CS 406 Project
Cryptanalysis of Block Ciphers

Ankit Kumar Misra (190050020) and Kartikey Gupta (190050044)

August 16, 2021

Contents

1 What is Cryptanalysis?

Cryptanalysis is the mathematical study of cryptographic security systems, with the objective of finding hidden aspects which can be exploited to obtain information about the plaintext from the ciphertext without the use of a key.

Most cryptanalytic techniques are derived from two major types, namely linear cryptanalysis and differential cryptanalysis. Some popular extensions of these two include differential-linear, non-linear, CPA-linear, partial/truncated differential, and higher order differential cryptanalytic methods.

2 Linear Cryptanalysis

Linear cryptanalysis is a type of Known Plaintext Attack (KPA) that uses linear relations between input plaintexts and output ciphertexts, that hold true with a certain probability. These relations, along with a collection of plaintexts and corresponding ciphertexts, can be used to retrieve some information about the key(s) used in the cryptographic system.

Usually, block cipher encryption schemes are linear except for the parts involving S-boxes. S-boxes, or substitution boxes, are PRFs that are used to obscure the relationship between ciphertext and plaintext, and thus they are made non-linear. So, linear cryptanalysis boils down to finding approximate linear relationships between the input and output bits of the S-boxes used in the cryptosystem. Given an S-box which maps n bit inputs X to m bit outputs Y, there are $(2^n - 1)(2^m - 1)$ possible linear relations, some of which are more probable, and others are less probable. They are of the form:

$$X_{i_1} \oplus X_{i_2} \oplus \cdots \oplus X_{i_a} \oplus \cdots \oplus Y_{j_1} \oplus Y_{j_2} \oplus \cdots \oplus Y_{j_b} = 0$$

where $i_1, i_2, \ldots, i_a \in [n]$ are bits of X, and likewise $j_1, j_2, \ldots, j_b \in [m]$ for Y.

These linear relations over individual S-boxes are then combined over multiple S-boxes, so as to obtain just one most/least likely linear approximation that relates the plaintext, the ciphertext, and the key. Linear relations derived from different S-boxes can be combined, with probabilities computed using the Piling-Up Lemma:

Piling-Up Lemma: Given k independent Bernoulli random variables X_i, we have:

$$Pr[X_1 \oplus X_2 \oplus \cdots \oplus X_k = 0] = \frac{1}{2} + 2^{k-1} \prod_{i=1}^{k} \varepsilon_i \tag{1}$$

where:

$$\varepsilon_i = Pr[X_i = 0] - \frac{1}{2}$$

Using the Piling-Up Lemma, the probabilities of global linear approximations can be calculated, and the most/least probable one is used for the linear cryptanalytic attack; the one having probability most distance from $\frac{1}{2}$ is chosen.

Next, we move on to the actual cryptanalytic attack. A linear approximation over the first $r-1$ rounds of the cipher is constructed by combining linear relations from individual S-boxes. Then, ciphertexts are traced backwards in the rth round to obtain bitstrings which are combined with K_r, the rth round key. An exhaustive linear search is performed over the key space for K_r, and each time the linear approximation is tested for every plaintext being used. The key value which offers the maximum bias is selected as K_r, i.e., the key value for which the fraction of satisfying plaintexts is as far from 1/2 as possible is chosen as the extracted key.

The bitstrings are then decrypted through the rth round, and the same procedure is performed on the $(r-1)$th round to extract K_{r-1}, and so on.

The number of plaintext-ciphertext pairs required for a linear cryptanalytic attack on a block cipher is known to be of the order of $\frac{1}{\varepsilon^2}$, where ε is the value of the bias (difference of probability from $\frac{1}{2}$) of the best linear approximation for that cryptographic scheme, i.e., the linear relation derived from the procedure described above.

To prevent ease of linear cryptanalytic attacks, block cipher schemes should be designed using a larger number of S-boxes, and the biases of linear approximations of S-boxes should be kept minimal.

3 Differential Cryptanalysis

Differential cryptanalysis is a type of Chosen Plaintext Attack (CPA), which analyzes the effects of changes in the plaintext on changes in the resulting ciphertext. Basically, at each step, it produces two plaintext messages differing by some value (the difference is defined as XOR of the two bitstrings in most cases), and it observes the difference in the two ciphertexts generated by the encryptor. By analyzing such differences, the more probable key(s) can be computed, thus enabling an attack on the cryptographic system involved.

Linear operations in the scheme affect differentials in a bijective way, so once again the problem boils down to analyzing the effects of the non-linear S-boxes on differentials. For every possible input difference, the distribution of output differences is analyzed. Each S-box differential is a tuple $(\Delta X, \Delta Y, p)$ such that an input difference of ΔX gives an output difference of ΔY with probability p.

When this is done and the differential effects of each S-box are known, they are combined together to derive a useful differential characteristic for the entire cryptosystem, which holds with a high probability. This probability can be calculated simply as:

$$\prod_{i=1}^{n} p_i$$

where p_i is the probability of the differential propagation assumed at the ith S-box, and n is the number of S-boxes (assuming the differential characteristics of individual S-boxes are independent of each other).

Now, once a differential relation has been derived for the entire encryption scheme, the attack is designed. Suppose the encryption scheme occurs in r rounds. Then, a differential characteristic over the first $r-1$ rounds is devised. Plaintext pairs are generated having that particular difference, and ciphertext pairs are obtained from the encryptor. With backward propagation from the ciphertext, the bitstrings that were used with K_r, the key for the rth round, are calculated. Then, a search over the key space is done for K_r, and the differential characteristic is verified. The key value satisfying the differential characteristic the maximum number of times is chosen as the estimated value of K_r.

The same procedure is now extended backwards to calculate K_{r-1}, and so on. Thus, with some non-negligible probability, the keys involved in the block cipher scheme are extracted successfully.

It is known that the number of chosen plaintext pairs required for a differential cryptanalytic attack on the rth round is approximately $\frac{c}{p}$, where p is the probability of the characteristic being used over the first $r-1$ rounds, and c is some small constant.

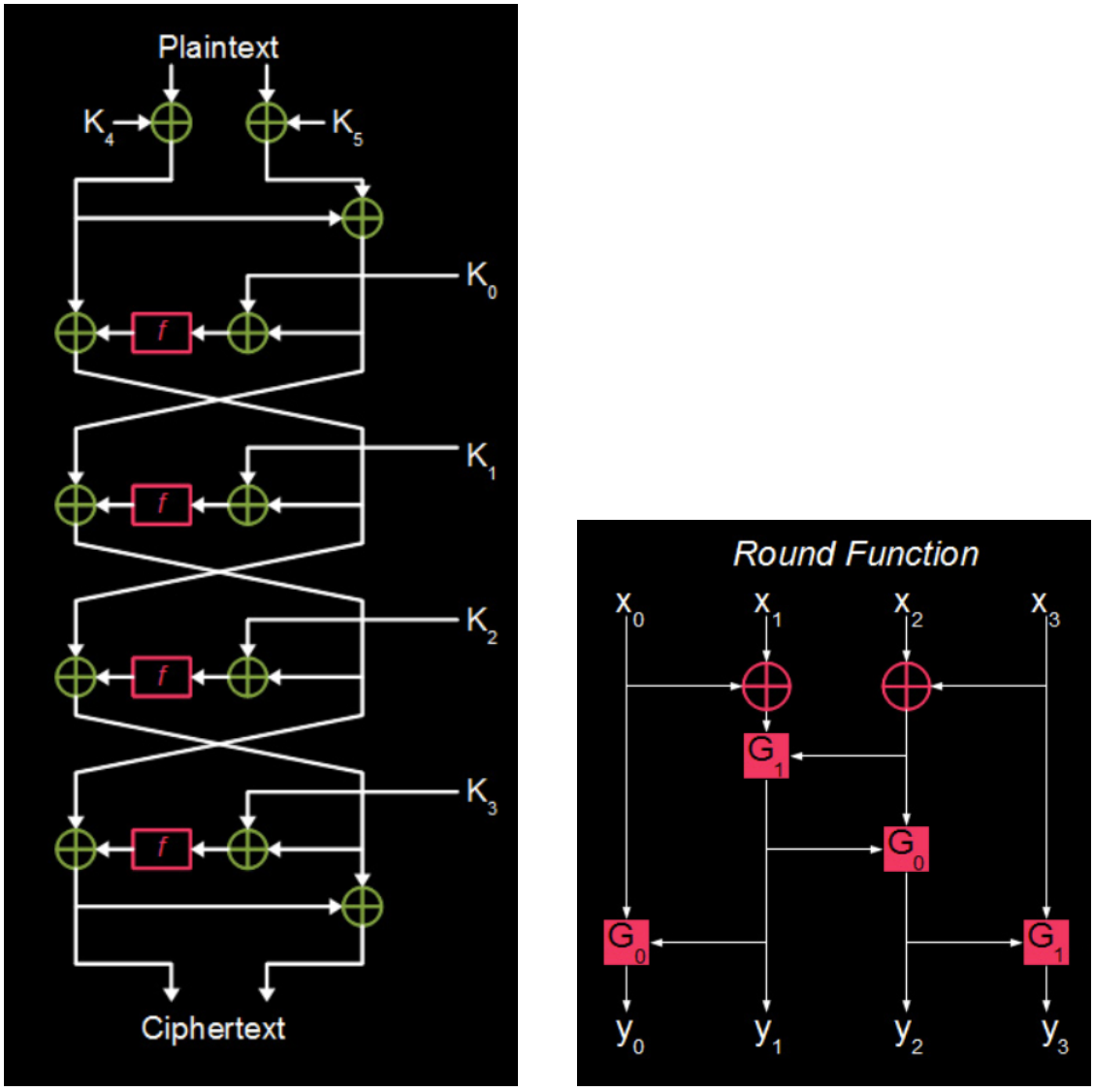
To prevent ease of differential cryptanalytic attacks, block ciphers should use a larger number of S-boxes (so that probabilities diminish on getting multiplied), and the S-box structure should be such that differences are not easily propagated from input to output.

4 Cryptanalysis of FEAL-4

As a case study, we consider the cryptanalysis of the Fast Data Encipherment Algorithm 4 (FEAL-4), consisting of a 4-layer Feistel network type design.

4.1 Structure of FEAL-4

The design of FEAL-4 is given below.



(a) FEAL-4 Encryption Structure

(b) Internal Structure of f-Boxes

Figure 1: Structure of FEAL-4

As can be seen, the scheme consists of 6 subkeys of size 32 bits each, and 4 rounds, with 64-bit input plaintext and 64-bit output ciphertext. The smaller f-boxes (round functions) have an internal structure as shown in the figure to the right. The functions G_i are defined as:

$$G_i(a, b) = ((a + b + i) \ \% \ 256) << 2$$

for $i = 0$ and $i = 1$.

4.2 Differential Cryptanalytic Attack

We now describe a differential cryptanalytic attack on the FEAL-4 encryption scheme.

Firstly, note that the difference of two bitstrings propagates through the linear parts of the encryption system similarly to the actual bitstrings themselves. For example:

$$\text{Diff}(a_1 \oplus b_1, a_2 \oplus b_2) = a_1 \oplus b_1 \oplus a_2 \oplus b_2 = \text{Diff}(a_1, a_2) \oplus \text{Diff}(b_1, b_2)$$

This shows that the difference in XORs is equal to XOR of differences, implying that differences can indeed be propagated directly through XOR operations. Also, it is clear that rotating bitstrings by 2 places to the left before XORing them is equivalent to XORing them first and then rotating by 2 places to the left, which implies that differences can be propagated directly through the G-Boxes within the f-Boxes.

As explained already, we first need to find a differential pair (input difference and output difference) for just one individual S-box, or in this case, f-Box.

Consider a pair of inputs having their difference as $\Delta x_0 = \Delta x_1 =$ 0x80 and $\Delta x_2 = \Delta x_3 =$ 0x00. We claim that, with a probability of 1, this is going to produce an output difference having $\Delta y_0 =$ 0x02 and $\Delta y_1 = \Delta y_2 = \Delta y_3 =$ 0x00. This can be proved by tracing the differences through the f-Box, as done in the following diagram:

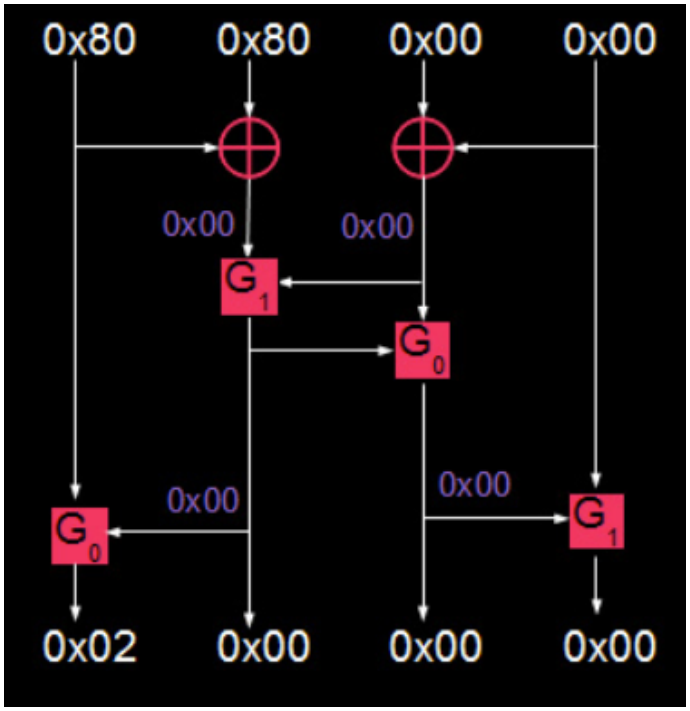


Figure 2: Difference Propagation through an f-Box

Since all f-Boxes are identical in structure and function, this holds true over all of them with probability equal to 1. This is extended to obtain a differential characteristic for the overall scheme as shown below. Also note that it can be shown that an input difference of 0x00000000 results in an output difference of 0x00000000 (with probability 1), which is another differential relationship for the individual f-Box, and is also used to construct the overall differential characteristic.

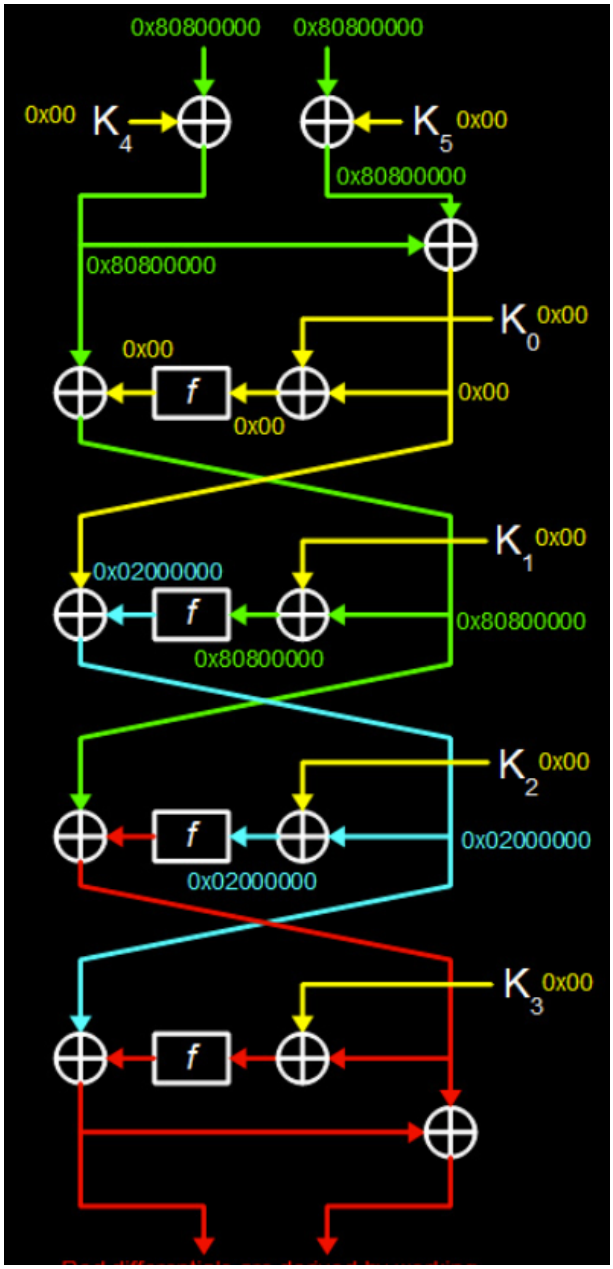


Figure 3: Differential Characteristic for FEAL-4

All values in the figure are differences, which is why every subkey has 0x00 (the same key is applied for both bitstrings, so their difference is zero). Note that the red marked arrows have differences that aren't determined by the input plaintext difference, and are thus unknown.

As can be seen from the figure, an input plaintext difference of 0x8080000080800000 results in a useful differential characteristic for the first 3 layers. Using this, we extract subkey K_3 used in the fourth layer. (Note that the probability of this differential characteristic is 1, as each step in it has probability 1.)

Using the ciphertexts (known), the values of the bitstrings which were XOR-ed with K_3 in the final round can be calculated (as left-ciphertext XOR right-ciphertext). We now do a linear search over the key space

(32-bit strings) of K_3, and for every key value, we XOR it with the bitstrings calculated just a step ago. Then the results are passed through the f-Box, to get encryptions on the other side. These are XORed with the left-ciphertext, and then the differences of appropriate pairs are checked to be 0x02000000, which was the differential as per our characteristic. The key value that gives a difference of 0x02000000 at this step for every plaintext-ciphertext pair is chosen as the estimate for subkey K_3.

Thus, K_3 is extracted. Using K_3, the ciphertexts are further back-propagated, up to the third round of the scheme. This time we use an input plaintext difference of 0x0000000080800000, so that the differential characteristic only spreads over two rounds. (Note that this plaintext difference is simply the bitstring difference after the first round of the 3-round differential characteristic; thus it is easy to reduce higher-round differential characteristics to lower ones.) The same thing as before is now repeated at the third round, and K_2 is extracted.

Again, the bitstrings are decrypted through K_2, and using a plaintext difference of 0x0000000002000000, a key search is done at the second round. Thus, K_1 is extracted.

Cracking the first round is slightly more difficult, due to the presence of three subkeys, namely K_0, K_4, K_5. The strategy adopted here is a key space search for K_0. For every possible value of K_0, the bitstrings are decrypted through K_0, and K_4 and K_5 are calculated using these decrypted bitstrings as well as the input plaintext. The subkey values K_0 which produce the same K_4 and K_5 for every chosen plaintext pair is determined to be K_0, along with its corresponding K_4 and K_5.

Thus, FEAL-4 has been broken using differential cryptanalysis! Let us analyze the advantage of this method over brute force. A brute force attack would require a search over the entire key space of $6 \times 32 = 192$ bits, i.e., a search over $2^{192} \approx 10^{57}$ possible keys. However, with differential cryptanalysis, we only need to do individual key space searches for four 32-bit keys, i.e., each key space search is over $2^{32} = 4294967296$ keys, which sums up to a total of $4 \times 4294967296 = 17179869184 \approx 10^{10}$ keys.

Thus, differential cryptanalysis in this case is clearly a huge improvement over a brute force attack!

4.3 Implementation

We have implemented the attack described above, on the FEAL-4 cipher, in C++. The program is available in the following GitHub repository:
https://github.com/kartikey24/Cryptanalysis-of-FEAL

To run the code, simply compile and execute the C++ file, either without command line arguments (in which case 12 plaintexts will be used for each round of the attack), or with a command line argument equal to the number of plaintexts to be used in each round of the attack. The code generates 6 random 32-bit subkeys, and then attacks the resulting FEAL-4 cipher using the differential cryptanalytic attack explained above.

Following is a snapshot of the program output:

```
kartikey@kartikey-G3-3579:/mnt/1EBCF05CBCF0303F/Sem 4/CS406$ g++ -O2 AttackOnFEAL4.cpp
kartikey@kartikey-G3-3579:/mnt/1EBCF05CBCF0303F/Sem 4/CS406$ ./a.out 12
Differential Cryptanalysis of FEAL-4


Round 4: To find K3

Generating 12 plaintext-ciphertext pairs
Using input differential 0x8080000080800000
  Using output differential of 0x2000000
  Cracking...
found key : 0x6b063b3f
  Time to crack round #4 = 11 seconds

Round 3: To find K2
Generating 12 plaintext-ciphertext pairs
Using input differential 0x80800000
  Using output differential of 0x2000000
  Cracking...
found key : 0x199a441b
  Time to crack round #3 = 2 seconds

Round 2: To find K1
Generating 12 plaintext-ciphertext pairs
Using input differential 0x2000000
  Using output differential of 0x2000000
  Cracking...
found key : 0xd7938c3
  Time to crack round #2 = 2 seconds

Round 1: To find K0
Cracking ...
found key K0: 0x4c36319a
found key K4: 0xb7d57594
found key K5: 0xa0b50e94
Total time taken = 24 seconds



Generating 12 plaintext-ciphertext pairs
Using input differential 0x123fec3c243ba9b2
Each ciphertext created using Keys obtained above matches ciphertext generated by encryption a
lgorithm
Finished successfully
```

Figure 4: Sample Output of Attack Implementation

4.4 Linear Cryptanalytic Attack

Matsui and Yagamishi (1992) devised a linear cryptanalytic attack on the FEAL-4 block cipher which requires only 5 known plaintexts to extract the encryption key. They worked with a slightly differently structured FEAL-4 cipher, which is shown below. Their paper invented linear cryptanalysis, applying it first to FEAL cipher family.

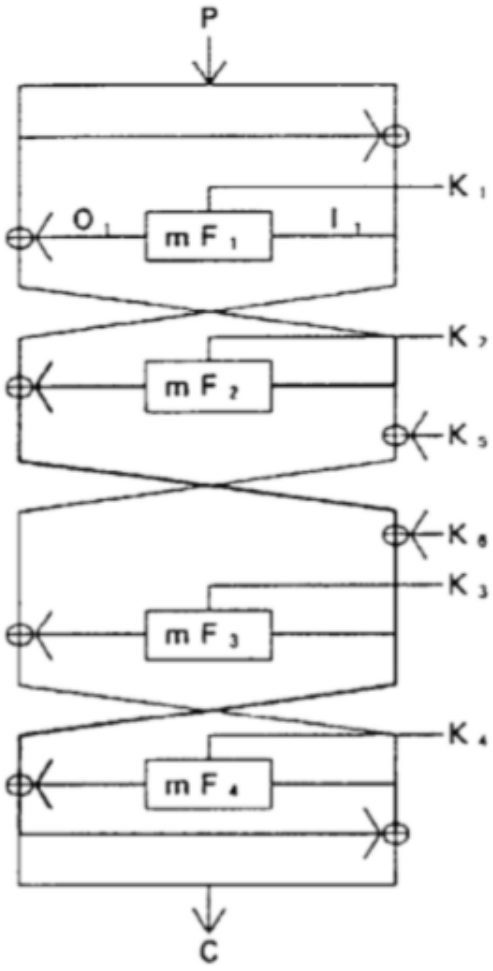
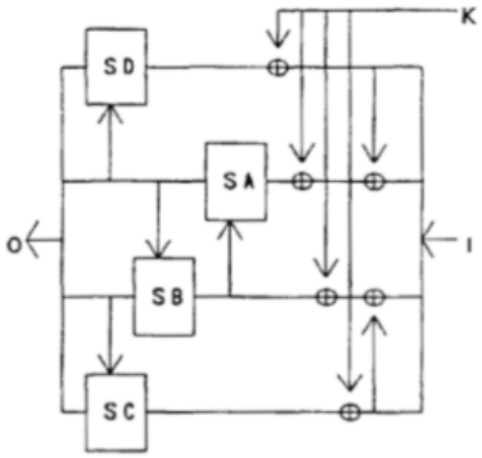


Figure 3: FEAL-4 cipher

(a) FEAL-4 Encryption Structure



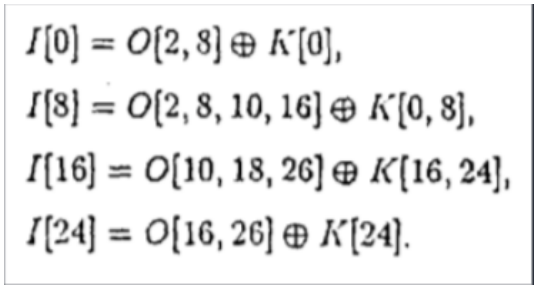
SA(x, y) = SC(x, y) = ROL2(x+y+1 (mod256))
SB(x, y) = SD(x, y) = ROL2(x+y (mod256))

Figure 1: Modified F function

(b) Internal Structure of f-Boxes

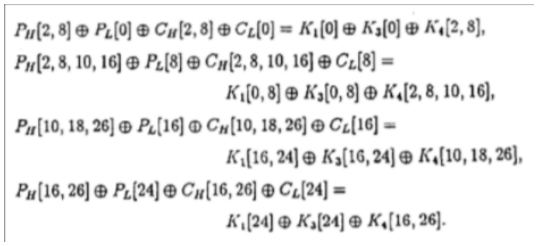
Figure 5: Structure of FEAL-4

As can be seen from the diagrams, this structure differs slightly from the previous one in the positioning of keys between the layers, and in the f-Box structure too, as key bits are XORed and manipulated inside, instead of XORing before passing into the f-Box as we saw previously.



$$I[0] = O[2,8] \oplus K[0],$$
$$I[8] = O[2,8,10,16] \oplus K[0,8],$$
$$I[16] = O[10,18,26] \oplus K[16,24],$$
$$I[24] = O[16,26] \oplus K[24].$$

(a) F-Box Linear Relations



$$P_H[2,8] \oplus P_L[0] \oplus C_H[2,8] \oplus C_L[0] = K_1[0] \oplus K_3[0] \oplus K_4[2,8],$$
$$P_H[2,8,10,16] \oplus P_L[8] \oplus C_H[2,8,10,16] \oplus C_L[8] = K_1[0,8] \oplus K_3[0,8] \oplus K_4[2,8,10,16],$$
$$P_H[10,18,26] \oplus P_L[16] \oplus C_H[10,18,26] \oplus C_L[16] = K_1[16,24] \oplus K_3[16,24] \oplus K_4[10,18,26],$$
$$P_H[16,26] \oplus P_L[24] \oplus C_H[16,26] \oplus C_L[24] = K_1[24] \oplus K_3[24] \oplus K_4[16,26].$$

(b) Three Layer Linear Relations

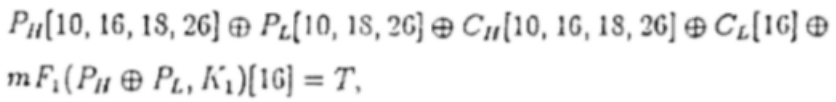
Figure 6: Linear Relations in FEAL-4

Using the above structure, through analysis, Matsui and Yagamishi obtained linear relations shown in Figure 5. The left collection shows linear approximations that hold over an individual f-Box, whereas the right shows linear approximations that hold over a 3-round FEAL cipher, obtained by extending the individual f-Box relations and combining them.

Note that $X[i_1, i_2, \ldots, i_k]$ denotes $X[i_1] \oplus X[i_2] \oplus \cdots \oplus X[i_k]$.

The linear approximations from above can be used to approximate three rounds of the FEAL-4 cipher, and thus obtain the keys used in the remaining round through an exhaustive linear search over the space of key bits which actually affect the relation.

The key-bit values which satisfy the linear approximation for the maximum number of plaintexts are chosen as the extracted key bits. For example, key bits 0-5 and 8-29 of FEAL-4 can be determined by this technique. For the remaining 4 bits, a linear search over the key space is sufficient as there are only a few (16) options to choose from.



$$P_H[10,16,18,26] \oplus P_L[10,18,26] \oplus C_H[10,16,18,26] \oplus C_L[16] \oplus mF_1(P_H \oplus P_L, K_1)[16] = T,$$

Figure 7: Linear relation over rounds 2-4 of FEAL-4

5 Interpolation Cryptanalytic Attacks

Another type of cryptanalytic attacks is interpolation attacks. These are applied to ciphers whose round functions can be expressed as simple algebraic functions.

Suppose the S-boxes involved can be expressed as polynomial functions. Then, this functionality can be extended over the entire encryption scheme, so as to represent the entire system as a polynomial function having key values as coefficients.

Using known plaintext-ciphertext pairs along with Lagrange interpolation, the coefficients can be computed, and thus the keys can be extracted. This is based on Lagrange interpolation:

Definition 1. *Let K be a field. Given $2n$ elements $x_1, ..., x_n, y_1, ..., y_n \in R$, where the x_i's are distinct, define:*

$$f(x) = \sum_{i=1}^{n} y_i \prod_{1 \le j \le n, j \ne i} \frac{x - x_j}{x_i - x_j}$$

Then $f(x)$ is the only polynomial over K of degree at most $n - 1$ such that $f(x_i) = y_i$ for $i = 1 \dots n$. This equation is called ***Lagrange interpolation formula****.*

Figure 8: Lagrange Interpolation

This is a global deduction type of attack, as an equivalent expression for the cipher is constructed instead of extracting keys involved in rounds directly. However, this can easily be extended to a key recovery attack by approximating the first $r - 1$ rounds and guessing the last round key.

This can also be adapted to ciphers which are reducible to a rational expression.

6 References

1. Cryptanalysis of Block Ciphers: A Survey
2. A New Method for Known Plaintext Attack of FEAL Cipher
3. Differential Cryptanalysis of FEAL by Jon King
4. Wikipedia, the free encyclopedia