ASSIGNMENT-5

Team Name: codemeisters

Members:

VIJIT MALIK (170791)

HITESH KUMAR (170305)

PRAVEEN KUMAR (170504)

STEPS:

1. IDENTIFICATION OF THE TYPE OF ENCRYPTION.

We entered go-wave-dive-go-read and we got to the problem. It was given that the problem was similar to AES.

2. OBTAINING A matrix and E vector:

The transformation was given as EAEAE, where E is element wise exponentiation of 8 byte vector and A was a linear transformation by multiplication by a matrix. The multiplication was defined in F128 as polynomial multiplication modulo an irreducible polynomial. Similar to DES in Assignment 4 the encoding here was also done as f corresponding to 0, g corresponding to 1 and so on… until u corresponds to 15.
We observed that putting 'f' before any input does not change the output so f is basically used as padding here.
We recognized A as a lower triangular matrix because when we gave it the inputs of the form 'ff_____' it's output was 'ff_____' and for 'ffff_____' output was 'ffff_____' which implies that the ith byte of the cipher depends only on the first i bytes of the plain text. This proves that A is a lower triangular matrix.
Now we proceeded to find the diagonal elements of the matrix A first.

We used total of 24 chosen plaintexts :

1. 8 have '1' on ith position where i varies from 1 to 8. Rest are all 0.
2. 8 have '2' on ith position where i varies from 1 to 8. Rest are all 0.
3. 8 have '3' on ith position where i varies from 1 to 8. Rest are all 0.

If we have x as the non-zero input byte at the ith position then the encrypted cipher text ith byte is given as (A[i][i] *((A[i][i] * (x^E[i]))^E[i]))^E[i]. Iterating through all 128 values of A[i][i] from 0 to 127 and values of E[i] from 1 to 126 we get some possible pair of values of A[i][i] and E[i].
after using i from 1 to 8 we had 3 pairs of (A[i][i],E[i]) each for each i. Now in order to eliminate those pairs we made more equations and found other values of A[i][j] and subsequently eliminating invalid pairs. For arithmetic in F128 we used the pyfinite library of python.
We found A and E as

A = [100 0 0 0 0 0 0 0]
[122 56 0 0 0 0 0 0]
[6 121 40 0 0 0 0 0]
[10 97 77 50 0 0 0 0]
[58 14 78 10 16 0 0 0]
[9 76 114 116 92 87 0 0]
[104 30 98 92 104 44 14 0]
[13 91 54 58 113 17 37 103]

E = [85 52 38 72 116 38 66 50]

3. BREAKING THE ENCRYPTED TEXT:

We broke the password without even using A and E. As we knew that the ith byte of ciphertext depends only upon the first i bytes of the plain text. First, divide the password into 2 equal halves and use the same strategy for both.
First byte has 128 possibilities (due to finite field). Iterating over all such possibilities and appending each with f's and giving it as input in the game we get 128 outputs out of which one has it's first byte same as the password's first byte. So we have found 1st byte of plain text. Now 2nd byte of cipher depends only upon the 1st and 2nd byte of plain text. We know the first byte so now we iterate over all 128 possibilities of the 2nd byte and give these as inputs to the game and find 2nd byte.
The process is repeated until we find all the 8 bytes.
To obtain ciphertext from the game fast we created an API.

We had password: ***ktirlqhtlqijmmhqmgkplijngrluiqlq***

Obtained Plain text: ***lhlgmjmkmglqlompmoltmglilqlmlgmh***

But when we entered it into the game it did not work. So we converted the consecutive pairs into bytes and then their decimal values. We found their decimal values to be ranging from the ASCII values of 'a' to 'z'. So, we converted the decimal values (taken as ASCII values) into their corresponding character and we got.

batuqkizynqckgar

When we entered it into the game we reached chapter 6.