Practical Session 2 – Tiny Encryption Algorithm

Criptografía
Facultad de Ingeniería
Departamento de Computación

The **Tiny Encryption Algorithm** (TEA) is a symmetric-key block cipher very simple to describe and implement, (typically a few lines of code). It was designed by David Wheeler and Roger Needham of the Cambridge Computer Laboratory; it was first presented at the Fast Software Encryption workshop in Leuven in 1994 and first published in the proceedings of that workshop [1]. It is not subject to any patents. TEA is known for its compactness and efficiency, as it requires minimal computational resources and has a small code footprint. Despite its simplicity, TEA has shown to offer a reasonable level of security for various applications that require lightweight encryption.

TEA operates on two 32-bit unsigned integers (could be derived from a 64-bit data block) and uses a 128-bit key. It has a Feistel structure with a suggested 64 rounds, typically implemented in pairs termed cycles. It has an extremely simple key schedule, mixing all the key material in the same way for each cycle. Different multiples of a magic constant are used to prevent simple attacks based on the symmetry of the rounds.

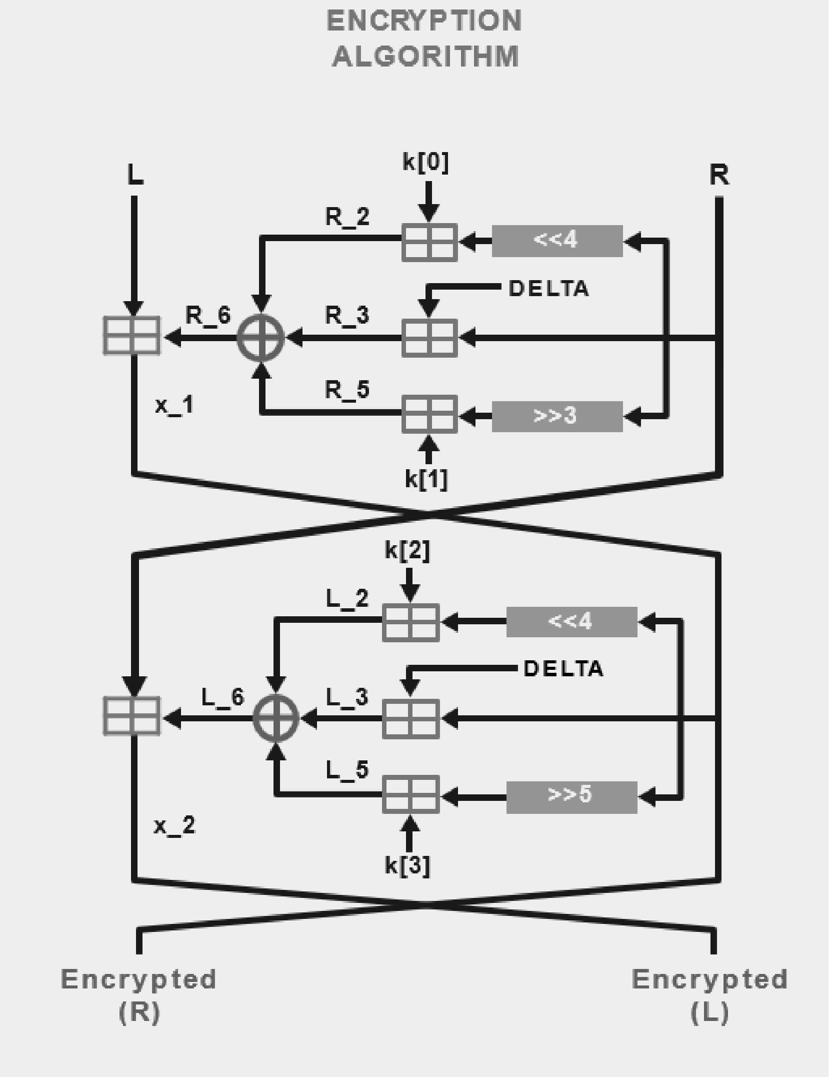
Pseudo code

Here is the pseudo code for encryption:

```
TEA_Encrypt(v0, v1, key):
    sum = 0
    delta = 0x9E3779B9
    k0, k1, k2, k3 = key

    for i = 1 to 32:
        sum = (sum + delta)
        v0 = (v0 + (((v1 << 4) + k0) XOR (v1 + sum) XOR ((v1 >> 5) + k1)))
        v1 = (v1 + (((v0 << 4) + k2) XOR (v0 + sum) XOR ((v0 >> 5) + k3)))

    return v0, v1
```

Here is the pseudo code for decryption:

```
TEA_Decrypt(v0, v1, key):
    sum = (delta * 32)
    k0, k1, k2, k3 = key

    for i = 1 to 32:
        v1 = (v1 - (((v0 << 4) + k2) XOR (v0 + sum) XOR ((v0 >> 5) + k3)))
        v0 = (v0 - (((v1 << 4) + k0) XOR (v1 + sum) XOR ((v1 >> 5) + k1)))
        sum = (sum - delta)

    return v0, v1
```



Test Vectors

This can be used as test vectors to test your own TEA implementation.

Plaintext (v0, v1)	Key (k0, k1, k2, k3)	Ciphertext (c0, c1)
0x12345678, 0x9ABCDEF0	0x01234567, 0x89ABCDEF, 0xFEDCBA98, 0x76543210	0xE8159ED0,0x1867CD84
0x00000000, 0x00000000	0x00000000, 0x00000000, 0x00000000, 0x00000000	0x41EA3A0A,0x94BAA940
0xFFFFFFFF, 0xFFFFFFFF	0xFFFFFFFF, 0xFFFFFFFF, 0xFFFFFFFF, 0xFFFFFFFF	0x319BBEFB,0x016ABDB2
0x11223344, 0x55667788	0xAABBCCDD, 0xEEFF0011, 0x22334455, 0x66778899	0xC7FE2275,0x8E6B25DC
0x89ABCDEF, 0x01234567	0x0F1E2D3C, 0x4B5A6978, 0x8697A6B5, 0xC3D2E1F0	0x09B428CB,0xF101AB09
0xDEADBEEF, 0xCAFEBABE	0x1337C0DE, 0x0BADF00D, 0xFACEB00C, 0xBA5EBA11	0x89EEE942,0x1DF8A5EB
0x10203040, 0x50607080	0x88776655, 0x44332211, 0xAABBCCDD, 0xEEFF0011	0x16BFA80F,0x76AB6B02
0x7F7F7F7F, 0x7F7F7F7F	0x7F7F7F7F, 0x7F7F7F7F, 0x7F7F7F7F, 0x7F7F7F7F	0xDA1D3A2D,0x42E6D0A9
0xDA1D3A2D, 0x42E6D0A9	0x0ACE0ACE, 0xDEEDBEEF, 0xABAD1DEA, 0xFACEFEED	0x3B4BC0AB,0x5167C682
0xCAFEBABE, 0xDEADBEEF	0xFEEDFACE, 0xC0DEC0DE, 0xBADF00D0, 0x1337C0DE	0x89EEE942,0x1DF8A5EB

Implementation

1. Make an individual submission of your implementation on Alphagrader using the programming language of your choice. The testing cases are the ones presented on the table above.

References

[1] Wheeler, David J.; Needham, Roger M. (16 December 1994). "*TEA, a tiny encryption algorithm*". Fast Software Encryption. Lecture Notes in Computer Science. Vol. 1008. Leuven, Belgium. pp. 363–366. doi:10.1007/3-540-60590-8_29. ISBN 978-3-540-60590-4.

[2] Al-Ajarmah, Mansour. (2023). Exploring the Tiny Encryption Algorithm: A Comparative Analysis of Parallel and Sequential Computation. International Journal of Scientific and Engineering Research. 8. 693. 10.1729/Journal.35208.