



# Broken AES Cipher E-CORP

D0pp3lgang3r

17 mai 2022

## 1 Introduction

Hey, Boss I'm sending you the new kind of AES Block Cipher that I invented, for E-CORP.

Do you think this cipher algorithm is breakable? Try to break it yourself!

## 2 Cipher Algorithm

### 2.1 Dividing message into matrices

We start by taking 64 bytes of the message we want to cipher, and then put their decimal value into the big matrix M.

$$M_{8,8} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,8} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,8} \\ \vdots & \vdots & \ddots & \vdots \\ a_{8,1} & a_{8,2} & \cdots & a_{8,8} \end{pmatrix}$$

Afterward we create 4 matrices M1, M2, M3, M4 and put each coefficients from our big matrix M, inside of them, so we get :

$$M1_{4,4} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{pmatrix}, \quad M2_{4,4} = \begin{pmatrix} a_{1,5} & a_{1,6} & a_{1,7} & a_{1,8} \\ a_{2,5} & a_{2,6} & a_{2,7} & a_{2,8} \\ a_{3,5} & a_{3,6} & a_{3,7} & a_{3,8} \\ a_{4,5} & a_{4,6} & a_{4,7} & a_{4,8} \end{pmatrix}$$

$$M3_{4,4} = \begin{pmatrix} a_{5,1} & a_{5,2} & a_{5,3} & a_{5,4} \\ a_{6,1} & a_{6,2} & a_{6,3} & a_{6,4} \\ a_{7,1} & a_{7,2} & a_{7,3} & a_{7,4} \\ a_{8,1} & a_{8,2} & a_{8,3} & a_{8,4} \end{pmatrix}, \quad M4_{4,4} = \begin{pmatrix} a_{5,5} & a_{5,6} & a_{5,7} & a_{5,8} \\ a_{6,5} & a_{6,6} & a_{6,7} & a_{6,8} \\ a_{7,5} & a_{7,6} & a_{7,7} & a_{7,8} \\ a_{8,5} & a_{8,6} & a_{8,7} & a_{8,8} \end{pmatrix}$$

Then we set up a KEY Matrix, full of decimal values :

$$K = \begin{pmatrix} 36 & 33 & 36 & 35 \\ 40 & 60 & 62 & 41 \\ 38 & 35 & 43 & 42 \\ 45 & 47 & 40 & 41 \end{pmatrix}$$

## 2.2 Algorithm

Now, that everything is initialized can we start to have a look at, what the algorithm is actually doing?

### 2.2.1 Rotations

Firstly we rotate left M1, M2, M3, M4 so M1 becomes :

$$M1_{4,4} = \begin{pmatrix} a_{1,4} & a_{2,4} & a_{3,4} & a_{4,4} \\ a_{1,3} & a_{2,3} & a_{3,3} & a_{4,3} \\ a_{1,2} & a_{2,2} & a_{3,2} & a_{4,2} \\ a_{1,1} & a_{2,1} & a_{3,1} & a_{4,1} \end{pmatrix}$$

And we repeat this operation for M2, M3 and M4.

### 2.2.2 Xoring

Then we xor each coefficient of each matrix with the coefficient of the key so basically, M1 becomes :

$$M1_{4,4} = \begin{pmatrix} a_{1,4} \oplus 36 & a_{2,4} \oplus 33 & a_{3,4} \oplus 36 & a_{4,4} \oplus 35 \\ a_{1,3} \oplus 40 & a_{2,3} \oplus 60 & a_{3,3} \oplus 62 & a_{4,3} \oplus 41 \\ a_{1,2} \oplus 38 & a_{2,2} \oplus 35 & a_{3,2} \oplus 43 & a_{4,2} \oplus 42 \\ a_{1,1} \oplus 45 & a_{2,1} \oplus 47 & a_{3,1} \oplus 40 & a_{4,1} \oplus 41 \end{pmatrix}$$

We do the same for others...

### 2.2.3 Trace Addition

Next we improve the "security", by adding the trace of K divided by 2 [let  $tr = \frac{trace(K)}{2}$ ], to each xored coefficients of each matrix, M1 becomes :

$$M1_{4,4} = \begin{pmatrix} (a_{1,4} \oplus 36) + tr & (a_{2,4} \oplus 33) + tr & (a_{3,4} \oplus 36) + tr & (a_{4,4} \oplus 35) + tr \\ (a_{1,3} \oplus 40) + tr & (a_{2,3} \oplus 60) + tr & (a_{3,3} \oplus 62) + tr & (a_{4,3} \oplus 41) + tr \\ (a_{1,2} \oplus 38) + tr & (a_{2,2} \oplus 35) + tr & (a_{3,2} \oplus 43) + tr & (a_{4,2} \oplus 42) + tr \\ (a_{1,1} \oplus 45) + tr & (a_{2,1} \oplus 47) + tr & (a_{3,1} \oplus 40) + tr & (a_{4,1} \oplus 41) + tr \end{pmatrix}$$

We do the same for others...

### 2.2.4 Inversion

Then we inverse the coefficients  $k_{i,j}$ , that we obtain from the previous operations so  $(a_{1,4} \oplus 36) + tr$  becomes  $k_{1,1}$  and M1 becomes :

$$M1_{4,4} = \begin{pmatrix} k_{4,4} & k_{4,3} & k_{4,2} & k_{4,1} \\ k_{3,4} & k_{3,3} & k_{3,2} & k_{3,1} \\ k_{2,4} & k_{2,3} & k_{2,2} & k_{2,1} \\ k_{1,4} & k_{1,3} & k_{1,2} & k_{1,1} \end{pmatrix}$$

Don't forget to do the same for each matrices !

### 2.2.5 Writing the ciphered data

Here is the function, to write ciphered data sequentially...

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define SUB_MATRIX_N 4
5 #define SUB_MATRIX_COL 4
6 #define SUB_MATRIX_ROW 4
7
8 void write_ciphered(Matrix **ma_tab, FILE *fp)
9 {
10     // ma_tab is the table of matrices {M1, M2, M3, M4}
11     for (int k=0;k<SUB_MATRIX_N;k++)
12     {
13         for (int i=0;i<SUB_MATRIX_ROW;i++)
14         {
15             for (int j=0;j<SUB_MATRIX_COL;j++)
16             {
17                 putc((int)ma_tab[k]->matrix[i][j], fp);
18             }
19         }
20     }
21     fclose(fp);
22 }
```

If you need help don't hesitate to contact me : Doppelganger#5878 on Brainshell !