Birzeit University
Faculty of Engineering and Technology
Department of Electrical and Computer Engineering
ENCS4320 - Applied Cryptography (Term 1232)
Homework # 2 *(Programming Assignment)* – Due Monday, June 03, 2024

Description:

The tiny encryption algorithm (**TEA**) is a *symmetric key block cipher* designed for simplicity and efficiency, especially in resource-constraint environments. The TEA uses a **64-bit block length** and a **128-bit key**. The algorithm assumes a computing architecture with 32-bit words, ***all operations are implicitly modulo 2^{32}*** (i.e., any bits beyond the 32nd position are automatically truncated). The number of rounds is *variable* but must be relatively large. The conventional wisdom is that *32 rounds are secure*. However, each round of TEA is more like two rounds of a *Feistel cipher*, such as the data encryption standard (**DES**), as shown in Figure 1, so this is roughly equivalent to 64 rounds of DES.

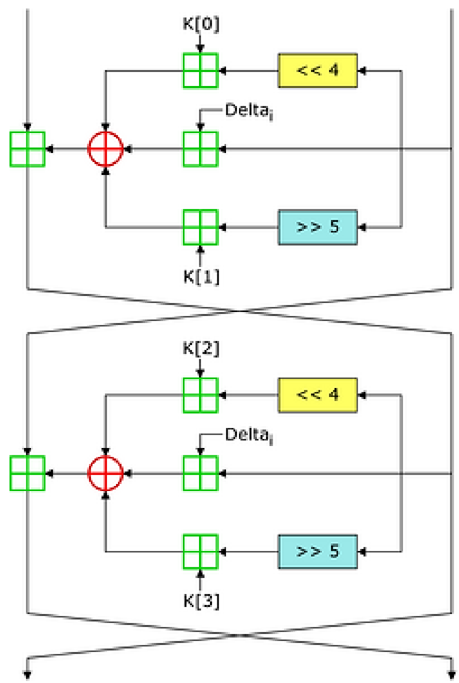

Figure 1: Cycle i of TEA Encryption Algorithm (two Feistel rounds).

In block cipher design, there is an inherent **trade-off** between the *complexity of each round* and the *number of rounds required*. Ciphers such as DES try to strike a balance between these two, while the advanced encryption standard (**AES**) reduces the number of rounds as much as possible, at the expense of having a more complex round function. In a sense, TEA can be seen as living at the opposite extreme of AES, since TEA uses a **very simple round function**. But as a consequence of its simple rounds, the number of rounds must be large to achieve a high level of security. TEA encryption and decryption algorithms, assuming 32 rounds are used, are shown in Table 1 and Table 2, respectively, where "≪" is a left (non-cyclic) shift and "≫" is a right (non-cyclic) shift.

Table 1: TEA Encryption.

$(K[0], K[1], K[2], K[3])$ = 128-bit **key**
(L, R) = **plaintext** (64-bit block)
$delta = 0x9E3779B9$
$sum = 0$
for i = 1 to 32 do:
$sum = sum + delta$
$L = L + \big(((R \ll 4) + K[0]) \oplus (R + sum) \oplus ((R \gg 5) + K[1])\big)$
$R = R + \big(((L \ll 4) + K[2]) \oplus (L + sum) \oplus ((L \gg 5) + K[3])\big)$
ciphertext = (L, R)

Table 2: TEA Decryption.

$(K[0], K[1], K[2], K[3])$ = 128-bit **key**
(L, R) = **ciphertext** (64-bit block)
$delta = 0x9E3779B9$
$sum = delta \ll 5$
for i = 1 to 32 do:
$R = R - \big(((L \ll 4) + K[2]) \oplus (L + sum) \oplus ((L \gg 5) + K[3])\big)$
$L = L - \big(((R \ll 4) + K[0]) \oplus (R + sum) \oplus ((R \gg 5) + K[1])\big)$
$sum = sum - delta$
plaintext = (L, R)

Requirements:

Using any programming language you prefer, implement both the electronic code book (**ECB**) mode and the cipher block chaining (**CBC**) mode of TEA (**TEA-ECB** and **TEA-CBC**) with 32 rounds for encryption and decryption. ***Leave the first 10 blocks unencrypted***. The *implementation should be based on your own genuine effort*. Additionally, your program should ask the user to enter the parameters for the TEA-ECB and the TEA-CBC (i.e., key, plaintext/ciphertext, and initialization vector (**IV**)). Test your implementation of both TEA-ECB and TEA-CBC by encrypting the following linked image and then decrypting the resulting ciphertext to show diagrams analogous to those in the slides (Chapter 4 – Slides 12 and 21).

Deliverables:

1) Submit a simple report including a copy of your diagrams.
2) Submit a **well-documented** soft copy of your implementation along with a **readme file** on how to execute your implementation.

GOOD LUCK