COMPUTER SCIENCE & ENGINEERING
IIT KANPUR

Department of Computer Science and Engineering

Assignment #5

Enciphered

Roll: *21111261 —— 21111263 —— 21111063*

Class: *CS 641A*

Session: *Semester- II*

Email: *anindyag21@iitk.ac.in; anindyag@cse.iitk.ac.in*

Course: *CS 641A: Modern Cryptology* – Teacher: *Prof. Manindra Agrawal*

Submission date: *16th-Feb*

Problem 2

List the commands used in the game to reach the ciphertext.

Solution:

```
==> go ==> wave ==> dive ==> go ==> read
```

Problem 3

Give a detailed analysis of how you figured out the password? (Explain in less than 500 words)

Solution:

Facts

1. Screen suggest that, we are using block cipher
2. Size of the block is 8 bytes
3. Working finite field is $\mathbb{F}_{128}$. It is constructed using a degree 7 irreducible polynomial $x^7 + x + 1$. Arithmetic operation over $\mathbb{F}_{128}$ will be used.
4. Given two linear transformation

- A: Given by an invertible 8×8 key matrix with elements from $\mathbb{F}_{128}$.
- E: An exponentiation given by 8×1 vector whose elements are numbers between 1 and 126.

5. Encrypted Password: `lhkrktlnhojqfhfimjmmlthhgkfhimhj` This is obtained by typing password.

Encoding

We have clearly observe the ciphertext belongs to the range from f to u. There are 16 letters in this range. Now we map each letters to a four-bit string. Let provide a mapping of a letter to a 4-bit binary string.

f: 0000; g: 0001; h: 0010; i; 0011 j: 0100; k: 0101; l: 0110; m: 0111; n: 1000; o: 1001; p: 1010; q: 1011; r: 1100; s: 1101; t: 1110; u: 1111;

So, a byte contains two characters. Thus the input was of 8-bytes, however we are working on $\mathbb{F}_{128}$. Thus, we could have inputs from `ff` to `mu` considering MSB for any letter will be 0.

"Generate-inputs.ipynb" helps to generate all the possible plaintext. We write down these all plaintext in "plaintexts.txt".

Methodology

Structural cryptanalysis is the part of cryptography which explores the security of cryptosystems described by generic block diagrams. It explains the syntactic interaction between the various blocks. However it ignores their semantic definition as particular functions. Typical examples include meet in the middle attacks on double encryptions, the study of various chaining structures, and the properties of Feistel structures with a small number of rounds.

In this work we are studying structural cryptanalysis of EAEAE problem. This cryptosystem takes 8×1 vector of size 8 bytes over the field as input. Here we tabled1 a list of observation regarding input and output.

Let use explain a bit more. Suppose the plaintexts are denoted as $m_0, m_1, \cdots, m_7$ and ciphertext denoted as $c_0, c_1, \cdots, c_7$.

1. Input text: `ff ff ff ff ff ff ff ff`
   $\Rightarrow$ output text `ff ff ff ff ff ff ff ff`
2. Suppose input has a nonzero byte m_j and zero byte m_i with $\forall i \neq j$. We observed that the corresponding ciphertexts has $c_i = 0 \; \forall \; i < j$.
3. Now we change the input after k byte, then output got changed.

Input	Output
plaintext has all zero	ciphertext has all zero
first i byte of input plaintext were zero	irst i byte of input ciphertext were zero
if changed plaintext at ith bit	ciphertext at i th bit got changed

Table 1: Input-output relation

From these observation we draw ca conclusion that each row having zeros has at the end of that row. This statement tells us that A should be lower triangular. Suppose $(a_{i,j})$ denotes $i-j$-th elements of the matrix A and e_i stands for i-th element of the 8×1 vector E.

Generations of input plaintext follows the expression $C^{i-1}PC^{8-i}$. Here we have deployed 128×8 plaintext, all these are saved in `pt_text.txt`.

[See each input had maximum one non-zero block at a time. From the encoding we know that there are 128 bit possible plaintext. So for each block, we have 128 possible plaintext values. Therefore for eight block, it needs 8×128 possible plaintext values.][

We include `Python pexpect`, it helps to connect to the server after verifying the credentials. The ciphertext from plaintext is stored in `ct_text.txt`.

Earlier we mentioned that A should be a lower triangular matrix. Our input format suggests that one block is nonzero per input block. Suppose the value of the nonzero per input, suppose the value of the nonzero block i of a input is x. Now one can run over all possible values of diagonal elements and exponents. At the end, the output block has the value

$$v = (a_{i,i} \times (a_{i,i} \times x^{e_i})^{e_i})^{e_i} \qquad \cdots \quad (i)$$

Deploying the above equation, we able to compute the diagonal entries of A and all entries of E-vector. Earlier we mentioned that the linear transformation is lower triangular, this implies that for $i > j$, i-th input block is relies on the j-th output block.

During the brute force attack, multiplication and exponent operations are performed in alternative manner to verify the encrypted data is same as ciphertext or not.

Right now, a list of plaintext, ciphertext pairs we have in our hand. Now performed the computation for all possible diagonal entries $a_{i,i}[0, 2]$ and $e_i[0, 2]$ and observes if input maps to the outputs. We maintain a list in this purpose (see table 3).

Next task is to discard some pairs from the above list and compute non-diagonal entries of the matrix A. For this purpose we deploy more plaintext and ciphertext pair and repeat over the above pairs for computing the element within the range $[0, 127]$ so that it fulfills the equation (1). The method proceeds in a triangular manner so that the value of $a_{i,i}$ and $a_{j,j}$ helps to compute $a_{i,j}$. Therefore at the end we able to compute

Block Number	**Possible values of** $a_{i,i}[0, 2]$	**Possible values of** $e_i[0, 2]$
0	[84, 67]	[20, 108]
1	[8, 115, 70]	[46, 96, 112]
2	[43, 78, 87]	[36, 42, 49]
3	[12, 8, 104]	[78, 85, 91]
4	[118, 112, 20]	[40, 89, 125]
5	[11, 41, 127]	[53, 83, 118]
6	[27, 71, 92]	[23, 48, 56]
7	[104, 38, 38]	[1, 19, 107]

Table 2: Possibility of $a_{i,i}$ and e_i

elements next to each diagonal entries and one entry at every diagonal position of A and the vector E (see table 3).

The above procedure bring down all possible pairs to one element each as only for some . We provide element next to the diagonal entry. Thus we have successfully computed every element of the matrix by running over all possible values (0 to 127) and applying the final values of E vector and the above computed the values of L.T. matrix and verified the validity of the equation (i).

The linear transformation matrix is E vector:

$$[20, 112, 36, 78, 89, 53, 23, 19]$$

A matrix:
$$\begin{pmatrix} 84 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 114 & 70 & 0 & 0 & 0 & 0 & 0 & 0 \\ 16 & 16 & 43 & 0 & 0 & 0 & 0 & 0 \\ 127 & 21 & 25 & 12 & 0 & 0 & 0 & 0 \\ 98 & 39 & 13 & 116 & 112 & 0 & 0 & 0 \\ 30 & 52 & 19 & 46 & 101 & 11 & 0 & 0 \\ 3 & 121 & 10 & 105 & 0 & 93 & 27 & 0 \\ 89 & 14 & 83 & 23 & 22 & 69 & 24 & 38 \end{pmatrix}$$

Thus the task is almost done. In similar fashion we may proceed further to decrypt the password. Just run over all possible values for a block and verify with the output of given EAEAE function is same as the current password block we have .

The decrypted password is `tqmtazppbc`

References

[1] Biryukov, Alex, and Adi Shamir. "Structural cryptanalysis of SASAS." International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2001.

Block Number	**Possible values of $a_{i,i}[0,2]$**	**Possible values of $e_i[0,2]$**
0	[84]	[20]
1	[70]	[112]
2	[43]	[36]
3	[12]	[78]
4	[118]	[89]
5	[11]	[53]
6	[27]	[23]
7	[38]	[19]

Table 3: Possibility of $a_{i,i}$ and e_i