**Description for: CS 492 Homework 2**

Symmetric Cryptography – Block and Stream ciphers

Complete the problems below and submit this word document with the solution to the questions here. Also submit the source code **you** created to solve problem 1 as separate source files.

Name: Antonio Zea

**Problem 1 TEA Block encryption (50 pts):**

**YOU** implement the Tiny Encryption Algorithm (TEA) in Java, Python, or C++ from scratch not using any security libraries. There are many implementations out there, but make sure this implementation is **completely your own or expect consequences and an Academic Misconduct report to be filed**. Your understanding of the details of how the bit operations matter is critical to your understanding of the intricacies of implementing computer security.

In addition to submitting this word document you need to submit your commented source code.

**Part 1:** Use your TEA algorithm to encrypt the 64-bit plaintext block:

0x0FCB45670CABCDEF

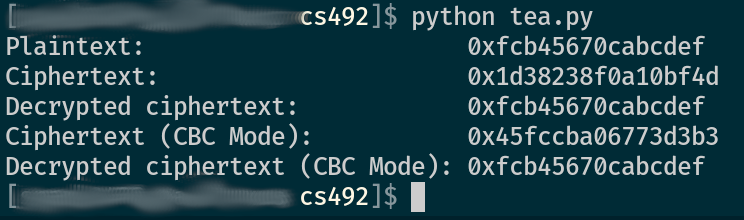
Using 128 bit key:

0xBF6BBBCDEF00F000FEAFAFBFACCDEF01

Enter your cipher text here:

0x1d38238f0a10bf4d

**Part 2:** Implement decryption and verify that you obtain the original plain text

Plaintext: 0xfcb45670cabcdef

Ciphertext: 0x1d38238f0a10bf4d

Decrypted ciphertext: 0xfcb45670cabcdef

Ciphertext (CBC Mode): 0x45fccba06773d3b3

Decrypted ciphertext (CBC Mode): 0xfcb45670cabcdef

**Part 3:** In comments in your code explain how you would make your code encode/decode a longer set of plaintext (i.e. multiple blocks) using CBC. You do not need to code this, but your explanation should be detailed related to your implementation.

I have coded the solution and included my comments here as well.

To encrypt in CBC mode, I would start by creating an initialization vector. This value will be XORed with the plain text and the result fed into the TEA block cipher. The result from the block cipher will be added to the ciphertext and take the place of the IV in the encryption of the next block of plaintext. This process continues until all blocks of plaintext have been encrypted.

To decrypt in CBC mode, the process is essentially the reverse of the encryption steps. We start by passing the ciphertext into the TEA block cipher. Those results are then XORed with the same IV that was used to encrypt. At this point, you have a block of plaintext. After this phase, the following phases have the following blocks of ciphertext take the place of the IV and the process continues until all ciphertext blocks have been decrypted.

**Problem 2 Chain block encryption (20 pts)**

Given the following message and a block size of 2 bits, encode it using ECB, CBC, and CTR (make sure to indicate which is which)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Use the following substitution cipher | | | | Plain | 00 | 01 | 10 | 11 |
|  |  |  |  | Encoded | 01 | 00 | 11 | 10 |
| IV | 10 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| ECB |  |  |  |  |  |  |  |  |
| Message | 01 | 00 | 11 |  |  |  |  |  |
| Ciphertext | 00 | 01 | 10 |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| CBC |  |  |  |  |  |  |  |  |
| Message | 01 | 00 | 11 |  |  |  |  |  |
| Ciphertext | 10 | 11 | 01 |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| CTR |  |  |  |  |  |  |  |  |
| Message | 01 | 00 | 11 |  |  |  |  |  |
| Ciphertext | 01 | 11 | 01 |  |  |  |  |  |

**Problem 3 Stream encryption (30 pts):**

You have a 4 bit LFSR with tap sequence [3,2,0] your initial key is 1101 encrypt the message “0101” and enter it below.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Bit 0** | Bit 1 | **Bit 2** | **Bit 3** | Key stream | Plaintext | Ciphertext |
| 1 | 1 | 0 | 1 |  |  |  |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 |
|  | 1 | 1 | 0 | 1 | 1 | 0 |

Encrypted message: 1110