Differential Cryptanalysis of Feal and N-Hash

Eli Biham Adi Shamir

The Weizmann Institute of Science
Department of Applied Mathematics and Computer Science
Rehovot 76100, Israel

Abstract

In [1,2] we introduced the notion of differential cryptanalysis and described its application to DES[11] and several of its variants. In this paper we show the applicability of differential cryptanalysis to the Feal family of encryption algorithms and to the N-Hash hash function. In addition, we show how to transform differential cryptanalytic chosen plaintext attacks into known plaintext attacks.

1 Introduction

Feal is a family of encryption algorithms, which are designed to have simple and efficient software implementations on eight-bit microprocessors. The original member of this family, called Feal-4[13], had four rounds. This version was broken by Den Boer[3] using a chosen plaintext attack with 100 to 10000 ciphertexts.

The designers of Feal reacted by creating a second version, called Feal-8[12,9] in which the number of rounds was increased to eight, while the F function was not changed.

Feal-8 was broken by the differential cryptanalytic chosen plaintext attack described in this paper. As a result, two new versions were added to the family: Feal-N[6] with any even number N of rounds, and Feal-NX[7] with an extended 128-bit key. In addition, The designers proposed a more complex eight-round version called N-Hash[8] as a cryptographically strong hash function which maps arbitrarily long inputs into 128-bit values.

Recently, two chosen plaintext attacks on Feal were published. The one analyses Feal-8 using 10000 encryptions[5]. This attack is partially derived from the attack described in this paper. The other analyses Feal-4 using 20 encryptions[10].

The main results reported in this paper are as follows: Feal-8 is breakable under a chosen plaintext attack with 2000 ciphertexts. Feal-N can be broken faster than via exhaustive search for any $N \leq 31$ rounds, and Feal-NX is just as easy to break as Feal-N for any value of N. The differential cryptanalytic chosen plaintext attacks can be transformed into known plaintext attacks which can be applied even in the CBC mode of operation, provided we have sufficiently many known plaintext/ciphertext pairs (about 2^{38} in the case of Feal-8). Variants of N-Hash with up to 12 rounds can be broken faster than via the birthday paradox, but for technical reasons we can apply this attack only when the number of rounds is divisible by three. Feal-4 is trivially breakable with eight chosen plaintexts or via a non-differential attack with about 100000 known plaintexts.

2 Differential Cryptanalysis of Feal

The notion of differential cryptanalysis and its application to DES-like cryptosystems are described in [1,2]. The basic tool of differential cryptanalytic attacks is a pair of ciphertexts whose corresponding plaintexts have a particular difference. The method analyses many pairs with the same difference, assigns probabilities to the different possible keys and locates the most probable key. For Feal the difference is chosen as a particular XORed value of the two plaintexts.

In this paper we use the notation introduced in [1,2] with additional Feal-specific notation:

n_x: An hexadecimal number is denoted by a subscript x (i.e., $10_x = 16$).

X^*, X': At any intermediate point during the encryption of pairs of messages, X and X^* are the corresponding intermediate values of the two executions of the algorithm, and X' is defined to be $X' = X \oplus X^*$.

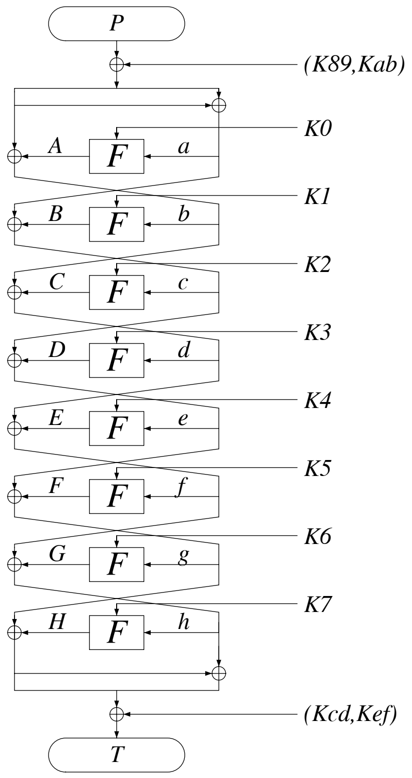
P, T: The plaintext and the ciphertext. Unlike in DES, they denote the real plaintext and ciphertext without ignoring the initial and final transformations. Thus, the characteristic's input XOR Ω_P is different from the corresponding plaintext XOR P'. Note that the definitions in [1,2] assume that P denotes the value after the initial transformation rather than the real plaintext.

(L, R): The left and right halves of the plaintext P are denoted by L and R respectively.

(l, r): The left and right halves of the ciphertext T are denoted by l and r respectively.

a, ..., h: The 32-bit inputs of the F function in the various rounds. See figure 1.

A, ..., H: The 32-bit outputs of the F function in the various rounds. See figure 1.



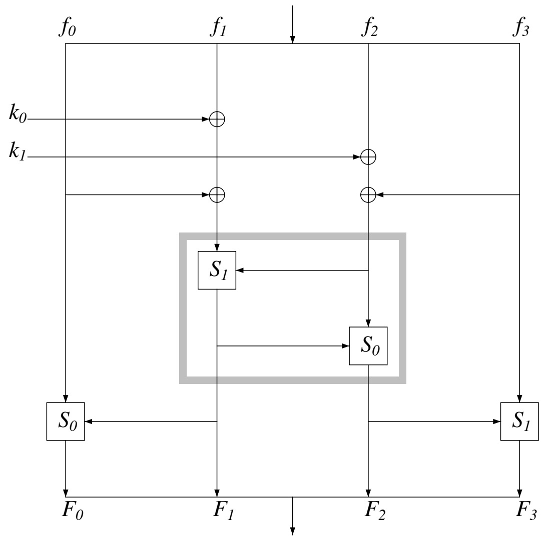


Figure 1. The outline of Feal-8 and the F function.

ROL$n(X)$, ROR$n(X)$: Rotation of the byte X by n bits to the left and to the right respectively.

$S_i(x, y)$: The Feal S boxes: $S_i(x, y) = \mathrm{ROL2}(x + y + i \pmod{256})$.

X_i: The i^{th} byte of the 16, 32 or 64-bit X or the i^{th} bit of the byte X.

$X_{i,j}$: The j^{th} bit of X_i (where 0 is the least significant bit).

am(K): The 32-bit value $(0, K_0, K_1, 0)$ where K is 16-bit long.

mx(X): The 16-bit value $(X_0 \oplus X_1, X_2 \oplus X_3)$ where X is 32-bit long.

$\oplus$: The exclusive-or operator.

The structure of Feal (see figure 1) is similar to the structure of DES with a new F function and modified initial and final transformations. The F function of Feal contains two new operations: byte rotation which is XOR-linear and byte addition which is not XOR-linear. The byte addition operation is the only non-linear operation in Feal and therefore the strength of Feal crucially depends on its non-linearity. At the beginning and at the end of the encryption process the right half of the data is XORed with the left half of the data and the whole data is XORed with additional subkeys, rather than permuted as in DES. Due to their linearity, these XORs pose only minor difficulty to our attack.

The addition operations in the S boxes are not XOR-linear. However, there is still a statistical relationship between the input XORs of pairs and their output XORs. A table which shows the distribution of the input XORs and the output XORs of an S box is called the *pairs XOR distribution table* of the S box. Such a table has an entry for each combination of input XOR and output XOR, and the value of an entry is the number of possible pairs with the corresponding input XOR and output XOR. Usually several output XORs are possible for each input XOR. A special case arises when the input XOR is zero, in which case the output XOR must be zero as well. We say that X *may cause* Y (denoted by $X \rightarrow Y$) if there is a pair in which the input XOR is X and the output XOR is Y. We say that X *may cause* Y *with probability* p if for a fraction p of the pairs with input XOR X, the output XOR is Y.

Since each S box has 16 input bits and only eight output bits it is not recommended to use the pairs XOR distribution tables directly. Instead, in the first stage of the analysis we use the joint distribution table of the two middle S boxes in the F function (inside the gray rectangle in figure 1). This combination has 16 input bits and 16 output bits, and the table has many interesting entries. For example, there are two entries with probability 1 which are $00\,00_x \rightarrow 00\,00_x$ and $80\,80_x \rightarrow 00\,02_x$. About 98% of the entries are impossible (contain value 0). The average value of all the entries is 1, but the average value of the possible entries is about 50. In appendix A we describe how we can easily decide if $X \rightarrow Y$ or not for given XOR values X and Y without consulting the table.

The S boxes also have the following properties with respect to pairs: Let $Z = S_i(X,Y)$. If $X' = 80_x$ and $Y' = 80_x$ then $Z' = 00_x$ always. If $X' = 80_x$ and $Y' = 00_x$ then $Z' = 02_x$ always. For any input XORs X' and Y' of the S boxes the resultant output XOR $Z' = \mathrm{ROL2}(X' \oplus Y')$ is obtained with probability about $\frac{1}{2^{\#(X'|Y')}}$ where $\#X$ is the number of bits set to 1 in the lower seven bits of the byte X and $|$ is the or operator. This happens because each bit which is different in the pairs (X and X^*, or Y and Y^*) gives rise to a different carry with probability close to $\frac{1}{2}$. If all the carries happen at the same bits in the pair then the equation is satisfied.

The input of the F function in the last round is a function of the ciphertext XORed with an additional subkey of the final transformation rather than just a function of the ciphertext (as in DES). There is an equivalent description of Feal in which the XOR with the subkeys in the final transformation is eliminated and the 16-bit subkeys XORed to the two middle bytes of the inputs of the F function in the various rounds are replaced by 32-bit values.

Definition 1 The 32-bit subkeys of the equivalent description in which the XOR with the subkeys in the final transformation is eliminated are called *actual subkeys.* The actual subkey which replaces the subkey Ki is denoted by AKi. The 16-bit XOR combinations $\mathrm{mx}(AKi) = (AKi_0 \oplus AKi_1, AKi_2 \oplus AKi_3)$ are called *16-bit actual* *subkeys*. The actual subkey of the last round of a cryptosystem is called the *last actual subkey*.

The actual subkeys in the even rounds $i+1$ are

$$AKi = Kcd \oplus Kef \oplus \mathrm{am}(Ki).$$

The actual subkeys in the odd rounds $i+1$ are

$$AKi = Kcd \oplus \mathrm{am}(Ki).$$

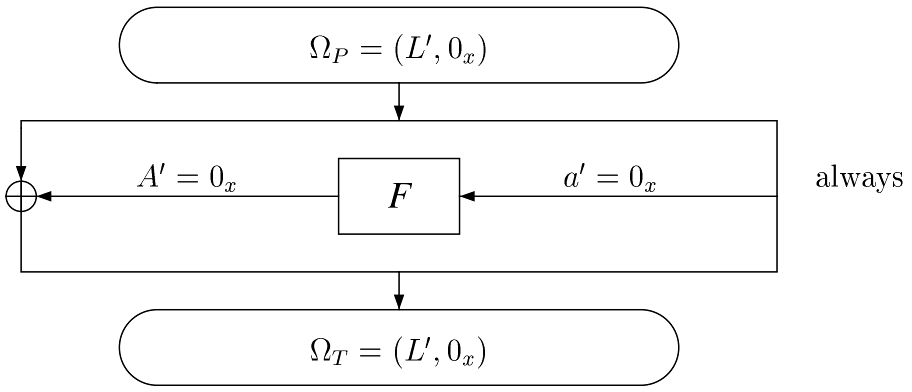
The actual subkeys of the initial transformation are

$$\begin{aligned} AK89 &= K89 \oplus Kcd \oplus Kef \\ AKab &= Kab \oplus Kef. \end{aligned}$$

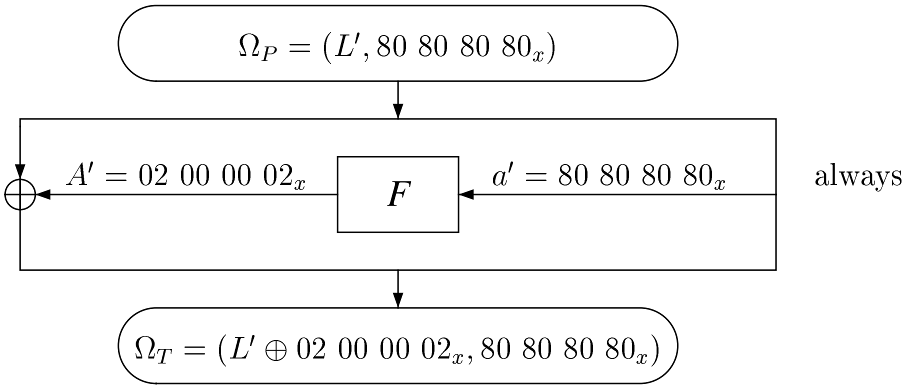
The actual subkeys of the final transformation are eliminated and thus their equivalent values are zero. Our attack finds the actual subkeys rather than the subkeys themselves since it finds XORs of the ciphertexts and internal values in the F function.

A tool which pushes the knowledge of the XORs of pairs as many rounds as possible is called a *characteristic*. An *n-round characteristic* Ω starts with an input XOR value Ω_P and assigns a probability in which the data XOR after n rounds becomes Ω_T. Two characteristics Ω^1 and Ω^2 can be concatenated to form a longer characteristic whenever Ω^1_T equals the swapped value of the two halves of Ω^2_P, and the probability of Ω is the product of the probabilities of Ω^1 and Ω^2. A pair whose intermediate XORs equal the values specified by a characteristic is called a *right pair* with respect to the characteristic. Any other pair is called a *wrong pair* with respect to the characteristic. Note that in Feal, the plaintext XOR P' is different from the input XOR of the characteristic Ω_P due to the initial and final transformations.

The simplest example of a one-round characteristic with probability 1 is:

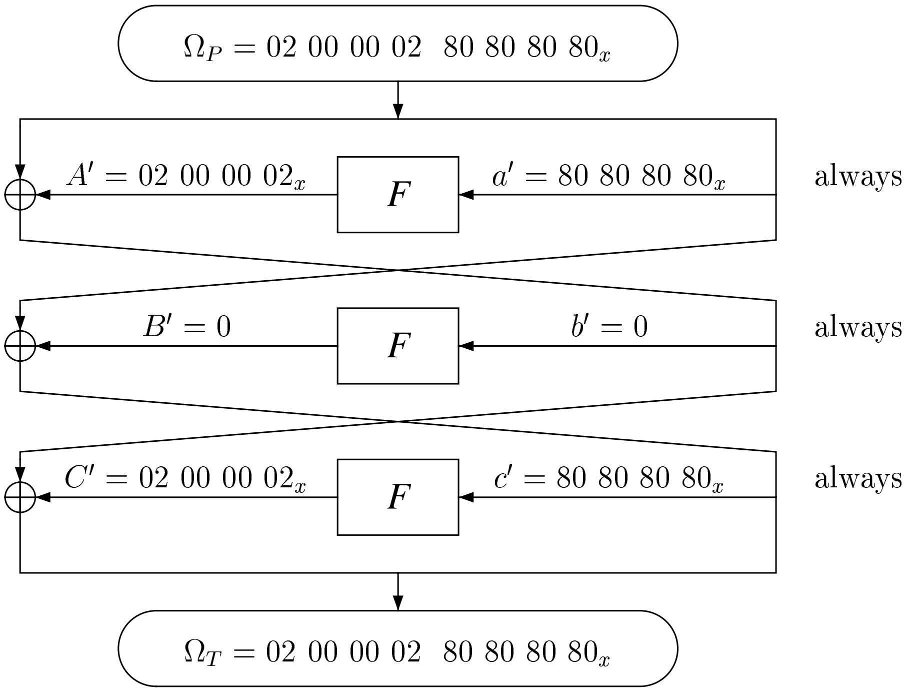


This characteristic is similar to the one-round characteristic with probability 1 of DES. Unlike the case of DES, Feal has three other one-round characteristics with probability 1. A typical one is:

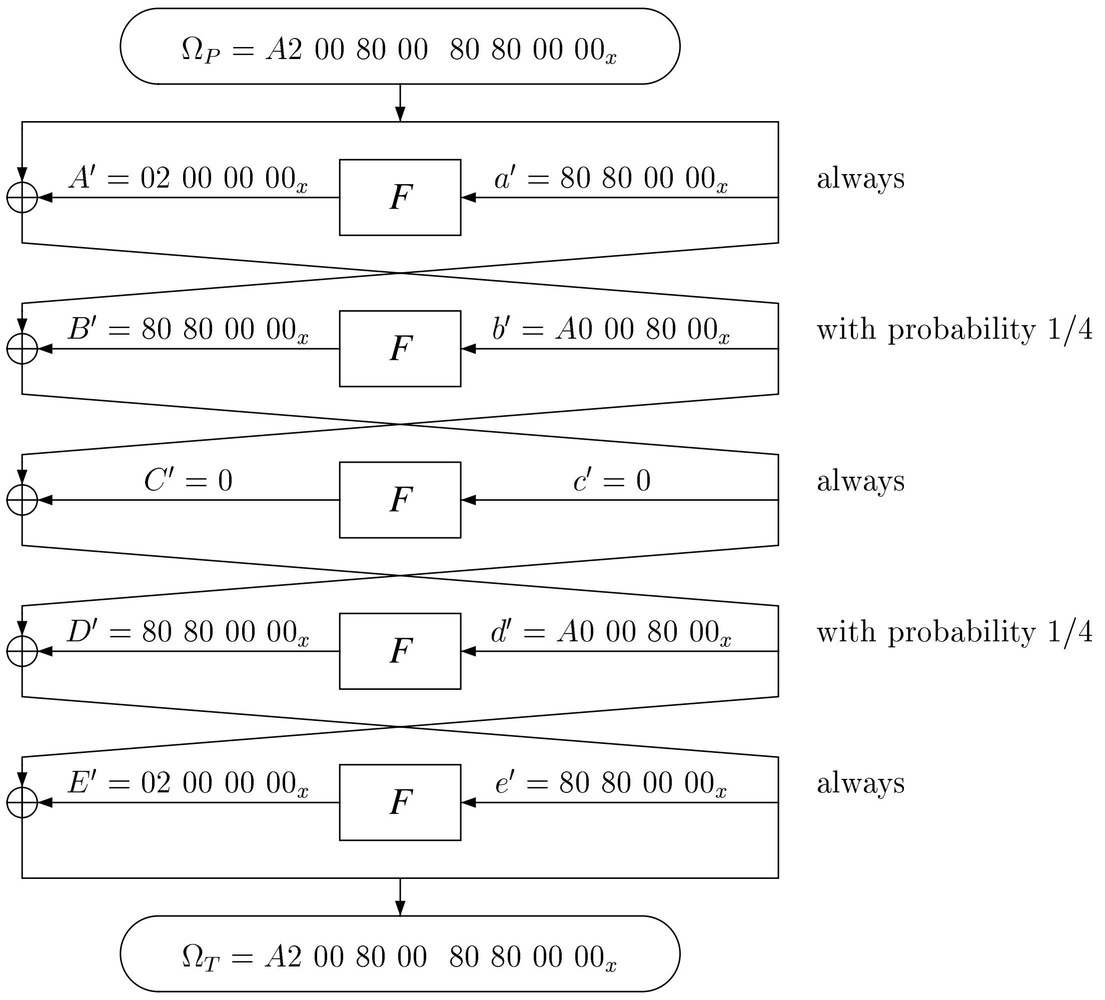


$\Omega_P = (L', 80\ 80\ 80\ 80_x)$
$A' = 02\ 00\ 00\ 02_x$
F
$a' = 80\ 80\ 80\ 80_x$
always
$\Omega_T = (L' \oplus 02\ 00\ 00\ 02_x, 80\ 80\ 80\ 80_x)$

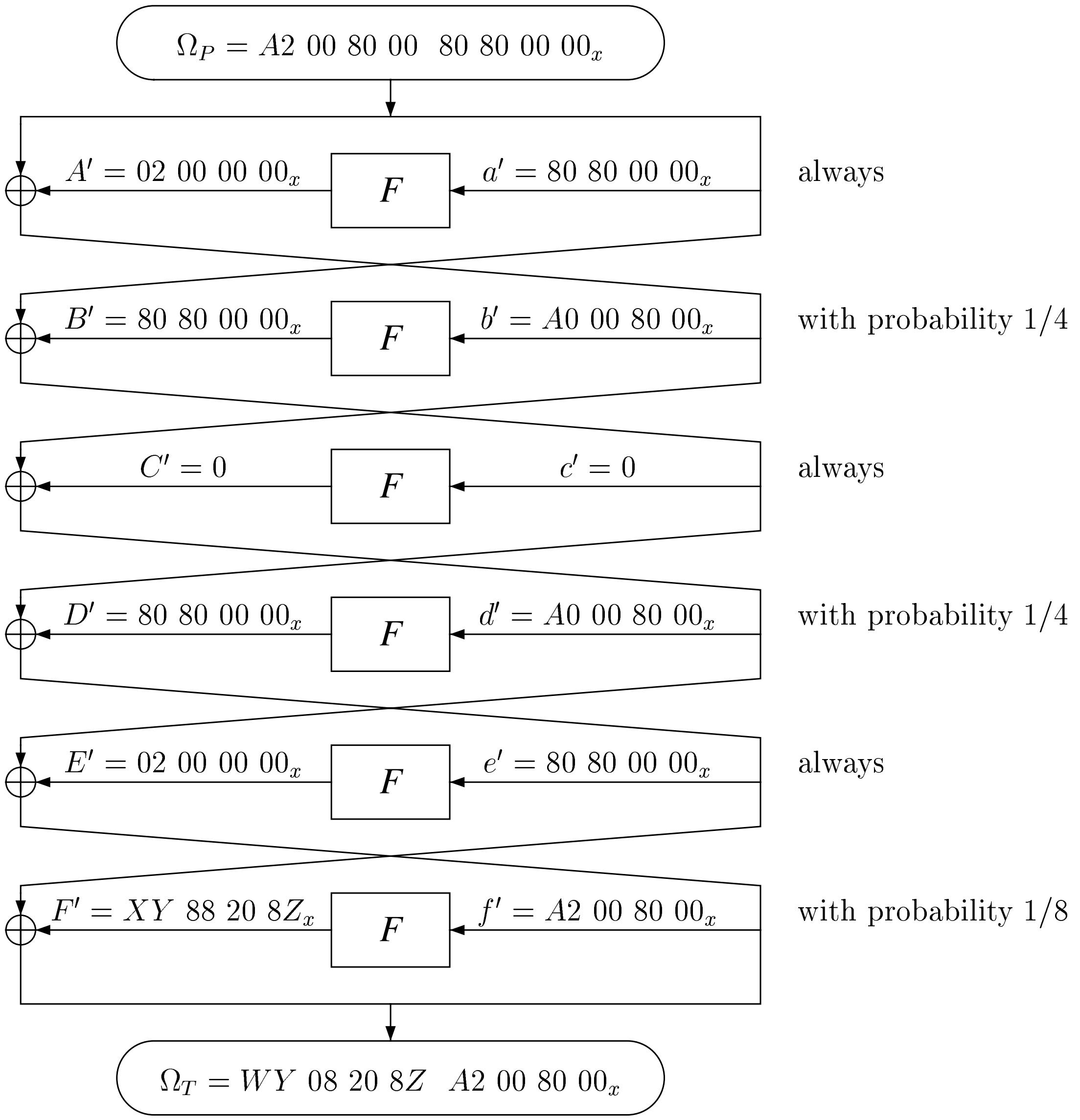
Three non-trivial three-round characteristics with probability 1 also exist. The one derived from the above one-round characteristic is:



The following is a five-round characteristic with probability $\frac{1}{16}$:

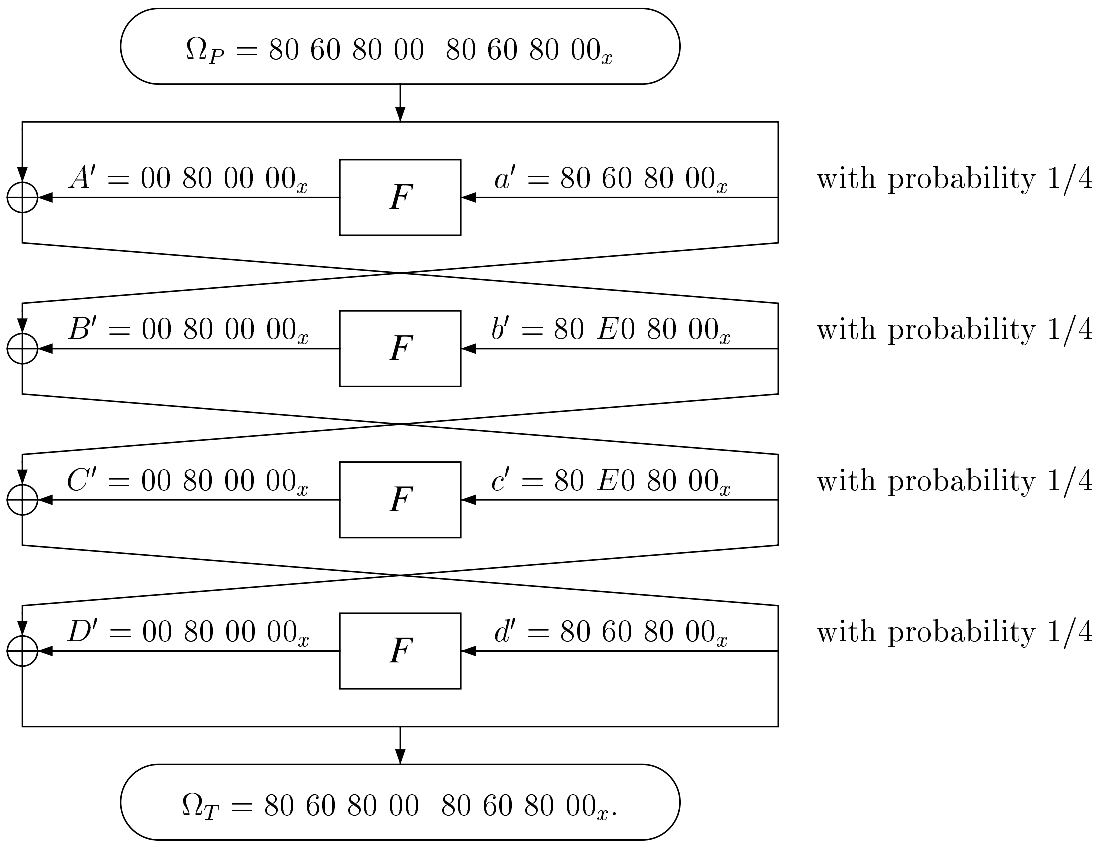


This five-round characteristic can be extended to a six-round characteristic with probability $\frac{1}{128}$, for which not all the bit differences at the left half of the data after the sixth round are fixed:



where the values of X, Y, Z and W can range (for different right pairs) over $X \in \{5, 6, 7, 9, A, B, D, E, F\}$, $Y \in \{9, A, B\}$, $Z \in \{0, 1, 3\}$ and $W = X \oplus 8$. There is another five-round characteristic with probability $\frac{1}{16}$ which has a similar extension to six rounds.

Among the most useful characteristics are those that can be iterated. A characteristic Ω is called an *iterative characteristic* if the swapped value of the two halves of Ω_P equals Ω_T. The iterative characteristics of Feal do not include one in which a non-zero input XOR of the F function may cause a zero output XOR since the F function is reversible, but there are other kinds of iterative characteristics. The following is an iterative characteristic which has probability $\frac{1}{4}$ for each round:



Given a sufficiently long characteristic and a right pair we can calculate the output XOR of F function in the last round. The inputs themselves of this F function are known from the ciphertexts up to a XOR with subkeys. For any possible value of the last actual subkey, we count the number of possible pairs for which the output XOR is as expected. Every right pair suggests the right value of the actual subkey. The wrong pairs suggest random values. Since the right pairs occur with the characteristic's probability, the right value of the actual subkey should be counted more often than any other value. Therefore, it can be identified.

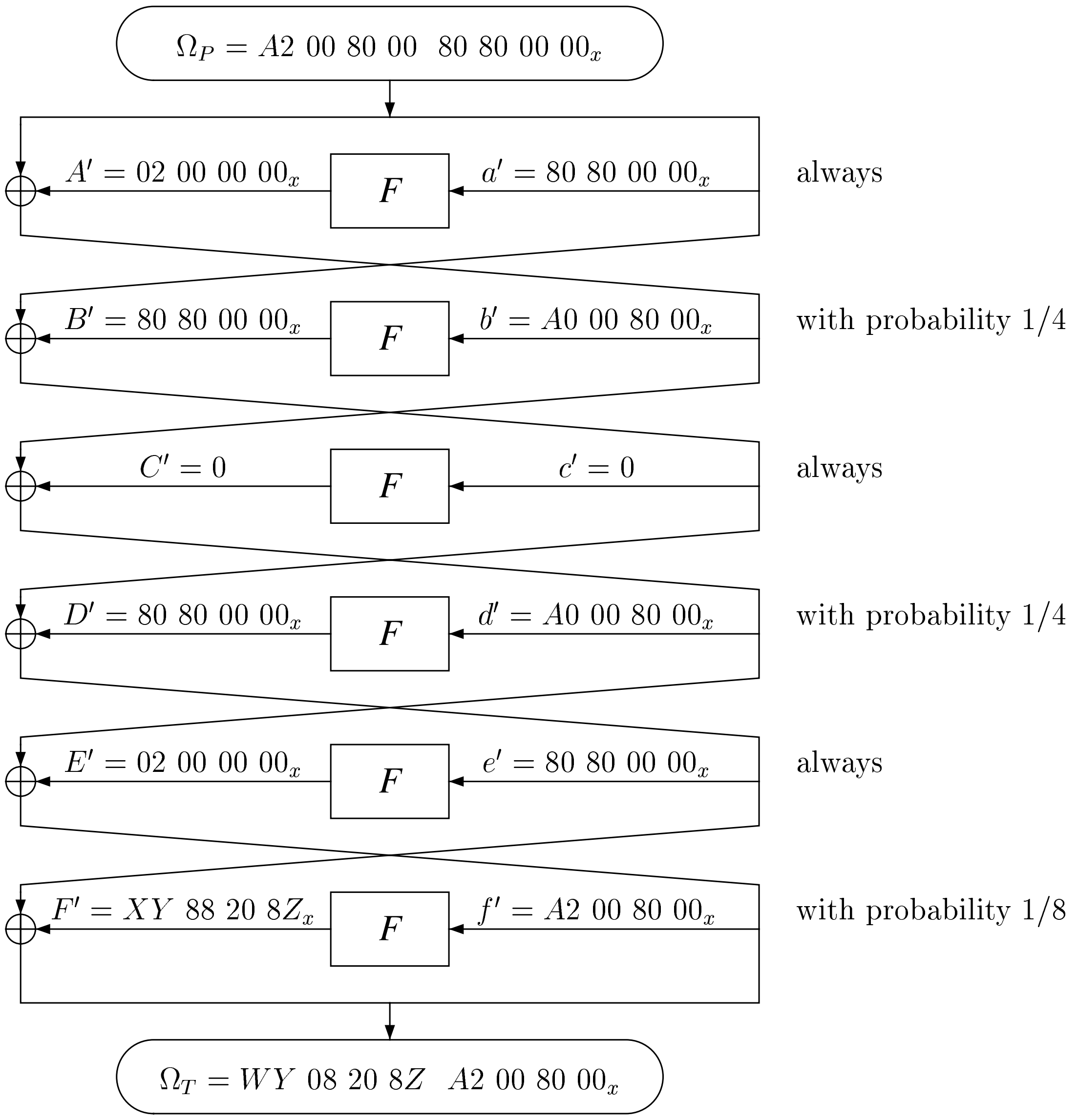
The number of pairs needed for a differential cryptanalytic attack depends on the characteristic's probability, on the number of subkey bits counted and on the level of identification of the right key. The ratio between the number of right pairs and the average count in a counting scheme is called the *signal to noise ratio of the counting scheme* and is denoted by S/N. The signal to noise ratio of a counting scheme is

$$S/N = \frac{2^k \cdot p}{\alpha \cdot \beta}$$

where k is the number of subkey bits which are counted in 2^k counters, p is the characteristic's probability, α is the average count per counted pair and β is the fraction of the counted pairs among all the pairs. The value of the signal to noise ratio indicates how many right pairs are needed to the attack and thus the total number of pairs needed. If the signal to noise ratio of a counting scheme is high only few pairs are needed. If the signal to noise ratio is low many right pairs are needed. If the signal to noise ratio is too low the attack may become impractical.

3 Cryptanalysis of Feal-8

This differential cryptanalytic chosen plaintext attack on Feal-8 uses about 1000 pairs of ciphertexts whose corresponding plaintexts are chosen at random satisfying $P' = A2\ 00\ 80\ 00\ 22\ 80\ 80\ 00_x$. This plaintext XOR is motivated by the following six-round characteristic whose probability is 1/128, for which not all the bits of Ω_T are fixed:



where the values of X, Y, Z and W can range (for different right pairs) over $X \in \{5, 6, 7, 9, A, B, D, E, F\}$, $Y \in \{9, A, B\}$, $Z \in \{0, 1, 3\}$ and $W = X \oplus 8$.

Five shorter characteristics are derived from the first rounds of this six-round characteristic. Each characteristic has a different number of rounds but all of them have the same value of Ω_P. The one-round characteristic which is derived from the first round of the six-round characteristic has probability 1. The two-round characteristic which is derived from the first two rounds has probability 1/4. The three-round characteristic also has probability 1/4. The four-round and the five-round characteristics have probability 1/16.

3.1 Reducing Feal-8 to seven rounds

In order to find the last actual subkey we do the following. Given the ciphertexts T and T^* of a right pair, we can deduce:

$$\begin{aligned} g' &= WY\ 08\ 20\ 8Z_x \\ h' &= l' \oplus r' \\ G' &= f' \oplus h' = A2\ 00\ 80\ 00_x \oplus l' \oplus r' \\ H' &= l' \oplus g' = l' \oplus WY\ 08\ 20\ 8Z_x. \end{aligned}$$

Therefore, all the bits of h' and G' and 24 bits of each of g' and H' are known.

The counting method is used to find the 16-bit last actual subkey. Filtering can be done by the knowledge of bits in the other two bytes of H' and in the seventh round. Assuming $g' \to G'$ we can reverse calculate the values of $g'_{i,0}$ from G' by

$$\begin{aligned} g'_{0,0} &= G'_{0,2} \oplus G'_{1,0} \\ g'_{3,0} &= G'_{3,2} \oplus G'_{2,0} \\ g'_{2,0} &= G'_{2,2} \oplus G'_{1,0} \oplus g'_{3,0} \\ g'_{1,0} &= G'_{1,2} \oplus g'_{0,0} \oplus g'_{2,0} \oplus g'_{3,0} \end{aligned}$$

and verify that the two known bits $g'_{1,0}$ and $g'_{2,0}$ from the characteristic are the same. About $\frac{3}{4}$ of the wrong pairs are discarded by this verification. We can also discard about $\frac{4}{5}$ of the other wrong pairs for which $g' \not\to G'$. Assuming $h' \to H'$ we can verify the four bits of $H'_{i,2}$ by

$$\begin{aligned} H'_{0,2} &= H'_{1,0} \oplus h'_{0,0} \\ H'_{1,2} &= h'_{0,0} \oplus h'_{1,0} \oplus h'_{2,0} \oplus h'_{3,0} \\ H'_{2,2} &= H'_{1,0} \oplus h'_{2,0} \oplus h'_{3,0} \\ H'_{3,2} &= H'_{2,0} \oplus h'_{3,0}. \end{aligned}$$

This verification discards about $\frac{15}{16}$ of the remaining wrong pairs.

All the right pairs must be verified correctly. Only about $\frac{1}{4}\cdot\frac{1}{5}\cdot\frac{1}{16}=\frac{1}{320}$ of the wrong pairs should pass the three filters. Since the right pairs occur with the characteristic's probability of $\frac{1}{128}$, most of the remaining pairs are right pairs.

The counting scheme counts the number of pairs for which each value of the 16-bit last actual subkey mx($AK7$) is possible. The expected signal to noise ratio is

$$S/N=\frac{2^{16}\cdot 2^{-7}}{\frac{1}{4}\cdot\frac{1}{5}\cdot\frac{1}{4}\cdot 1}\approx 2^{15}.$$

This ratio is so high that only eight right pairs are typically needed for the attack, and thus the total number of pairs we have to examine is about $8\cdot 128\approx 1000$. Note that we cannot distinguish between the right value of the 16-bit actual subkey and the same value XORed with 80 80_x. Therefore, we find two possibilities for the 16-bit last actual subkey.

The following counting scheme is used to complete the last actual subkey. For this counting scheme the five-round characteristic with probability 1/16 suffices. For each pair (out of all the pairs) we calculate $\hat{H}$ and $\hat{H}^*$ and get $\hat{H}'$ where for any 32-bit X, $\hat{X}$ is the 16-bit value of its two middle bytes (i.e., (X_1,X_2)). Then we calculate $\hat{g}'=\hat{l}'\oplus\hat{H}'$, $\hat{F}'=\hat{e}'\oplus\hat{g}'$ and few other bits of g' and discard any pair for which we can conclude that $g'\not\rightarrow G'$ by the F function using the bits we have found.

We try the 128 possibilities for the lowest seven bits of $AK7_0$. For each value we calculate H_0, H_0^*, $H_0'=H_0\oplus H_0^*$ and $F_0'=e_0'\oplus H_0'\oplus l_0'$ and verify that f_0' (from the characteristic) and F_1' (from $\hat{F}'$) may cause this F_0'. We count the number of the pairs satisfying this condition. The value of $AK7_0$ which is counted most often is likely to be the right value. We cannot distinguish the upper bit of the value, so we try just 128 possibilities (instead of 256 as was expected) and then try the two possible values in the following steps, till the wrong one fails. In a similar way we find seven bits of $AK7_3$. As a result, we find eight possibilities for $AK7$ and we can reduce the cryptosystem to a seven-round cryptosystem.

3.2 Reducing the seven-round cryptosystem to six rounds

We assume that the last actual subkey is already known, so the cryptosystem can be reduced to a seven-round cryptosystem. A right pair with respect to the five-round characteristic with probability 1/16 satisfies

$$\begin{aligned}
f' &= A2\ 00\ 80\ 00_x\\
g' &= l'\oplus H'\\
G' &= h'\oplus f' = h'\oplus A2\ 00\ 80\ 00_x\\
F' &= e'\oplus g' = l'\oplus H'\oplus 80\ 80\ 00\ 00_x.
\end{aligned}$$

We verify that $f'\rightarrow F'$ and $g'\rightarrow G'$ and count in two steps: the first step counts on the 16-bit actual subkey and the second step counts on each one of the other two bytes. The signal to noise ratio of the first step which finds the 16-bit actual subkey mx($AK6$) is

$$S/N = \frac{2^{16}}{16 \cdot \left(\frac{1}{7}\right)^4 \cdot \left(\frac{1}{7}\right)^2 \cdot 1} \approx 2^{29}.$$

The signal to noise ratio of the second step which finds $AK6_0$ and $AK6_3$ is

$$S/N = \frac{2^{8}}{16 \cdot \left(\frac{1}{7}\right)^4 \cdot 2^{-16} \cdot 1} \approx 2^{31}.$$

In the first step one bit is indistinguishable and in the second step two bits are indistinguishable. Therefore, we try all the eight possibilities of $AK6$ in parallel in the following steps.

In total we find at most 64 possibilities for the last two actual subkeys and can thus reduce the cryptosystem to six rounds.

3.3 Reducing the cryptosystem to 5, 4, 3, 2 and 1 rounds

Using the last two actual subkeys we can calculate H and G for any ciphertext T and reduce the cryptosystem to six rounds. All the right pairs with respect to the five-round characteristic satisfy $f' = h' \oplus G' = A2\ 00\ 80\ 00_x$ and $f' \rightarrow g' \oplus 80\ 80\ 00\ 00_x$ (g' can be calculated using the known $AK7$). Two bytes of $AK5$ equal their counterparts in $AK7$ and only $AK5_1$ and $AK5_2$ are different. We try all the 2^{16} possibilities of these two bytes. For each possibility and each pair we calculate F, F^* and $F' = F \oplus F^*$. A right pair satisfies $F' = g' \oplus 80\ 80\ 00\ 00_x$. We count the number of pairs whose $f' = A2\ 00\ 80\ 00_x$ (as is enforced by the five-round characteristic) and whose above values of F' are equal. The value of $AK5$ which is counted more often than any other is likely to be the real value. The signal to noise ratio of this step is

$$S/N = \frac{2^{16}}{16 \cdot 2^{-32} \cdot 2^{-16}} = 2^{60}.$$

In this step we can always distinguish all the bits using less than 1000 pairs.

Given $AK5$ we reduce the cryptosystem to five rounds and find $AK4$ using the three-round characteristic. For each possible value of $AK4$ we count the number of pairs which satisfy $e' = g' \oplus F' \neq 80\ 80\ 00\ 00_x$ (the pairs whose $e' = 80\ 80\ 00\ 00_x$ are useless because it enforces a fixed output XOR), $e' \rightarrow E'$ and $d' \rightarrow D' = g' \oplus F'$. $AK3$ is calculated similarly by counting the pairs which satisfy $d' = A0\ 00\ 80\ 00_x$ and $d' \rightarrow D'$. $AK2$ is calculated similarly using the one-round characteristic and counting the pairs which satisfy $c' \neq 0$, $c' \rightarrow C'$ and $b' \rightarrow B'$. $AK1$ is calculated similarly by counting the pairs which satisfy $b' \rightarrow B'$.

$AK0$ cannot be calculated by this characteristic and plaintext XOR because $A' = 02\ 00\ 00\ 00_x$ always and thus all the possibilities succeed under the A' condition with equal distribution. However, it can be found using other characteristics. The actual subkeys of the initial transformation $AK89$ and $AKab$ cannot be found without the value of a plaintext even if all the other actual subkeys are known. In our case $AK0$, $AK89$ and $AKab$ are not needed since the key itself can be easily obtained from the actual subkeys we already found.

Although we find seven actual subkeys with the (true) assumption that many actual subkeys have the same values in their first bytes, and the same values in their last bytes, it is possible to extend this attack to the general case where all the actual subkeys are independent (i.e., $8 \cdot 32 + 2 \cdot 32 = 320$ independent bits).

3.4 Calculating the key itself

Using the values of the actual subkeys $AK1$–$AK7$ the following XORs of the original subkeys can be obtained:

$$\begin{gathered} K5 \oplus K7 \\ K4 \oplus K6 \\ K3 \oplus K5 \\ K2 \oplus K4 \\ K1 \oplus K3. \end{gathered} \tag{1}$$

We can easily derive the key itself by analyzing the structure of the key processing algorithm using these values.

We try all the 256 possible values of $K5_1$. For each value we calculate [the values in brackets are known from (1)]:

$$\begin{aligned} K7_1 &= K5_1 \oplus [K5_1 \oplus K7_1] \\ K3_1 &= K5_1 \oplus [K3_1 \oplus K5_1] \\ K1_1 &= K3_1 \oplus [K1_1 \oplus K3_1]. \end{aligned}$$

By the fourth round of the key processing:

$$\begin{aligned} K7_0 &= K1_1 \oplus K5_1 \oplus S_1^{-1}(K7_1, K3_1) \\ K5_0 &= K7_0 \oplus [K5_0 \oplus K7_0] \\ K3_0 &= K5_0 \oplus [K3_0 \oplus K5_0] \\ K1_0 &= K3_0 \oplus [K1_0 \oplus K3_0]. \end{aligned}$$

Now, we find two bytes of the key itself, one by the third round of the key processing and the other by the second round:

$$\begin{aligned} Key_7 &= K3_1 \oplus K5_0 \oplus S_1^{-1}(K5_1, K1_1) \\ Key_3 &= K1_1 \oplus K3_0 \oplus S_1^{-1}(K3_1, Key_7) \end{aligned}$$

and verify by the first round of the key processing that

$$S_1(K1_0 \oplus Key_7, Key_3) = K1_1.$$

For each remaining value we try all the 256 possibilities of $K4_0$. Then

$$\begin{aligned} K6_0 &= K4_0 \oplus [K4_0 \oplus K6_0] \\ K2_0 &= K4_0 \oplus [K2_0 \oplus K4_0]. \end{aligned}$$

By the fourth round of the key processing:

$$\begin{aligned} K6_1 &= K1_0 \oplus K5_0 \oplus S_0^{-1}(K6_0, K2_0) \\ K4_1 &= K6_1 \oplus [K4_1 \oplus K6_1] \\ K2_1 &= K4_1 \oplus [K2_1 \oplus K4_1] \\ K0_0 &= K4_0 \oplus K3_0 \oplus K3_1 \oplus S_1^{-1}(K6_1, K2_0 \oplus K2_1) \\ K0_1 &= K4_1 \oplus K6_1 \oplus S_0^{-1}(K7_0, K3_0 \oplus K3_1). \end{aligned}$$

The rest of the key can be found by the third round of the key processing:

$$\begin{aligned} Key_4 &= K2_0 \oplus K1_0 \oplus K1_1 \oplus S_1^{-1}(K4_1, K0_0 \oplus K0_1) \\ Key_5 &= K2_1 \oplus K4_1 \oplus S_0^{-1}(K5_0, K1_0 \oplus K1_1) \\ Key_6 &= K3_0 \oplus K4_1 \oplus S_0^{-1}(K4_0, K0_0) \end{aligned}$$

and by the second round:

$$\begin{aligned} Key_0 &= K0_0 \oplus Key_6 \oplus Key_7 \oplus S_1^{-1}(K2_1, Key_4 \oplus Key_5) \\ Key_1 &= K0_1 \oplus K2_1 \oplus S_0^{-1}(K3_0, Key_6 \oplus Key_7) \\ Key_2 &= K1_0 \oplus K2_1 \oplus S_0^{-1}(K2_0, Key_4). \end{aligned}$$

Given the key, we verify that it is really processed to the known actual subkeys and that the XOR of a decrypted pair of ciphertexts equals the chosen plaintext XOR value. If this verification succeeds then the calculated key is very likely to be the real key.

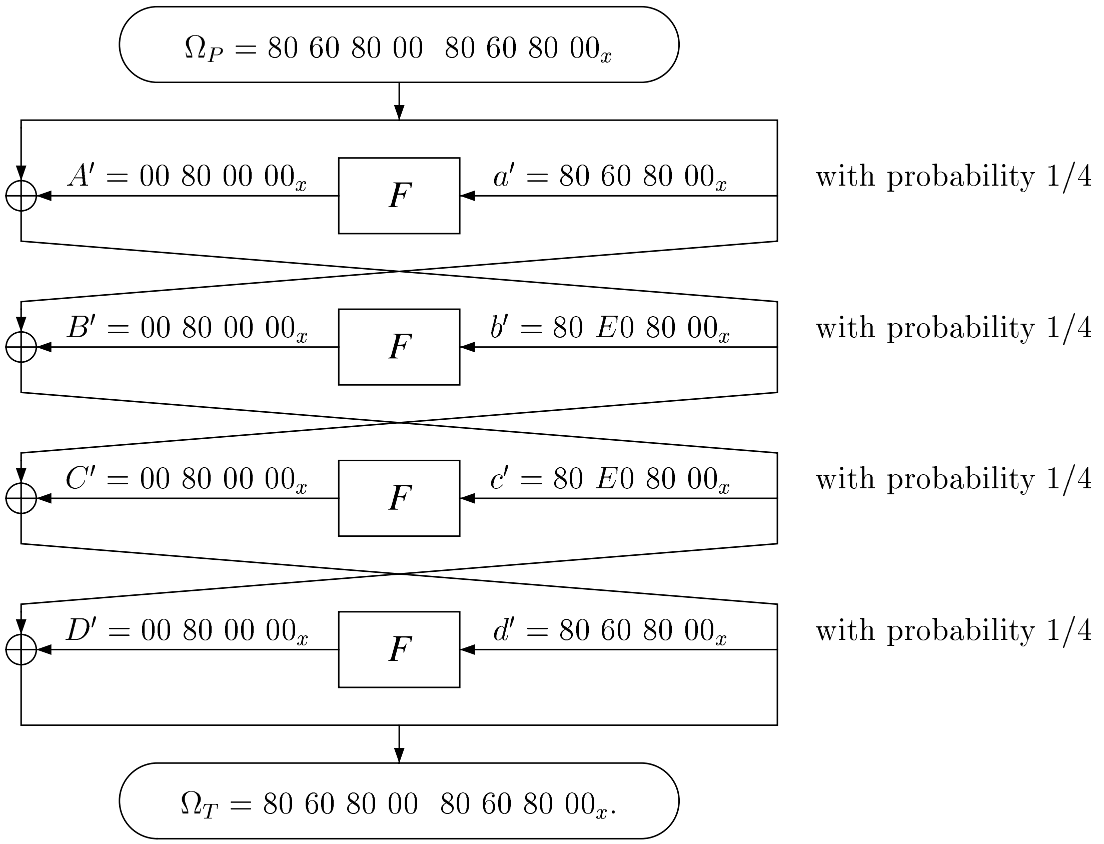
3.5 Results

This attack was implemented on a COMPAQ personal computer. It finds the key in less than two minutes using 1000 pairs with more than 95% success rate. Using quartets with two characteristics we need 1000 ciphertexts for this attack. Using 2000 pairs it finds the key with almost 100% success rate. The program uses 280K bytes of memory.

4 Cryptanalysis of Feal-N and Feal-NX with $N \leq 31$ rounds

Feal-N[6] was suggested as an N-round extension of Feal-8 after our attack on Feal-8 was announced. Feal-NX[7] is similar to Feal-N but uses a longer 128-bit key and a different key processing algorithm. Since our attack ignores the key processing algorithm and finds the actual subkeys, we can apply it to both Feal-N and Feal-NX with identical complexity and performance.

The attack on Feal with an arbitrary number of rounds is based on the following iterative characteristic (whose plaintext XOR is $P' = 80\ 60\ 80\ 00\ \ 00\ 00\ 00\ 00_x$):



The probability of each round of this characteristic is 1/4, and it can be concatenated to itself any number of times since the swapped value of the two halves of Ω_P equals Ω_T. Thus, for any arbitrary n, an n-round characteristic with probability $\frac{1}{4^n} = 2^{-2n}$ can be obtained.

An attack based on a characteristic which is shorter by two rounds than the cryptosystem is called a *2R-attack*. In this case, we know the ciphertext XOR T' and the input XOR of the last round (w.l.g. we employ the notation of an eight-round cryptosystem) h' by the ciphertext, and f' and g' by the characteristic. Thus, $G' = f' \oplus h'$ and $H' = g' \oplus l'$. Each pair is verified to have $g' \rightarrow G'$ and $h' \rightarrow H'$

and the resultant pairs are used in the process of counting the possibilities in order to find the last actual subkey. Two bits of the last actual subkey are indistinguishable. Therefore, we must try the following steps in parallel for the four possibilities of these two bits. The verification of $g' \rightarrow G'$ leaves only 2^{-19} of the pairs (since for either $g' = 80\ 60\ 80\ 00_x$ or $g' = 80\ E0\ 80\ 00_x$ there are only about 2^{13} possible output XORs G' and $2^{13} \cdot 2^{-32} = 2^{-19}$). The verification of $h' \rightarrow H'$ leaves 2^{-11} of the pairs (the fraction of the possible entries in the pairs XORs distribution table of the F function). The signal to noise ratio of this process is thus

$$S/N = \frac{2^{32}}{2^{2(N-2)} \cdot 2^{-19} \cdot 1} = 2^{55-2N}.$$

The identification leaves

$$I = 2^{2(N-2)} \cdot 2^{-19} \cdot 2^{-11} = 2^{2N-34}$$

wrong pairs for each right pair. Therefore, the right value of the last subkey is counted with a detectably higher probability than a random value up to $N \leq 28$ rounds, and thus we can break Feal-N with 2R-attacks for any $N \leq 28$ rounds, faster than via exhaustive search, as shown in table 1.

An attack based on a characteristic which is shorter by one round than the cryptosystem is called a *1R-attack*. Using 1R-attacks (w.l.g. we employ the notation of an eight-round cryptosystem), we know T' and h' from the ciphertext and g' and h' from the characteristic. Also, $H' = g' \oplus l'$. We can verify that h' calculated by the ciphertext equals the h' of the characteristic, and that $h' \rightarrow H'$. The successfully filtered pairs are used in the process of counting the number of times each possible value of the last actual subkey is suggested, and finding the most popular value. Complicating factors are the small number of bits set in h' (which is a constant defined by the characteristic), and the fact that many values of H' suggest many common values of the last actual subkey. The signal to noise ratio of this process is

$$S/N = \frac{2^{32}}{2^{2(N-1)} \cdot 2^{-32} \cdot 1} = 2^{66-2N}.$$

The identification leaves

$$I = 2^{2(N-1)} \cdot 2^{-32} \cdot 2^{-19} = 2^{2N-53}$$

wrong pairs for each right pair. Therefore, the right value of the last subkey is counted with detectably higher probability than a random value up to $N \leq 31$ rounds. A summary of the 1R-attacks on Feal-N appears in table 1, and shows that the differential cryptanalysis is faster than exhaustive search up to $N \leq 31$.

Note that in both the 1R-attacks and the 2R-attacks we use octets (structures of eight encryptions) with four characteristics (this is a special case in which an octet can have four characteristics since $\Omega_P^4 = \Omega_P^1 \oplus \Omega_P^2 \oplus \Omega_P^3$). These four characteristics are the four possible rotations of the given characteristic. Thus, each octet gives rise

	2R-attack					1R-attack				
N	Prob	S/N	I	Pairs	Data	Prob	S/N	I	Pairs	Data
8	2^{-12}	2^{39}	2^{-18}	2^{14}	2^{13}	2^{-14}	2^{50}	2^{-37}	2^{17}	2^{16}
9	2^{-14}	2^{37}	2^{-16}	2^{16}	2^{15}	2^{-16}	2^{48}	2^{-35}	2^{19}	2^{18}
10	2^{-16}	2^{35}	2^{-14}	2^{18}	2^{17}	2^{-18}	2^{46}	2^{-33}	2^{21}	2^{20}
11	2^{-18}	2^{33}	2^{-12}	2^{20}	2^{19}	2^{-20}	2^{44}	2^{-31}	2^{23}	2^{22}
12	2^{-20}	2^{31}	2^{-10}	2^{22}	2^{21}	2^{-22}	2^{42}	2^{-29}	2^{25}	2^{24}
13	2^{-22}	2^{29}	2^{-8}	2^{24}	2^{23}	2^{-24}	2^{40}	2^{-27}	2^{27}	2^{26}
14	2^{-24}	2^{27}	2^{-6}	2^{26}	2^{25}	2^{-26}	2^{38}	2^{-25}	2^{29}	2^{28}
15	2^{-26}	2^{25}	2^{-4}	2^{28}	2^{27}	2^{-28}	2^{36}	2^{-23}	2^{31}	2^{30}
16	2^{-28}	2^{23}	2^{-2}	2^{30}	2^{29}	2^{-30}	2^{34}	2^{-21}	2^{33}	2^{32}
17	2^{-30}	2^{21}	1	2^{32}	2^{31}	2^{-32}	2^{32}	2^{-19}	2^{35}	2^{34}
18	2^{-32}	2^{19}	2^{2}	2^{34}	2^{33}	2^{-34}	2^{30}	2^{-17}	2^{37}	2^{36}
19	2^{-34}	2^{17}	2^{4}	2^{36}	2^{35}	2^{-36}	2^{28}	2^{-15}	2^{39}	2^{38}
20	2^{-36}	2^{15}	2^{6}	2^{38}	2^{37}	2^{-38}	2^{26}	2^{-13}	2^{41}	2^{40}
21	2^{-38}	2^{13}	2^{8}	2^{40}	2^{39}	2^{-40}	2^{24}	2^{-11}	2^{43}	2^{42}
22	2^{-40}	2^{11}	2^{10}	2^{42}	2^{41}	2^{-42}	2^{22}	2^{-9}	2^{45}	2^{44}
23	2^{-42}	2^{9}	2^{12}	2^{44}	2^{43}	2^{-44}	2^{20}	2^{-7}	2^{47}	2^{46}
24	2^{-44}	2^{7}	2^{14}	2^{46}	2^{45}	2^{-46}	2^{18}	2^{-5}	2^{49}	2^{48}
25	2^{-46}	2^{5}	2^{16}	2^{49}	2^{48}	2^{-48}	2^{16}	2^{-3}	2^{51}	2^{50}
26	2^{-48}	2^{3}	2^{18}	2^{52}	2^{51}	2^{-50}	2^{14}	2^{-1}	2^{53}	2^{52}
27	2^{-50}	2	2^{20}	2^{55}	2^{54}	2^{-52}	2^{12}	2^{1}	2^{55}	2^{54}
28	2^{-52}	2^{-1}	2^{22}	2^{58}	2^{57}	2^{-54}	2^{10}	2^{3}	2^{57}	2^{56}
29	2^{-54}	2^{-3}	2^{24}			2^{-56}	2^{8}	2^{5}	2^{59}	2^{58}
30	2^{-56}					2^{-58}	2^{6}	2^{7}	2^{61}	2^{60}
31	2^{-58}					2^{-60}	2^{4}	2^{9}	2^{64}	$\underline{2^{63}}$
32	2^{-60}					2^{-62}	2^{2}	2^{11}	2^{67}	$\underline{2^{66}}$

Table 1. Attacks on Feal-N.

to 16 pairs (rather than four) which greatly reduces the required number of chosen plaintexts. In both kinds of attacks there are two indistinguishable bits at each of the last two actual subkeys. The attacking program should try all the 16 possible values of these bits when analyzing the earlier subkeys.

5 Differential Cryptanalytic Known Plaintext Attacks

Differential cryptanalytic attacks are chosen plaintext attacks in which the plaintext pairs can be chosen at random as long as they satisfy the plaintext XOR condition. Unlike other chosen plaintext attacks, differential cryptanalytic attacks can be easily converted to known plaintext attacks by the following observation.

Cryptosystem	Number of pairs of one char	Number of chosen plaintexts	Number of known plaintexts
Feal-4	4	8	$2^{33.5}$
Feal-8	1000	2000	$2^{37.5}$
Feal-12	2^{20}	2^{21}	$2^{42.5}$
Feal-16	2^{28}	2^{29}	$2^{46.5}$
Feal-20	2^{36}	2^{37}	$2^{50.5}$
Feal-24	2^{44}	2^{45}	$2^{54.5}$
Feal-28	2^{55}	2^{56}	2^{60}
Feal-30	2^{59}	2^{60}	2^{62}
Feal-31	2^{62}	2^{63}	$2^{63.5}$
DES-6	120	240	2^{36}
DES-8	25000	50000	2^{40}
DES-9	2^{25}	2^{26}	2^{45}
DES-10	2^{34}	2^{35}	$2^{49.5}$
DES-11	2^{35}	2^{36}	2^{50}
DES-12	2^{42}	2^{43}	$2^{53.5}$
DES-13	2^{43}	2^{44}	2^{54}
DES-14	2^{50}	2^{51}	$2^{57.5}$
DES-15	2^{51}	2^{52}	2^{58}

Table 2. Known plaintext attacks on Feal and DES.

Assume that the differential cryptanalytic chosen plaintext attack needs m pairs, and that we are given $2^{32} \cdot \sqrt{2m}$ random known plaintexts and their corresponding ciphertexts. Consider all the $\frac{\left(2^{32} \cdot \sqrt{2m}\right)^2}{2} = 2^{64} \cdot m$ possible pairs of plaintexts they can form. Each pair has a plaintext XOR which can be easily calculated. Since the block size is 64 bits, there are only 2^{64} possible plaintext XOR values, and thus there are about $\frac{2^{64} \cdot m}{2^{64}} = m$ pairs creating each plaintext XOR value. In particular, with high probability there are about m pairs with each one of the several plaintext XOR values needed for differential cryptanalysis.

The known plaintext attack is not limited to the electronic code book (ECB) mode of operation. In particular, the cipher block chaining (CBC) mode can also be broken by this attack since when the plaintexts and the ciphertexts are known, it is easy to calculate the real input of the encryption function.

Table 2 summarizes the differential cryptanalytic known plaintext attacks on Feal and DES. For each of the listed cryptosystems with the listed number of rounds, the table describes the number of pairs of each characteristic and the total number of random plaintexts needed for the chosen plaintext attack and for the known plaintext attack.

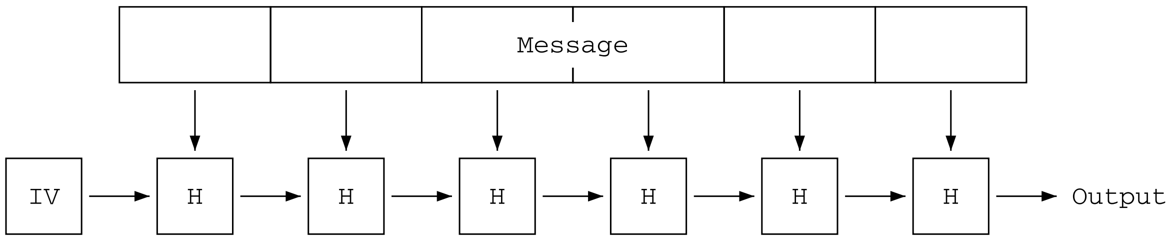


Figure 2. Outline of N-Hash.

6 Cryptanalysis of N-Hash

N-Hash[8] is designed as a cryptographically strong hash function which hashes messages of arbitrary length into 128-bit values. The messages are divided into 128-bit blocks, and each block is mixed with the hashed value computed so far by a randomizing function g. The new hashed value is the XOR of the output of the g-function with the block itself and with the old hashed value. The g-function contains eight randomizing rounds, and each one of them calls the F function (similar to the one of Feal) four times. A graphic description of N-Hash is given in figures 2, 3, and 4.

Breaking a cryptographically strong hash function means finding two different messages which hash to the same value. In particular, we break N-Hash by finding two different 128-bit messages which are hashed to the same 128-bit value. Since the output of the g-function is XORed with its input in order to form the hashed value, it suffices to find a right pair for a characteristic of the g-function in which $\Omega_P = \Omega_T$. After XORing the input with the output of the g-function, the hashed value XOR becomes zero and thus the two messages have the same hashed value.

The following characteristic is a three-round iterative characteristic with probability 2^{-16} (N-Hash does not swap the two halves after each round since the swap operation is part of the round itself. Therefore, the concatenation of the characteristic Ω^1 with the characteristic Ω^2 is possible whenever $\Omega^1_T = \Omega^2_P$ without swapping). In the description of this characteristic we refer to the value $80\ 60\ 80\ 00_x$ as ψ and to the value $80\ E0\ 80\ 00_x$ as φ. Note that both $\psi \rightarrow (\psi \oplus \varphi)$ and $\varphi \rightarrow (\psi \oplus \varphi)$ with probability $\frac{1}{4}$ by the F function. The behavior of the XORs in the F function in this characteristic is similar to their behavior in the iterative characteristic of Feal. The characteristic itself is based on the input XOR:

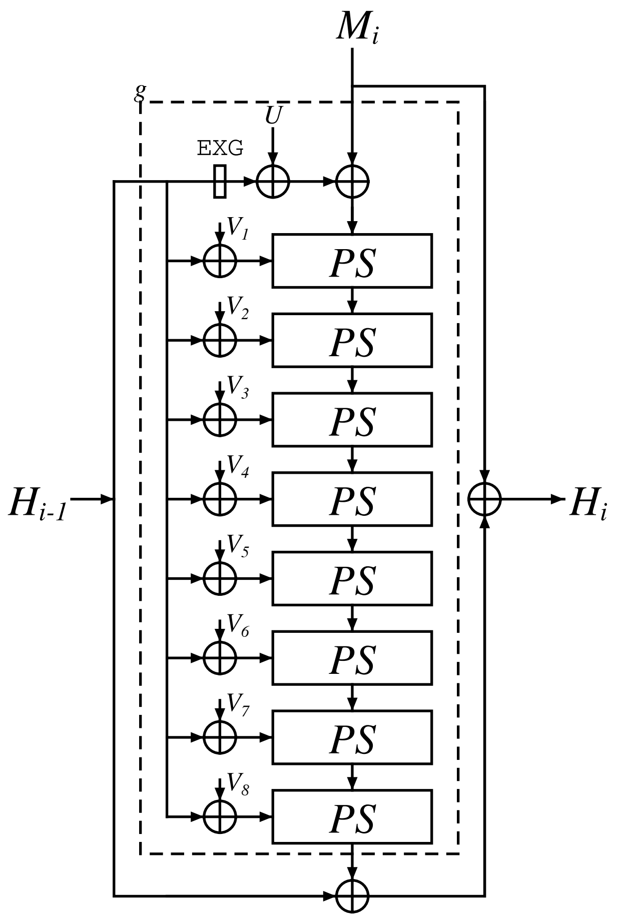
$$\Omega_P = (\psi, \psi, 0, 0).$$

With probability $\frac{1}{256}$ the data XOR after the first round is

$$(0, 0, \varphi, \varphi).$$

With probability $\frac{1}{256}$ the data XOR after the second round is

$$(\psi, \psi, \varphi, \varphi).$$



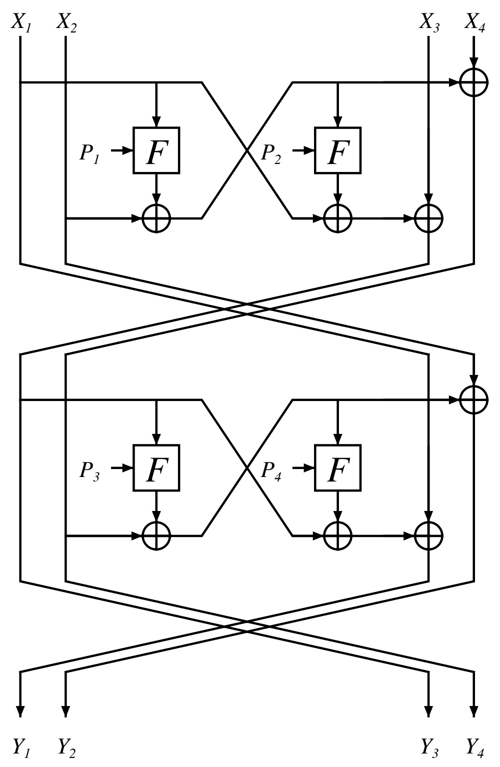


Figure 3. The function H and one round (PS) of N-Hash.

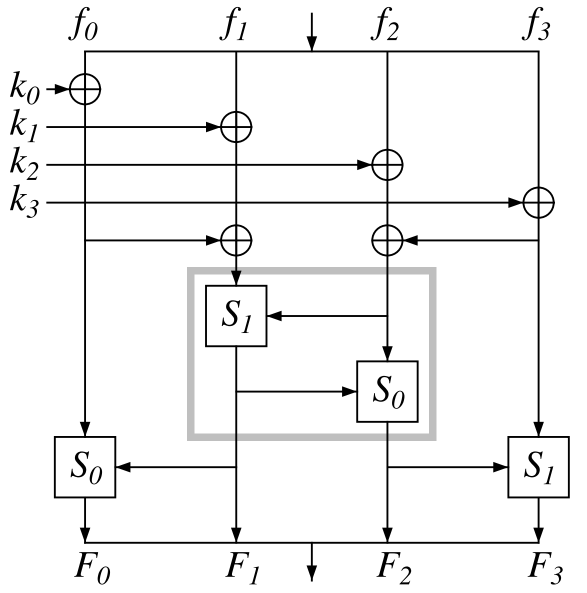


Figure 4. The F function of N-Hash.

Number of Rounds	Complexity
3	2^8
6	2^{24}
9	2^{40}
12	2^{56}
15	2^{72}

Table 3. Results of the attack on N-Hash.

The data after the third round is always

$$\Omega_T = \Omega_P = (\psi, \psi, 0, 0).$$

Therefore, the probability of the characteristic is 2^{-16}.

A pair of messages whose XOR equals Ω_P has probability $(2^{-16})^2 = 2^{-32}$ to have Ω_T as its output XOR after the sixth round of the g-function, and thus to have the same hashed value after their inputs and outputs are XORed by the six-round variant of N-Hash. Instead of trying about 2^{32} random pairs of messages we can choose only pairs from a smaller set in which the characteristic is guaranteed to be satisfied in the four F functions of the first round. The pairs in this set are chosen by the following algorithm. For each F function in the first round we search a priori a list of input pairs for which the input XOR and the output XOR are as expected by the characteristic. To get a new pair we choose a random input pair for each F function and from the four input pairs and their corresponding outputs we deduce the two messages backwards. Therefore, the probability in this set is increased by a factor of 256, and only about 2^{24} such pairs have to be tested in order to find a pair of messages which hash to the same value.

This specific attack works only for variants of N-Hash whose number of rounds is divisible by three. Table 3 describes the results of this attack. We can see from the table that this attack is faster than the birthday attack (whose complexity is 2^{64}) for variants of N-Hash with up to 12 rounds.

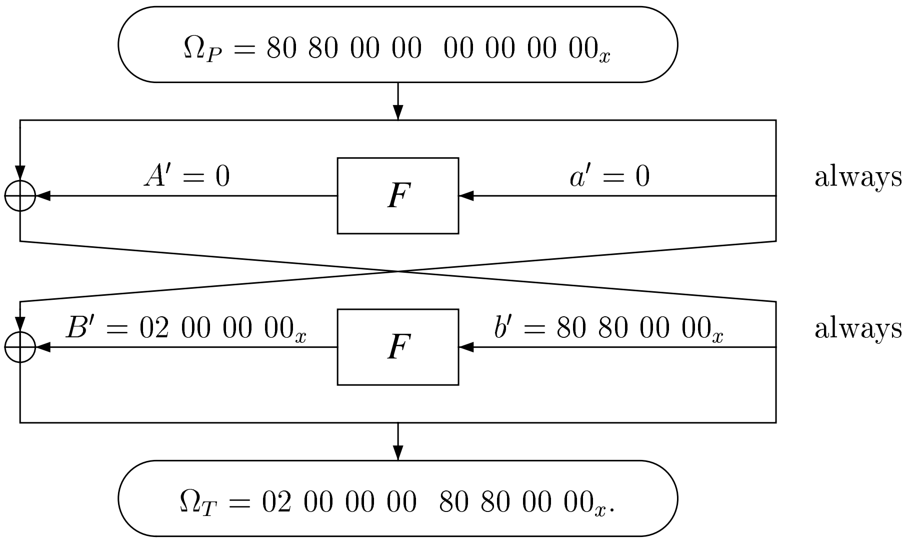
The attack on N-Hash with six rounds was implemented on a personal computer and the following pairs of messages (as well as many others) were found within about two hours:

- - `CAECE595 127ABF3C 1ADE09C8 1F9AD8C2`
  - `4A8C6595 921A3F3C 1ADE09C8 1F9AD8C2`
  - Common hash value: `12B931A6 399776B7 640B9289 36C2EF1D`
- - `5878BE49 F2962D67 30661E17 0C38F35E`
  - `D8183E49 72F6AD67 30661E17 0C38F35E`
  - Common hash value: `29B0FE97 3D179E0E 5B147598 137D28CF`.

7 Cryptanalysis of Feal-4

Feal-4 is breakable by a chosen plaintext attack which uses eight ciphertexts and the plaintext of one of them. We keep the notation used in the attack on Feal-8. Note that the attack described here really breaks an extension of Feal-4 whose all subkeys are 32-bit long.

We use the following two-round characteristic with probability 1 (for which $P' = 80\ 80\ 00\ 00\ 80\ 80\ 00\ 00_x$):


$\Omega_P = 80\ 80\ 00\ 00\ \ 00\ 00\ 00\ 00_x$
$A' = 0$
F
$a' = 0$
always
$B' = 02\ 00\ 00\ 00_x$
F
$b' = 80\ 80\ 00\ 00_x$
always
$\Omega_T = 02\ 00\ 00\ 00\ \ 80\ 80\ 00\ 00_x$.

A right pair with respect to this characteristic (and therefore any pair with this plaintext XOR P') satisfy

$$c' = 02\ 00\ 00\ 00_x.$$

From the other direction

$$d' = r' \oplus l'.$$

Thus,

$$\begin{aligned} C' &= d' \oplus b' = r' \oplus l' \oplus 80\ 80\ 00\ 00_x \\ D' &= l' \oplus c' = l' \oplus 02\ 00\ 00\ 00_x. \end{aligned}$$

The last actual subkey of this cryptosystem is $AK3$. Given the value of $AK3$ the value of D can be calculated for any ciphertext. For each possible value of $AK3$ we count the number of pairs for which D' calculated above from the characteristic equals D' calculated using $AK3$ and for which $c' \rightarrow C'$. The value of $AK3$ which is counted by all the pairs must be the right value. There is only a small probability that more than one such value is counted by all the pairs using four pairs. This counting can be done with complexity 2^{16} by counting the possible values of $\text{mx}(AK3)$, comparing D' and then counting the values of $AK3$ whose $\text{mx}(AK3)$ is as found in the first step.

Given $AK3$ we can reduce the cryptosystem to three rounds. For each possible value of $AK2$ we count the number of pairs whose values of C' from both directions are equal using another characteristic. The value which is counted by all the pairs is the real value of $AK2$. Similarly we find $AK1$ and $AK0$ using other characteristics. The value of the actual subkey used in the initial transformation is easily found using the given plaintext.

In the search for $AK2$ we use a one-round characteristic with probability 1 which cannot be extended to two rounds with probability 1, since otherwise the input XOR of the third round would be constant for all the pairs. In the search for $AK1$ and $AK0$ we use pairs with random plaintext XORs. All the plaintext XORs needed can be obtained by a structure of eight encryptions.

8 A Known Plaintext Attack on Feal-4

This known plaintext attack is based on the property of the addition operation that there is a fixed pattern of carry bits which is generated when many pairs of eight-bit numbers are added together. This carry type depends on the additional constant which is added to the sum. A similar attack is applicable to Feal-5.

Definition 2 Let X and Y be eight-bit variables and let i be an eight-bit constant. The *carry type of the sum* $X+Y+i$ is defined to be $(X+Y+i \pmod{256}) \oplus (X \oplus Y \oplus i)$. The *carry type of the sum* $X+Y$ is an abbreviation of the carry type of the sum $X+Y+0$.

Note that the carry types always end with a zero.

The following lemma derives the main properties of carry types:

Lemma 1 Let X and Y be eight-bit numbers. Then:

1. A fraction of $\left(\frac{3}{4}\right)^7 \approx \frac{1}{7.5}$ of all the sums $X+Y$ have carry type 0.
2. A fraction of $\left(\frac{3}{4}\right)^7 \approx \frac{1}{7.5}$ of all the sums $X+Y+1$ have carry type FE_x.
3. A fraction of $\left(\frac{10}{16}\right)^7 \approx \frac{1}{27}$ of all the pairs of sums X_1+Y_1 and X_2+Y_2 have the same carry type for both sums. The same fraction holds for the sums X_1+Y_1+1 and X_2+Y_2+1.

Proof 1. $(X+Y)\oplus X \oplus Y = 0$ if for any $j \in \{0,\ldots,6\}$ either $X_j = 0$ or $Y_j = 0$. If $X_j = Y_j = 1$ there is a carry from bit j to bit $j+1$. Therefore, for each bit, in three out of four cases there is no carry and the total fraction is $(\frac{3}{4})^7 \approx \frac{1}{7.5}$.

2. $(X+Y+1) \oplus X \oplus Y \oplus 1 = FE_x$ if for any $j \in \{0,\ldots,6\}$ either $X_j = 1$ or $Y_j = 1$. If $X_j = Y_j = 0$ there is no carry from bit j to bit $j+1$. Therefore, for each bit, in three out of four cases there is a carry and the total fraction is $(\frac{3}{4})^7 \approx \frac{1}{7.5}$.

3. If the two carry types are equal then any bit which has a carry in one addition has a carry in the other and any bit which does not have a carry in one addition, has no carry in the other. The probability for one bit to satisfy it is $\frac{3}{4}\cdot\frac{3}{4}+\frac{1}{4}\cdot\frac{1}{4} = \frac{10}{16}$ and the total fraction is $(\frac{10}{16})^7 \approx \frac{1}{27}$.

■

For each encryption with plaintext P and ciphertext T the value of $A \oplus C$ is known up to a XOR with a key dependent value by

$$A \oplus C = L \oplus KL \oplus l \oplus Kl \oplus r \oplus Kr.$$

where (KL, KR) are the subkeys XORed with the plaintext during the encryption process and (Kl, Kr) are the subkeys XORed with the ciphertext. Lets concentrate on

$$A_0 \oplus C_0 = L_0 \oplus KL_0 \oplus l_0 \oplus Kl_0 \oplus r_0 \oplus Kr_0.$$

A_0 and C_0 are

$$\begin{aligned} A_0 &= S_0(a_0, A_1) = \mathrm{ROL2}(a_0 + A_1) \\ C_0 &= S_0(c_0, C_1) = \mathrm{ROL2}(c_0 + C_1). \end{aligned}$$

$a_0 + A_1$ and $c_0 + C_1$ have the same carry type Z with probability about $\frac{1}{27}$. In this case

$$\begin{aligned} A_0 &= \mathrm{ROL2}(a_0 \oplus A_1 \oplus Z) \\ C_0 &= \mathrm{ROL2}(c_0 \oplus C_1 \oplus Z) \end{aligned}$$

and

$$\begin{aligned} A_0 \oplus C_0 &= \mathrm{ROL2}(a_0 \oplus A_1 \oplus Z \oplus c_0 \oplus C_1 \oplus Z) = \\ &= \mathrm{ROL2}(a_0 \oplus A_1 \oplus c_0 \oplus C_1). \end{aligned}$$

The value $A_1 \oplus C_1$ is known up to the key by

$$A_1 \oplus C_1 = L_1 \oplus KL_1 \oplus l_1 \oplus Kl_1 \oplus r_1 \oplus Kr_1$$

and a_0 is just

$$a_0 = L_0 \oplus KL_0 \oplus R_0 \oplus KR_0.$$

Thus,

$$\begin{aligned} &L_0 \oplus KL_0 \oplus l_0 \oplus Kl_0 \oplus r_0 \oplus Kr_0 = \\ &\quad \mathrm{ROL2}(L_0 \oplus KL_0 \oplus R_0 \oplus KR_0 \oplus c_0 \oplus \\ &\qquad L_1 \oplus KL_1 \oplus l_1 \oplus Kl_1 \oplus r_1 \oplus Kr_1). \end{aligned}$$

Extracting c_0:

$$\begin{aligned} c_0 &= L_0 \oplus R_0 \oplus L_1 \oplus l_1 \oplus r_1 \oplus \\ &\quad \mathrm{ROR2}(L_0 \oplus l_0 \oplus r_0) \oplus \\ &\quad KL_0 \oplus KR_0 \oplus KL_1 \oplus Kl_1 \oplus Kr_1 \oplus \\ &\quad \mathrm{ROR2}(KL_0 \oplus Kl_0 \oplus Kr_0). \end{aligned} \tag{2}$$

On the other hand:

$$\begin{aligned} c_0 \oplus D_0 &= l_0 \oplus Kl_0 \\ D_0 &= S_0(d_0, D_1) = \mathrm{ROL2}(d_0 + D_1) \\ D_1 &= S_1(d_0 \oplus d_1 \oplus K4_0, d_2 \oplus d_3 \oplus K4_1) = \\ &= \mathrm{ROL2}((d_0 \oplus d_1 \oplus K4_0) + (d_2 \oplus d_3 \oplus K4_1) + 1) \end{aligned}$$

with probability about $\frac{1}{7.5}$:

$$D_0 = \mathrm{ROL2}(d_0 \oplus D_1)$$

and with probability about $\frac{1}{7.5}$:

$$\begin{aligned} D_1 &= \mathrm{ROL2}((d_0 \oplus d_1 \oplus K4_0) \oplus (d_2 \oplus d_3 \oplus K4_1) \oplus 1 \oplus FE_x) = \\ &= \mathrm{ROL2}(d_0 \oplus d_1 \oplus d_2 \oplus d_3 \oplus K4_0 \oplus K4_1 \oplus FF_x) \end{aligned}$$

where for $i \in \{0, \ldots, 3\}$

$$d_i = l_i \oplus r_i \oplus Kl_i \oplus Kr_i. \tag{3}$$

Thus,

$$\begin{aligned} c_0 &= D_0 \oplus l_0 \oplus Kl_0 = \\ &= \mathrm{ROL2}(d_0 \oplus D_1) \oplus l_0 \oplus Kl_0 = \\ &= FF_x \oplus l_0 \oplus Kl_0 \oplus \mathrm{ROL2}(l_0 \oplus r_0 \oplus Kl_0 \oplus Kr_0) \oplus \\ &\quad \mathrm{ROL4}(l_0 \oplus l_1 \oplus l_2 \oplus l_3 \oplus r_0 \oplus r_1 \oplus r_2 \oplus r_3 \oplus \\ &\quad Kl_0 \oplus Kl_1 \oplus Kl_2 \oplus Kl_3 \oplus Kr_0 \oplus Kr_1 \oplus Kr_2 \oplus Kr_3 \oplus \\ &\quad K4_0 \oplus K4_1). \end{aligned} \tag{4}$$

By equating equations 2 and 4 and dividing the variables into key variables K_C and plaintext/ciphertext variables E_C we get with probability about $\frac{1}{27} \cdot \left(\frac{1}{7.5}\right)^2$:

$$K_C = E_C$$

where

$$\begin{aligned} K_C &= Kl_0 \oplus KL_0 \oplus KR_0 \oplus KL_1 \oplus Kl_1 \oplus Kr_1 \oplus \\ &\quad \mathrm{ROR2}(KL_0 \oplus Kl_0 \oplus Kr_0) \oplus \\ &\quad \mathrm{ROL2}(Kl_0 \oplus Kr_0) \oplus \\ &\quad \mathrm{ROR4}(Kl_0 \oplus Kl_1 \oplus Kl_2 \oplus Kl_3 \oplus \\ &\qquad Kr_0 \oplus Kr_1 \oplus Kr_2 \oplus Kr_3 \oplus K4_0 \oplus K4_1) \\ E_C &= FF_x \oplus l_0 \oplus L_0 \oplus R_0 \oplus L_1 \oplus l_1 \oplus r_1 \oplus \\ &\quad \mathrm{ROR2}(L_0 \oplus l_0 \oplus r_0) \oplus \\ &\quad \mathrm{ROL2}(l_0 \oplus r_0) \oplus \\ &\quad \mathrm{ROL4}(l_0 \oplus l_1 \oplus l_2 \oplus l_3 \oplus r_0 \oplus r_1 \oplus r_2 \oplus r_3). \end{aligned}$$

K_C is a constant depending on the key only. E_C can be calculated for every plaintext/ciphertext pair. The probability that $E_C = K_C$ for a plaintext/ciphertext pair is greater than 1/256, since the probability we calculated is added to the probability of random occurrence. In addition, other carry phenomena cancel each other and increase the probability of this case. It is possible to prove the following:

- The probability of $E_C = K_C$ in a random plaintext/ciphertext pair is about 1/220.
- Given about 100000 plaintext/ciphertext pairs we can count the number of occurrences of each possible value of E_C and with a high probability the most frequent value is the value of K_C.

The value of K_C does not provide any practical knowledge about the key. However, using K_C we can filter the data leaving only those encryptions satisfying $E_C = K_C$. This filtration enrich the fraction of the plaintext/ciphertext pairs which have a zero carry type at the corresponding S boxes. If the carry type is zero in the S box outputting D_1:

$$\begin{aligned} D_1 &= \mathrm{ROL2}((d_0 \oplus d_1 \oplus K4_0) + (d_2 \oplus d_3 \oplus K4_1) + 1) = \\ &= \mathrm{ROL2}(d_0 \oplus d_1 \oplus d_2 \oplus d_3 \oplus K4_0 \oplus K4_1 \oplus FF_x) \end{aligned}$$

i.e., by equation 3

$$\begin{aligned} & (l_0 \oplus r_0 \oplus l_1 \oplus r_1 \oplus Kl_0 \oplus Kr_0 \oplus Kl_1 \oplus Kr_1 \oplus K4_0) \\ + \; & (l_2 \oplus r_2 \oplus l_3 \oplus r_3 \oplus Kl_2 \oplus Kr_2 \oplus Kl_3 \oplus Kr_3 \oplus K4_1) + 1 = \\ = \; & l_0 \oplus r_0 \oplus l_1 \oplus r_1 \oplus l_2 \oplus r_2 \oplus l_3 \oplus r_3 \\ \oplus \; & Kl_0 \oplus Kr_0 \oplus Kl_1 \oplus Kr_1 \oplus Kl_2 \oplus Kr_2 \oplus Kl_3 \oplus Kr_3 \\ \oplus \; & K4_0 \oplus K4_1 \oplus FF_x \end{aligned} \tag{5}$$

Trying all the 2^{16} possibilities of

$$Kl_0 \oplus Kr_0 \oplus Kl_1 \oplus Kr_1 \oplus K4_0$$

and

$$Kl_2 \oplus Kr_2 \oplus Kl_3 \oplus Kr_3 \oplus K4_1$$

we count the occurrences of the values satisfying equation 5. The value that occurs most often is likely to be the real value. One bit is indistinguishable and for the others we need much more data than in the caes of K_C. However, the XOR of these two values is usually the right value of their XOR.

Using those pairs we know D_1 (assuming the carry type is FE_x) and can assume a zero carry type in $D_0 = S_0(d_0, D_1)$ to find more key bits. Similar calculations can then find all the bits of the last actual subkey. The other actual subkeys can be found with much better identification after the reduction to a smaller number of rounds.

The attacking program finds the actual subkeys in less than two minutes on a personal computer using 100000 known plaintexts/ciphertext pairs. The program uses 250K bytes of memory.

A Other Properties of Feal

In this appendix we describe several properties of Feal which are not described elsewhere in this paper.

1. The F function is partially invertible: Given the value $Y = F(X, K)$ we can find all the internal values inside the F function and half of the actual input bytes by:

$$\begin{aligned} X_0 &= S_0^{-1}(Y_0, Y_1) \\ X_3 &= S_1^{-1}(Y_3, Y_2) \\ X_2 \oplus K_1 &= X_2 \oplus X_3 \oplus K_1 = S_0^{-1}(Y_2, Y_1) \\ X_1 \oplus K_0 &= X_0 \oplus X_1 \oplus K_0 = S_1^{-1}(Y_1, [X_2 \oplus K_1]). \end{aligned}$$

2. The F_k function of the key processing algorithm is partially invertible: Let $Z = F_k(X, Y)$. Then, given any three values out of Z_2, Z_3, X_3, Y_3, the fourth value is easily calculated using the formula:

$$Z_3 = S_1(X_3, Z_2 \oplus Y_3).$$

   In particular,

$$Z_{3,2} = X_{3,0} \oplus Z_{2,0} \oplus Y_{3,0} \oplus 1$$

   since S is linear in the least significant bit of the addition operation.

3. The following equation of the subkeys is satisfied by Feal-8:

$$Kef_{3,2} \oplus Kcd_{3,2} = Kcd_{3,0} \oplus Kef_{2,0} \oplus Kcd_{2,0} \oplus K7_{1,0}$$

   or in other writing, by the actual subkeys:

$$AK7_{3,2} = AK6_{3,0} \oplus AK7_{2,0}.$$

   Therefore, given the value of $AK7$, it is easy to calculate the value of the bit $AK6_{3,0}$. This property is used to discard wrong values of AK6 during the search for the actual subkeys.

4. The key processing algorithm of Feal-8 yields 256 subkey bits, of which 32 bits are redundant. Only 224 bits are needed during the encryption/decryption

processes. They are:

$$\begin{array}{rcl}
K0^\dagger & = & K0 \oplus \widehat{Kcd} \\
K1^\dagger & = & K1 \oplus \widehat{Kcd} \oplus \widehat{Kef} \\
K2^\dagger & = & K2 \oplus \widehat{Kcd} \\
K3^\dagger & = & K3 \oplus \widehat{Kcd} \oplus \widehat{Kef} \\
K4^\dagger & = & K4 \oplus \widehat{Kcd} \\
K5^\dagger & = & K5 \oplus \widehat{Kcd} \oplus \widehat{Kef} \\
K6^\dagger & = & K6 \oplus \widehat{Kcd} \\
K7^\dagger & = & K7 \oplus \widehat{Kcd} \oplus \widehat{Kef} \\
K89^\dagger & = & K89 \oplus \mathrm{am}(\widehat{Kcd} \oplus \widehat{Kef}) \\
Kab^\dagger & = & Kab \oplus \mathrm{am}(\widehat{Kef}) \\
Kcd^\dagger & = & (Kcd_0, 0, 0, Kcd_3) \\
Kef^\dagger & = & (Kef_0, 0, 0, Kef_3)
\end{array}$$

where for any 32-bit X, $\hat{X}$ is the 16-bit value of its two middle bytes (i.e., (X_1, X_2)). The encryption and decryption using the new values of the subkeys give the same results as with the original values. Another equivalent description of the subkeys is denoted by the actual subkeys in which the subkeys of the rounds are extended to 32 bits and the subkey of the final transformation is eliminated.

5. The following property can be used to decide if some input XOR value may cause some output XOR value by the F function and to find real values of bits by the input XOR and the output XOR. The decision is done in parallel for each S box in the F function.

   Let $Z = S_i(X, Y)$ and $Z^* = S_i(X^*, Y^*)$. The lowest bit in the addition operation satisfy

   $$Z_2' = X_0' \oplus Y_0'.$$

   Let C be the byte of carries in the addition operation $(X + Y + i) \pmod{256}$ in S_i, defined as $C = (X + Y + i \pmod{256}) \oplus X \oplus Y$ (i (which is either zero or one) is interpreted here as a carry into the least significant bit). C_j is the carry bit passed from the $(j-1)^{\text{th}}$ bit of the addition in S_i to the j^{th} bit. Thus,

   $$\forall j \in \{1, \ldots, 7\} : \; C_j = \begin{cases} 1, & \text{if } X_{j-1} + Y_{j-1} + C_{j-1} \geq 2; \\ 0, & \text{if } X_{j-1} + Y_{j-1} + C_{j-1} \leq 1 \end{cases}$$

   and C_j' is the value of $C_j \oplus C_j^*$. $C_0 = i$ and thus the value of C_0' is always zero. Let W be defined as $\mathrm{ROR2}(Z) = (X + Y + i) \pmod{256}$. Then,

   $$C = W \oplus X \oplus Y$$

   and C' is easily calculated from the input XORs and the output XOR by

   $$C' = W' \oplus X' \oplus Y'.$$

X'_j	Y'_j	$C'_j = 1$	$C'_j = 0$
0	0	$X_j \oplus Y_j = C'_{j+1}{}^{\ddagger}$	$C'_{j+1} = 0^*$
0	1	$Y_j \oplus C_j = C'_{j+1} \oplus 1$	$X_j \oplus C_j = C'_{j+1}{}^{\dagger}$
1	0	$X_j \oplus C_j = C'_{j+1} \oplus 1$	$Y_j \oplus C_j = C'_{j+1}{}^{\dagger}$
1	1	$C'_{j+1} = 1^*$	$W_j \oplus C_j = X_j \oplus Y_j = C'_{j+1} \oplus 1^{\ddagger}$

Table 4. Possible known values given the XORs in pairs.

The combination of the values of X'_j, Y'_j, C'_j and C'_{j+1} (for $j \in \{0, \ldots, 6\}$) can derive some new knowledge. For example, assume that $X'_j = Y'_j = 0$ and $C'_j = 1$ and study the two possibilities of C'_{j+1}. If $C'_{j+1} = 0$ then either (1) $X_j + Y_j + C_j \leq 1$ and $X^*_j + Y^*_j + C^*_j \leq 1$ and thus $X_j = Y_j = 0$, or (2) $X_j + Y_j + C_j \geq 2$ and $X^*_j + Y^*_j + C^*_j \geq 2$ and thus $X_j = Y_j = 1$. In both cases $X_j = Y_j$. If $C'_{j+1} = 1$ then similarly $X_j \neq Y_j$ and therefore in general $X_j \oplus Y_j = C'_{j+1}$. Table 4 generalizes this observation for all the combinations of X'_j, Y'_j and C'_j. The entries marked by * are particularly useful because they can be used to identify wrong pairs. The entries marked by † can be used to derive the values of the bits X_0 and Y_0. The entries marked by ‡ can be used to derive the value of $X_j \oplus Y_j$ and the value of Z_2 (W_0).

The F function contains four S boxes. Some input bytes are used as inputs to two S boxes and the output bytes of some S boxes are used as inputs to other S boxes. By combining the knowledge obtained from the four S boxes we can find contradictions on the values of bits, or calculate by one S box the value of bits needed in another S box.

References

[1] Eli Biham, Adi Shamir, *Differential Cryptanalysis of DES-like Cryptosystems (extended abstract)*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'90, pp. 2–21, 1990.

[2] Eli Biham, Adi Shamir, *Differential Cryptanalysis of DES-like Cryptosystems*, Journal of Cryptology, Vol. 4, No. 1, pp. 3–72, 1991.

[3] Bert Den-Boer, *Cryptanalysis of F.E.A.L.*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'88, pp. 293–300, 1988.

[4] Walter Fumy, *On the F-function of FEAL*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'87, pp. 434, 1987.

[5] Henry Gilbert, Guy Chasse, *A Statistical Attack on the FEAL-8 Cryptosystem*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'90, pp. 22–33, 1990.

[6] Shoji Miyaguchi, *FEAL-N specifications*, technical note, NTT, 1989.

[7] Shoji Miyaguchi, *The FEAL cipher family*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'90, pp. 627–638, 1990.

[8] S. Miyaguchi, K. Ohta, M. Iwata, *128-bit hash function (N-Hash)*, proceedings of SECURICOM'90, pp. 123–137, March 1990.

[9] Shoji Miyaguchi, Akira Shiraishi, Akihiro Shimizu, *Fast Data Encryption Algorithm FEAL-8*, Review of electrical communications laboratories, Vol. 36, No. 4, pp. 433–437, 1988.

[10] Sean Murphy, *The Cryptanalysis of FEAL-4 with 20 Chosen Plaintexts*, The Journal of Cryptology, Vol. 2, No. 3, pp. 145–154, 1990.

[11] National Bureau of Standards, *Data Encryption Standard*, U.S. Department of Commerce, FIPS pub. 46, January 1977.

[12] Akihiro Shimizu, Shoji Miyaguchi, *Fast Data Encryption Algorithm FEAL*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'87, pp. 267–278, 1987.

[13] Akihiro Shimizu, Shoji Miyaguchi, *Fast Data Encryption Algorithm FEAL*, Abstracts of EUROCRYPT'87, pp. VII-11–VII-14, April 1987.