CS641A: Modern Cryptology Assignment 5

objectStrongly

Dewansh Singh(170241)

Nikhil Naidu(170758)

Parv jain (170464)

- To get to the cipher text in this level, we used these commands in the following order : “go”, “wave”, “dive”, “go”, “read”.
- As given in the puzzle, the encryption method is a weak-modification of AES, and quite similar to SASAS. It is mentioned that the field is GF(2^7).
- Similar to the previous assignment, the input and output are given in alphabets starting from ‘f ‘to ‘u’, each character corresponding to 0 to 15.
- It was observed that whenever a chosen plaintext of a 8 byte vector was passed with the first n bytes equal to zero the first n bytes of the ciphertext also came out to be zero. From this, we can mathematically show that the upper half of the matrix A are all zeroes. Hence the given matrix A is a lower-triangular matrix.
- We have used fieldmath.py that is available online, for performing field operations on the given inputs.
- We used the simple observation that if the matrix is lower triangular and an input like [t,0,0,0,0,0,0,0] was used the corresponding output should be (a 1 k.(k+1)) * (t k.k.k), where a 1 is the first diagonal entry of the matrix and k is the first entry in the Exponentiation E vector(where . denotes usual multiplication of integers and * denotes multiplication in F 128).
- We took 8 input pairs in input_16.txt in such a way that each input pair corresponds to a possible set of values for diagonal elements of matrix A.
- Firstly, the diagonal elements and the exponentiation layers were solved using chosen plaintext attack. we took plaintexts of the form [0,...0,1,...] and [0,...0,2,...], where each element represents one byte. Taking 2 inputs [t 1 ,0,0,0,0,0,0,0] and [t 2 ,0,0,0,0,0,0,0] and observing 1st byte of output ciphertexts, say out1 and out2.
- As a1 and k are unknown, by brute-force method in D&E.py, we obtained these following values:
- Possible diagonal elements:

  [[19, 26, 89], [90,46,46], [57, 66, 77], [36,101,33], [40, 115,32], [46,100,35], [4,36,25], [118,12,78]]

- Corresponding possible exponentiation elements are:

[[33,102,119], [1,19,107], [72,84,98], [36,42,49], [65,92,97], [39,106,109], [6,7,114], [53,83,118]]

- In the file solver.py, we now try to find the unknown entries of the elements in matrix A. the main idea behind finding the unknown values is by using the possible values of diagonal elements and using a brute-force way to find the actual value.
- In the file input_nondiag.txt, inputs are given such that the non-zero byte in the input array corresponds to position of unknown element in matrix A.
- For the first step, taking inputs of the form; [6, 0, ..., 0, 0, 0, ..., 0] and [2, 0, ..., 0, 0,0, ..., 0], with 6 and 2 in the first byte, and encrypted these in solver.py using the possible values of diagonal entries and exponential values and matched the outputs to the actual output to get desired values. In this case we obtained the values of A[0][0]=26 A[1][1]=46 A[1][0]=71 .
- Now similarly using these kinds of inputs [0, 0, ..., 0, 6, 0, ..., 0] and [0, 0, ..., 0, 2,0, ..., 0], with 6 and 2 in the i-th place and comparing with corresponding row-th byte of ciphertext.(0< i< 8)

- We proceed row by row by finding the values (1,0)->(2,1)->(2,0)->(3,2)->(3,1)->(3,0)->(4,3)->.........->(7,0)
- For the file solver.py, we require about 70 pair of inputs, each corresponding to single entry in matrix A.
- The key matrix was found to be:

26	0	0	0	0	0	0	0
71	46	0	0	0	0	0	0
50	75	57	0	0	0	0	0
112	0	79	101	0	0	0	0
41	119	91	116	40	0	0	0
120	7	12	60	58	100	0	0
11	112	37	15	13	118	4	0
25	32	106	4	75	13	11	118

- The exponentiation matrix was found to be:

  [102,107,72,42,65,106,6,53]

- Then we used final.py' to decrypt the "password" using the inverse of the key matrix (computed using the module fieldmath.py) and the inverse of the Exponentiation operation.
- Encrypted Password ="fgffgnkqjfjlgrkolglnkklqirlgjlkh"
- Decrypted Password =[99, 108, 113, 122, 120, 108, 111, 111, 97, 121, 113, 121, 122, 122, 108, 121]
- Since the final decrypted password contained numbers from 97 to 122 we thought they might be ASCII values for the final answer.

- Hence, Final Answer = “clqzxlooayqyzzly”