springer-Verlag, Aug. 2002.

[7] S. Morioka and A. Satoh, "A 10-Gbps full-AES crypto design with a twisted BDD S-box architecture," *IEEE Transactions on VLSI Systems*, vol. 12, no. 7, pp. 686-691, July 2004.

[8] D. Mukhopadhyay and D. RoyChowdhury, "An efficient end to end design of Rijndael cryptosystem in 0.18 *mu* CMOS," in $18^{th}$ International Conference on VLSI Design, pp. 405-410, Jan. 2005.

[9] V. Rijmen, "Efficient implementation of the Rijndael-Sbox," 2007. (http://www.esat.kul euven.ac.be/ rijmen/rijndael)

[10] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, pp. 379-423 and 623-656, 1948.

[11] D. Stinson, *Cryptography, Theory and Practice*, Chapman & Hall/CRC, 2002.

[12] TIA Telecommunications Industry Association, *Common Cryptographic Algorithms*, Revision D.1, Publication Version, Sept. 13, 2003. (http://ftp.tiaonline.org/TR-45/TR45AHAG/Public/ComCryptAlgD1.pdf)

[13] D. Wagner, B. Schneier, and J. Kelsey, "Cryptanalysis of the cellular message encryption algorithm," in *Crypto'97*, pp. 526-537, 2002.

**Debdeep Mukhopadhyay** is currently an Assistant Professor in the Department of Computer Sc and Engg, Indian Institute of Technology, Madras, India. During the time of doing the presented research in the paper, he was doing his PhD from the Department of Computer Science and Engg, Indian Institute of Technology, Kharagpur, India. He received his Master of Science from the Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India in the year 2004 and his B.Tech from the Department of Electrical Engineering, India Institute of Technology, Kharagpur, India in the year 2001. His research interests are in the fields of Cryptology, Celluar Automata, VLSI Design and Testing.

**Dipanwita Roy Chowdhury** is a Professor in the Department of Computer Science and Engineering, Indian Institute of Technology, Indian Institute of Technology, Kharagpur, India. She received her PhD from the Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur, India. She completed her M.Tech and B.Tech from Calcutta University, India in the year 1989 and 1987 respectively. Her research interests are in the fields of Cellular Automata, Cryptography, Error correcting codes and VLSI Design and Test.