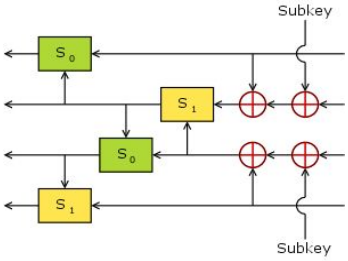
Cryptanalysis of FEAL 4

Sriram Balasubramanian(160070012), Shourya Pandey(160050013)

FEAL(Fast data Encipherment ALgorithm) is a block cipher proposed as a faster alternative to the Data Encryption Standard by Akihiro Shimizu and Shoji Miyaguchi. We have attacked FEAL 4 which uses four rounds of Feistel ciphers using differential cryptanalysis. 20 chosen plaintexts are used to recover the keys.

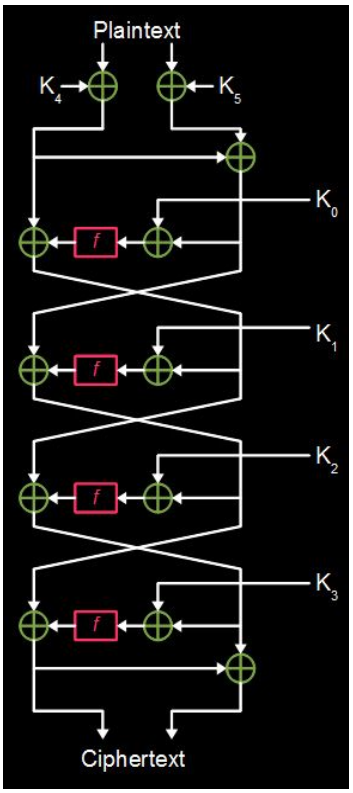
Implementation of FEAL 4

FEAL 4 is implemented as a four round Feistel cipher with a round function. The round function is best described by this diagram(denoted as f).



S0 and S1 are described as
S0(x, y) = Rot2(x + y mod 256) and S1(x, y) = Rot2(x + y + 1 mod 256).

These round functions are employed in the Feistel cipher using 6 keys(which can be divided into left and right halves) as follows:



In this setup, the left and right halves of the plaintext are xored with K4 and (K5 ⊕ K4 ⊕ P_left) respectively.
It then passes through the usual rounds of a Feistel cipher.
Decryption happens in the reverse order with key order reversed.

Cryptanalysis of FEAL 4

Reformulation of FEAL 4

The round function f is expressed in terms of the function G as described in the report. Subsequent cryptanalysis depends on the fast solution of the linear equations involving G.

Solving the linear equations involving G

The differential attack on FEAL 4 heavily relies on the need to solve the equations

$G(x \oplus a) = b,$

and the simultaneous equations

$G(x \oplus a) \oplus G(x \oplus b) = d$, $G(x \oplus a) \oplus G(x \oplus c) = e.$

Of course, we could simply have exploited the invertibility of the functions Si, but we want to give a general method to solve the above equations, mainly because of two reasons:

- To show that FEAL4 is a weak cipher no matter how Si is defined
- To motivate the solution of linear equations in G

Identifying differentials

Differentials which produce a fixed difference in the output are identified. These are 0x80800000 -> 0x02000000, 0x00008080 -> 0x00000002, 0x40400000 -> 0x01000000, 0x00004040 -> 0x00000001, 0x00000003

The attack we perform uses 20 chosen plaintexts to recover the key. The plaintexts we choose are in such a way so as to take advantage of the differentials that we identified; in particular, we exploit

$G(a) \oplus G(a \oplus 0x80800000) = 0x02000000$
$G(a) \oplus G(a \oplus 0x00008080) = 0x00000002$
$G(a) \oplus G(a \oplus 0x40400000) = 0x01000000$ or $0x03000000$
$G(a) \oplus G(a \oplus 0x00004040) = 0x00000001$ or $0x00000003$

After calculating the values M1, M2, M3, N1, N2, N3, we can do a final check by coding all twenty plaintexts and verifying it with the ciphertexts. The paper by Den Boer [2] describes how to recover the key from these values.

Of course, not all twenty plaintexts are needed to recover the key. Some of the plaintexts are only used for filtering the wrong paths. Not using them would of course mean that we would have to compute more possibilities for the various constants until later in the algorithm. This would increase the computing time.

Reference

1. The cryptanalysis of FEAL 4 with 20 chosen plaintexts [http://www.isg.rhul.ac.uk/~sean/feal.pdf]
2. B. Den. Boer, “Cryptanalysis of FEAL”