An Analysis of FOX

Jorge Nakahara Jr

[1]jorge_nakahara@yahoo.com.br

Abstract. *This paper[1] presents new cryptanalytic results on reduced-round versions of the FOX block cipher, also known as IDEA-NXT. We can recover all subkeys of 2-round variants of FOX, and derive internal cipher data from r-round FOX, for any $r > 2$. This information leakage phenomenon is based only on the high-level Lai-Massey scheme, and was already observed in Feistel ciphers such as DES, but is absent even in IDEA, whose design inspired the FOX ciphers. Moreover, this paper presents the first impossible-differential analysis of reduced-round FOX, and new results on 4-round and 5-round FOX.*

1. Introduction

FOX, also known as IDEA-NXT, is a family of block ciphers designed by P. Junod and S. Vaudenay [Junod and Vaudenay 2004]. There are two main variants of FOX, whose parameters are specified in Table 1, where $12 \leq r \leq 255$, and $0 \leq k \leq 256$, with k a multiple of 8.

The high-level structure of the FOX ciphers uses the Lai-Massey scheme, originally designed for the IDEA block cipher [Lai et al. 1991, Lai 1995]. All attacks and internal cipher data recovered from FOX in this paper were obtained due to the high-level structure of a round, which is based on the round structure of IDEA.

FOX ciphers are byte oriented, and all byte operations are performed in $\mathrm{GF}(2^8) = GF(2)[x]/(p(x))$, where $p(x) = x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1$ is an irreducible polynomial over GF(2). FOX64 and FOX64/k/r encryption operations iterate $r - 1$ (full) rounds, denoted **lmor64**, followed by a final round denoted **lmid64**. Formally, **lmor64**, **lmid64**: $\mathbb{Z}_2^{64} \times \mathbb{Z}_2^{64} \rightarrow \mathbb{Z}_2^{64}$, with inputs a 64-bit data block and 64-bit subkey, and as output a 64-bit data block (Fig.2). The encryption of a 64-bit data block P, under a key K, results in the ciphertext block $C =$ **lmid64**(**lmor64**(... **lmor64** (P, RK_0), ..., RK_{r-2}), RK_{r-1}), where the 64-bit subkeys RK_i, $0 \leq i \leq r - 1$, are derived from K according to a key schedule algorithm. The decryption transformation uses a round function called **lmio64**: $\mathbb{Z}_2^{64} \times \mathbb{Z}_2^{64} \rightarrow \mathbb{Z}_2^{64}$, and recovers the plaintext block P, from a given ciphertext block C, and key K, as $P =$ **lmid64**(**lmio64**(... **lmio64** (C, RK_{r-1}), ..., RK_1), RK_0).

[1]Research funded by FAPESP under contract 2005/02102-9.

Table 1. Parameters of the FOX ciphers.

Cipher	Block Size (bits)	Key Size (bits)	#Rounds
FOX64	64	128	16
FOX128	128	256	16
FOX64/k/r	64	k ($0 \leq k \leq 256$)	r ($12 \leq r \leq 255$)
FOX128/k/r	128	k ($0 \leq k \leq 256$)	r ($12 \leq r \leq 255$)

FOX128 and FOX128/k/r encryption schemes iterate $r-1$ (full) rounds, denoted **elmor128**, followed by a final round denoted **elmid128**. Similarly, a modified round function **elmio128** is used for decryption. Formally, **elmor128**, **elmid128**, and **elmio128**: $\mathbb{Z}_2^{128} \times \mathbb{Z}_2^{128} \rightarrow \mathbb{Z}_2^{128}$. The encryption of a 128-bit block P under a 128-bit key K results in a 128-bit ciphertext block C as follows: $C =$ **elmid128**(**elmor128**(... **elmor128**(P, RK_0), ..., RK_{r-2}), RK_{r-1}), where RK_i, $0 \leq i \leq r-1$ are 128-bit subkeys derived from K using a key schedule algorithm. Analogously, the decryption of a data block C, given a key K, results in the plaintext block P given by $P =$ **elmid128**(**elmio128**(... **elmio128** (C, RK_{r-1}), ..., RK_1), RK_0).

The **elmor64** round function is built as a Lai-Massey scheme combined with an orthomorphism mapping **or**:$\mathbb{Z}_2^{32} \rightarrow \mathbb{Z}_2^{32}$, defined as **or**$(a,b) = (b, a \oplus b)$, where $a, b \in \mathbb{Z}_2^{16}$, and $\oplus$ is bitwise exclusive-or. Formally, $Y = Y_L || Y_R =$ **lmor64**$(X_L || X_R) =$ **or**$(X_L \oplus f32(X_L \oplus X_R, RK_i)) || (X_R \oplus f32(X_L \oplus X_R, RK_i))$, where $||$ denotes concatenation of bit strings. The **lmid64** function is the same as **lmor64** but without **or**. The inverse of **or** is denoted **io**: $\mathbb{Z}_2^{32} \rightarrow \mathbb{Z}_2^{32}$,and defined as **io**$(a,b) = (a \oplus b, a)$. The mapping f32: $\mathbb{Z}_2^{32} \times \mathbb{Z}_2^{64} \rightarrow \mathbb{Z}_2^{32}$ is bijective, taking a 32-bit data, X and a 64-bit round subkey, $RK_i = RK_{0i} || RK_{1i}$ as inputs and returning a 32-bit output, $Y = f32(X, RK_i) = sigma4(mu4(sigma4(X \oplus RK_{0i})) \oplus RK_{1i}) \oplus RK_{0i}$. The transformation *sigma*4 consists of parallel application of a fixed 8×8-bit S-box, denoted S, bytewise. Its inverse is denoted simply S^{-1}. The linear transformation *mu*4 consists of a multiplication of the intermediate data with a 4×4 matrix over GF(2^8). Given a 32-bit input $A = (A_0, A_1, A_2, A_3)$ to *mu*4, its output is $B = (B_0, B_1, B_2, B_3)$:

$$\begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & x \\ 1 & x^{-1}+1 & x & 1 \\ x^{-1}+1 & x & 1 & 1 \\ x & 1 & x^{-1}+1 & 1 \end{pmatrix} \cdot \begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \end{pmatrix}$$

Previously known analysis of reduced-round FOX ciphers were reported by Junod in [Junod and Vaudenay 2004], and in [WEWoRC 2005].

Some noteworthy properties of the orthomorphism **or** (and its inverse **io**) are:

- **or** has order 3, namely **or**(**or**(**or**(a,b))) = **or**$^3(a,b) = (a,b)$; the same applies to **io**;
- $(a,b) \oplus$ **or**$(a,b) \oplus$ **or**$^2(a,b) = (0,0)$, for any $(a,b) \in \mathbb{Z}_2^{16} \times \mathbb{Z}_2^{16}$;
- **or**$(a,b) = (a,b) \Leftrightarrow (a,b) = (0,0)$;
- **or**$^2(a,b) = (a,b) \Leftrightarrow (a,b) = (0,0)$;
- **or**$(a,b) =$ **io**$(a,b) \Leftrightarrow (a,b) = (0,0)$.

The orthomorphism is essential in the round structure of FOX. Consider FOX64/k/r, but without the orthomorphism mapping. Let the plaintext be denoted $P = (P_L, P_R)$, and the ciphertext, $C = (C_L, C_R)$, and consider the exclusive-or of the two 32-bit words in a block, at the input and output after r rounds. Since the output of every f32 instance is exclusive-ored to both words in a block at the end of every round, it follows that $P_L \oplus P_R = C_L \oplus C_R$, that is, the f32 values are canceled in pairs. This invariant allows to distinguish this weak FOX64/k/r variant from a random permutation, for any $r > 0$ by simply comparing the xor of input words with the xor of output words (a 32-bit distinguisher that would hold with probability 2^{-32} in a random permutation). Similarly, in FOX128/k/r without the orthomorphism, the exclusive-or of all four 32-bit words of plaintext and ciphertext blocks are also equal, for any $r > 0$.

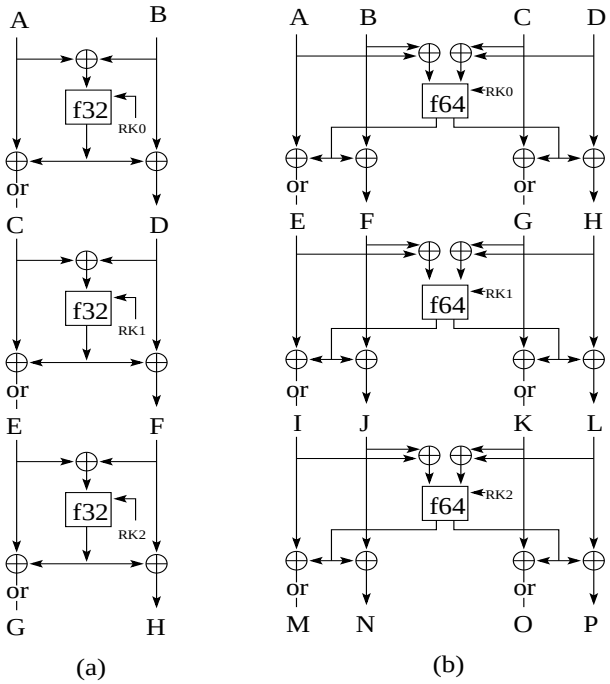

Figure 1. (a) 3-round FOX64/k/r, and (b) 3-round FOX128/k/r.

This paper is organized as follows: Sect. 2. presents an attack on 2-round FOX64/k/r requiring only two known plaintext-ciphertext pairs. Sect. 3. describes non-trivial information leaked from r-round FOX64/k/r, for $r > 2$. Sect. 4. presents an attack on 2-round FOX128/k/r. Sect. 5. describes non-trivial information leaked from r-round FOX128/k/r, for $r > 2$. Sect. 6. describes impossible differential attacks on reduced-round FOX. Sect. 7. concludes the paper.

2. Key-recovery Attack on 2-Round FOX64/k/r

A consequence of the Lai-Massey scheme, in both FOX64/k/r and FOX128/k/r, is that the input of the first and of the last round fx function, $x \in \{32, 64\}$, is always known from the plaintext and the ciphertext blocks. For example, in Fig. 1(a), $A \oplus B = \mathbf{io}(C) \oplus D$, and $E \oplus F = \mathbf{io}(G) \oplus H$. This is also a consequence of the absence of key-dependent pre-whitening and post-whitening, such as in the DES [NBS 1977].

Consider FOX64/k/r with two full rounds, namely, $Y =$ **elmor64**(**elmor64** (X, RK_0), RK_1). There is a straightforward attack that requires only two known plaintext-ciphertext (KP) pairs. Let the first KP pair be (X_1, Y_1), with $X_1 = (A||B)$, and $Y_1 = (E||F)$; and the output of the first round be $(C||D)$, with $A, B, C, D, E, F \in \mathbb{Z}_2^{16}$ (the first 2 rounds in Fig. 1(a)). Thus, from the round structure (or Lai-Massey scheme) of FOX64/k/r, it follows that

$$A \oplus B = \mathbf{io}(C) \oplus D \tag{1}$$

and

$$C \oplus D = \mathbf{io}(E) \oplus F\,. \tag{2}$$

From (1) and (2), we obtain:

$$\mathbf{io}(E) \oplus F = \mathbf{or}(A \oplus B \oplus D) \oplus D \rightarrow D \oplus \mathbf{or}(D) = \mathbf{or}(A \oplus B) \oplus \mathbf{io}(E) \oplus F\,, \tag{3}$$

and

$$\mathbf{io}(E) \oplus F = C \oplus A \oplus B \oplus \mathbf{io}(C) \rightarrow C \oplus \mathbf{io}(C) = A \oplus B \oplus \mathbf{io}(E) \oplus F\,. \tag{4}$$

If $D = (d_1||d_2)$, then $\mathbf{or}(D) \oplus D = (d_1 \oplus d_2)||(d_1)$. It follows that $\mathbf{or}(D) \oplus D$ uniquely determines D. Similarly, for $\mathbf{io}(C) \oplus C$. Thus, information on the plaintext $(A||B)$, and the ciphertext $(E||F)$ allows one to derive information from the middle of 2-round FOX64/k/r.

Given C and D, it is possible to attack both f32 functions, since both their input and output are known. For example, the input to the f32 in the first round is $A \oplus B$, and its output if $\mathbf{io}(C) \oplus A$ or $B \oplus D$. These values represent a 32-bit distinguisher: $\text{f32}(A \oplus B) = B \oplus D$. Given these values, and the description of f32 in Sect. 1., one can guess the full 32-bit RK_{00}, and obtain a unique value for the full 32-bit RK_{10}. This value can be verified using a second plaintext pair $(A'||B')$. There is a 2^{-32} chance of a false alarm. Using another plaintext-ciphertext pair the number of remaining false keys is $(2^{64} - 1) \cdot (2^{-32})^2 < 1$. Analogously for the f32 in the second round, its input is $C \oplus D$, and its output is $\mathbf{io}(E) \oplus F$. The total attack effort is $2 \cdot 2^{32}$ f32 evaluations, or about 2^{32} 2-round FOX64/k/r computations, and 2 KP pairs, to recover both round subkeys (128 bits). This attack works independently of the key schedule algorithm (moreover, it does not apply to 2-round IDEA [Lai 1995]). Thus, it holds for both FOX64 and FOX64/k/r.

3. Information Leakage from *r*-round FOX64/k/r, $r > 2$

In this section, we derive internal cipher data from FOX64/k/r, for $r > 2$ using only known plaintext. As an example, consider $r = 3$. Let, $Y = \mathbf{elmor64}(\,\mathbf{elmor64}(\,\mathbf{elmor64}(\,X,\, RK_0),\, RK_1),\, RK_2)$. Applying a similar reasoning as in Sect.2., let the plaintext be denoted $X = (A||B)$, the output of the first round be $(C||D)$, the output of the second round be $(E||F)$ and the ciphertext be $Y = (G||H)$ (Fig. 1(a)). Then, the same relations (1) and (2) can be derived, plus the following

$$\mathbf{io}(G) \oplus H = E \oplus F\,. \tag{5}$$

Combining (1), (2), and (5), results in the following relations depending only on plaintext $(A||B)$ and ciphertext $(G||H)$. For example, from (5), it follows that

$$\begin{gathered}\mathbf{io}(G) \oplus H = E \oplus F = C \oplus D \oplus \mathbf{io}(E) \oplus E = \mathbf{or}(A \oplus B \oplus D) \oplus D \oplus \mathbf{io}(E) \oplus E \rightarrow \\ D \oplus \mathbf{or}(D) \oplus E \oplus \mathbf{io}(E) = \mathbf{or}(A \oplus B) \oplus \mathbf{io}(G) \oplus H \rightarrow \\ E \oplus \mathbf{or}(D) \oplus \mathbf{or}(E \oplus \mathbf{or}(D)) = \mathbf{or}^2(A \oplus B) \oplus G \oplus \mathbf{or}(H)\,.\end{gathered} \tag{6}$$

Thus, $E \oplus \mathbf{or}(D)$ can be uniquely determined. Similarly, the following relations can be obtained

$$C \oplus E \oplus \mathbf{io}(C \oplus E) = A \oplus B \oplus \mathbf{io}(G) \oplus H\,, \tag{7}$$

$$F \oplus \mathbf{or}(D) \oplus \mathbf{or}(F \oplus \mathbf{or}(D)) = \mathbf{or}^2(A \oplus B) \oplus \mathbf{io}(G) \oplus H\,, \tag{8}$$

and

$$F \oplus C \oplus \mathbf{or}(F \oplus C) = \mathbf{or}(A \oplus B) \oplus \mathbf{io}(G) \oplus H\,, \tag{9}$$

which provide $F \oplus \mathbf{or}(D)$ and $F \oplus C$. All of these relations leak internal information from 3-round FOX64/k/r, given only known plaintext. This phenomenon is absent in IDEA [Lai et al. 1991].

The same reasoning can be applied to r-round FOX64/k/r, $r > 3$. The information leaked refers to 32-bit exclusive-or combination of intermediate cipher data across r rounds of FOX64/k/r, using only known plaintext.

Another approach at deriving non-trivial internal cipher data from FOX64/k/r is the following: consider $r = 3$, a plaintext block $X = (A||B)$, and the corresponding ciphertext block $Y = (G||H)$, with $A = (a_1||a_2)$, $B = (b_1||b_2)$, $G = (g_1||g_2)$, and $H = (h_1||h_2)$, a_i, b_i, c_i, $d_i \in \mathbb{Z}_2^8$, $1 \leq i \leq 2$. Moreover, let the 32-bit outputs of f32 functions in the 1st, 2nd and 3rd rounds be denoted $(x_1||x_2)$, $(y_1||y_2)$ and $(z_1||z_2)$, respectively. Using the round structure of FOX64/k/r, it follows from B and G that

$$B \oplus H = (x_1 \oplus y_1 \oplus z_1)||(x_2 \oplus y_2 \oplus z_2)\,. \tag{10}$$

Similarly, from A and G: $G = \mathbf{or}((z_1||z_2) \oplus \mathbf{or}((y_1||y_2) \oplus \mathbf{or}((x_1||x_2) \oplus A)))$. It results in

$$A \oplus G = (x_1 \oplus y_1 \oplus y_2 \oplus z_2)||(x_2 \oplus y_1 \oplus z_1 \oplus z_2)\,. \tag{11}$$

Equations (10) and (11) provide internal data of 3-round FOX64/k/r, distict from (6), (7), (8), (9). We can further isolate some values in (10) and (11). Let $B \oplus G = \alpha = (\alpha_1||\alpha_2)$, and $A \oplus G = \beta = (\beta_2||\beta_2)$. Then, $\alpha_1 \oplus \beta_1 \oplus \beta_2 = x_2 \oplus z_1$; $\alpha_1 \oplus \alpha_2 \oplus \beta_1 \oplus \beta_2 = y_1 \oplus z_2$; and $\alpha_2 \oplus \beta_1 \oplus \beta_2 = x_1 \oplus y_2$.

Relations (10) and (11) are similar to the exclusive-or combination of round output values observed in the DES cipher in [Davies and Murphy 1995]. If the f32 function was non-bijective for a fixed key, then the knowledge of the exclusive-or of several outputs of f32 functions across several rounds could allow to detect some non-uniform distribution, as in [Rijmen et al. 1997]. But, the f32 mapping is surjective.

4. An Attack on 2-Round FOX128/k/r

A similar attack to 2-round FOX64/k/r in Sect.2. can be applied to 2-round FOX128/k/r. Consider FOX128/k/2, namely, $Y =$ **elmor128**(**elmor128** $(X, RK_0), RK_1)$. Let $(X_1||Y_1)$ be a KP pair, with $X_1 = (A||B||C||D)$, and $Y_1 = (I||J||K||L)$, and $(E||F||G||H)$ be the output of the first round, with $A, B, C, D, E, F, G, H, I, J, K, L \in \mathbb{Z}_2^{16}$ (the first 2 rounds in Fig. 1(b)). From the round structure (or Lai-Massey scheme) of FOX128/k/r, it follows that

$$A \oplus B = \mathbf{io}(E) \oplus F\,, \tag{12}$$

$$C \oplus D = \mathbf{io}(G) \oplus H\,, \tag{13}$$

$$E \oplus F = \mathbf{io}(I) \oplus J\,, \tag{14}$$

$$G \oplus H = \mathbf{io}(K) \oplus L\,. \tag{15}$$

From (12) and (14), we obtain:

$$\mathbf{io}(I) \oplus J = E \oplus F = \mathbf{or}(A \oplus B \oplus F) \oplus F \rightarrow F \oplus \mathbf{or}(F) = \mathbf{or}(A \oplus B) \oplus \mathbf{io}(I) \oplus J\,, \tag{16}$$

so, F can be uniquely determined from $F \oplus \mathbf{or}(F)$. Analogously, we obtain

$$\mathbf{io}(I) \oplus J = E \oplus F = E \oplus A \oplus B \oplus \mathbf{io}(E) \rightarrow E \oplus \mathbf{io}(E) = A \oplus B \oplus \mathbf{io}(I) \oplus J\,, \tag{17}$$

and E can be uniquely determined from $E \oplus \mathbf{io}(E)$.

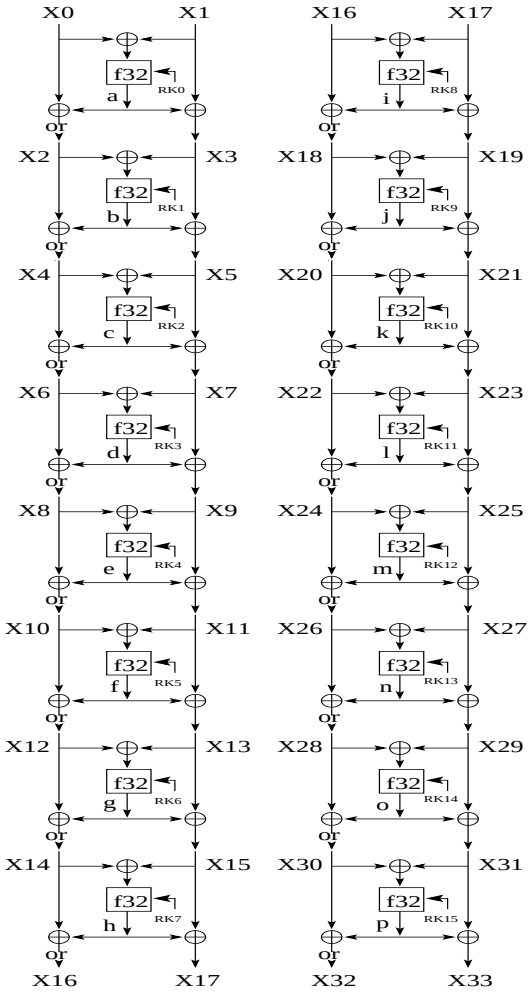

Figure 2. Computational graph of full 16-round FOX64.

Similarly, from (13) and (15), we obtain

$$\mathbf{io}(K) \oplus L = G \oplus H = \mathbf{or}(C \oplus D \oplus H) \oplus H \rightarrow H \oplus \mathbf{or}(H) = \mathbf{or}(C \oplus D) \oplus \mathbf{io}(K) \oplus L\,, \quad (18)$$

so, H can be uniquely determined from $H \oplus \mathbf{or}(H)$. Analogously, we obtain

$$\mathbf{io}(K) \oplus L = G \oplus H = G \oplus C \oplus D \oplus \mathbf{io}(G) \rightarrow G \oplus \mathbf{io}(G) = C \oplus D \oplus \mathbf{io}(K) \oplus L\,, \quad (19)$$

and G can be uniquely determined from $G \oplus \mathbf{io}(G)$.

Now, with the recovered values of E, F, G, and H, we can attack both f64 functions, since we know both their inputs and outputs. For example, the f64 in the first round has input $(A \oplus B)||(C \oplus D)$, and output $(B \oplus F)||(D \oplus H)$. It is a 64-bit distinguisher (it holds with probability 2^{-64} for a random permutation). From the internal structure of f64, on can guess the 64-bit RK_{00}, and recover a unique 64-bit value for RK_{01}. This value can be checked with another KP pair. The number of false alarms after this filtering is $(2^{128} - 1) \cdot (2^{-64})^2 < 1$. The same applies to f64 in the second round, whose input is $(E \oplus F)||(G \oplus H)$ and output $(F \oplus J)||(H \oplus L)$. The same two KP pairs can be reused to recover RK_{10} and RK_{11}. The total attack effort is $2 \cdot 2^{64}$ f64 computations, or about 2^{64} 2-round FOX128/k/r computations, and 2 KP, to recover two 128-bit subkeys. This attack is independent of the key schedule algorithm.

5. Information Leakage from *r*-round FOX128/k/r, $r > 2$

Similar to Sect.3., one can derive nontrivial information from r-round FOX128/k/r for $r > 2$. As an example, we detail the case $r = 3$. Consider the following relations from Fig. 1(b):

$$I \oplus J = \mathbf{io}(M) \oplus N\,, \quad (20)$$

$$K \oplus L = \mathbf{io}(O) \oplus P\,. \quad (21)$$

Equations (12) up to (21) lead to the following relation $I \oplus J = \mathbf{io}(M) \oplus N = \mathbf{or}(E \oplus F \oplus J) \oplus J = \mathbf{or}(\mathbf{or}(A \oplus B \oplus F) \oplus F \oplus J) \oplus J \rightarrow \mathbf{or}^2(F) \oplus \mathbf{or}(F) \oplus J \oplus \mathbf{or}(J) = \mathbf{or}^2(A \oplus B) \oplus \mathbf{io}(M) \oplus N$. Thus, $J \oplus \mathbf{or}(F)$ can be uniquely determined from $(A||B)$ and $(M||N)$. Similarly, the following relations can be obtained: $I \oplus J = \mathbf{io}(M) \oplus N = \mathbf{or}(E \oplus F \oplus J) \oplus J = \mathbf{or}(E \oplus A \oplus B \oplus \mathbf{io}(E) \oplus J) \oplus J \rightarrow \mathbf{or}(E \oplus J) \oplus E \oplus J = \mathbf{or}(A \oplus B) \oplus \mathbf{io}(M) \oplus N$, uniquely determining $E \oplus J$; also, $I \oplus J = \mathbf{io}(M) \oplus N = I \oplus E \oplus F \oplus \mathbf{io}(I) = I \oplus \mathbf{or}(A \oplus B \oplus F) \oplus F \oplus \mathbf{io}(I) \rightarrow F \oplus \mathbf{or}(F) \oplus I \oplus \mathbf{io}(I) = \mathbf{or}(A \oplus B) \oplus \mathbf{io}(M) \oplus N$, uniquely determining $I \oplus \mathbf{or}(F)$; and finally, $I \oplus J = \mathbf{io}(M) \oplus N = I \oplus E \oplus F \oplus \mathbf{io}(I) = I \oplus E \oplus A \oplus B \oplus \mathbf{io}(E \oplus I) \rightarrow I \oplus E \oplus bfio(I \oplus E) = A \oplus B \oplus \mathbf{io}(M) \oplus N$, uniquely determining $I \oplus E$.

Further internal data leaking from FOX128/k/r come from $D \oplus P$ and $B \oplus N$. Let $(x_1||x_2)$, $(y_1||y_2)$, $(z_1||z_2)$ be the 64-bit outputs from the f64 functions in the 1st, 2nd and 3rd rounds, respectively. Then, $D \oplus P = x_2 \oplus y_2 \oplus z_2$, and $N = \mathbf{or}(z_1 \oplus \mathbf{or}(y_1 \oplus \mathbf{or}(x_1 \oplus B))) \rightarrow \mathbf{or}(z_1) \oplus \mathbf{or}^2(y_1) \oplus x_1 = B \oplus N$. Similar values can be obtained for r-round FOX128/k/r, for any $r > 2$.

6. Impossible-Differentials of reduced-round FOX

The impossible-differential (ID) technique applied to FOX follows a similar approach as used against IDEA in [Biham et al. 1999]. This analysis considers truncated differentials. Thus, let Δ denote an arbitrary nonzero 32-bit xor difference, and 0 a null 32-bit xor difference. The exact value of Δ does not matter, but only the fact that it is the exclusive-or of two distinct values.

6.1. ID Distinguishers of FOX64

We have found several 3-round ID distinguishers for FOX64, using truncated differentials. For instance, consider a plaintext difference of $(\Delta, \Delta, \Delta, \Delta)$, and an output difference of $(\Delta, \Delta, 0, \Delta)$ after three full rounds. At the first round, the input difference to f32 is $(0, 0)$. Consequently, the round output difference is $(\Delta, 0, \Delta, \Delta)$. The input difference of the second f32 is $(0, \Delta)$. The third round output difference, $(\Delta, \Delta, 0, \Delta)$, leads to a difference of $(0, \Delta, 0, \Delta)$ before **or**, and an input difference of $(0, 0)$ to the third f32 mapping. Thus, the output difference of the second round is $(0, \Delta, 0, \Delta)$. Consequently, the output difference of the second f32 has to be $(\Delta, 0) \oplus (\Delta, 0) = (0, 0)$ (due to the left 32-bit output half) and also $(0, \Delta) \oplus (\Delta, \Delta) = (\Delta, 0)$ (due to the right 32-bit output half). This is a contradiction because f32 is a bijective mapping, and the input difference, as noted before, was nonzero, that is $(0, \Delta)$.

Thus, the difference $(\Delta, \Delta, \Delta, \Delta)$ cannot cause the difference $(\Delta, \Delta, 0, \Delta)$ after 3-round FOX64.

This ID distinguisher is denoted $(\Delta, \Delta, \Delta, \Delta) \overset{3rounds}{\nrightarrow} (\Delta, \Delta, 0, \Delta)$.

6.2. ID Attacks on reduced-round FOX-64

First, we describe an attack on 4-round FOX64, that recovers RK_0, the first round subkey, on top of the 3-round distinguisher of Sect. 6.1.. Choose a pool of $2^{35.5}$ plaintexts, denoted $P = (P_1, P_2, P_3, P_4)$, with $|P_i| = 16$ bits, which means $2^{35.5}(2^{35.5} - 1)/2 \approx 2^{70}$ text pairs. Collect about $2^{70}/(2^{16})^3 = 2^{22}$ pairs whose ciphertexts $C = (C_1, C_2, C_3, C_4)$ satisfy $\Delta C_1 = \Delta C_2 = \Delta C_4 \neq 0$ and $\Delta C_3 = 0$. For each such pair try all 2^{64} possible subkeys of the first RK_0 of the first f32 function, such that the round output difference is $(\Delta_1, \Delta_2, \Delta_3, \Delta_4)$, by partially encrypting one round. Collect about $2^{64}/(2^{16})^3 = 2^{16}$ subkey values satisfying $\Delta_1 = \Delta_2 = \Delta_3 = \Delta_4$. These subkeys are wrong because they lead to a pair that satisfies the ID distinguisher of Sect. 6.1.. Repeat the same analyses for each of the 2^{22} pairs. Each pair defines about 2^{16} wrong values. It is expected that after 2^{22} pairs, the number of wrong subkeys remaining is $2^{64}(1 - 2^{-16})^{2^{22}} = 2^{64}(1 - 2^{-16})^{2^{16}.2^6} \approx 2^{64} * e^{-64} < 1$, and the single correct RK_0 can be uniquely identified. The attack costs $2^{35.5}$ CP; $2^{64} * 2^{22}/4 = 2^{84}$ 4-round computations and about $2^{64}/64 = 2^{58}$ blocks of memory.

Next, we describe an attack on 5-round FOX64, recovering RK_0 and RK_4, both at the top and at the bottom of the distinguisher described in Sect. 6.1..

Choose a pool of 2^{36} plaintexts, denoted $P = (P_1, P_2, P_3, P_4)$, with $|P_i| = 16$ bits. This pool provides about 2^{71} text pairs. For each such pair:

(i) try all of the 2^{64} possible subkeys RK_0 of the first round, and partially encrypt each plaintext pair across one round. Keep those pairs leading to a difference $(\Delta_1, \Delta_2, \Delta_3, \Delta_4) \neq (0, 0, 0, 0)$ after one round. Collect about $2^{64}/(2^{16})^3 = 2^{16}$ 64-bit subkey values that lead to $\Delta_1 = \Delta_2 = \Delta_3 = \Delta_4$.

(ii) for each pair in (ii), try all 2^{64} subkey values RK_4 of the 5th round, and partially decrypt the last round for each text block of the pair. Keep the pairs that have difference $(\Delta^*, \Delta^{**}, 0, \Delta^{***})$. The number of pairs is a fraction of 2^{-16} due to the 16-bit zero difference: $2^{71}/2^{16} = 2^{55}$. Collect about $2^{64}/(2^{16})^2 = 2^{32}$ 64-bit subkey candidates for RK_4 that lead to $\Delta^* = \Delta^{**} = \Delta^{***}$.

Make a list of the 2^{48} 128-bit subkey values combining (i) and (ii). These subkeys are wrong because they lead to a pair that satisfies the ID distinguisher. Each such pair defines

a list of about 2^{48} wrong 128-bit subkey. The number of remaining wrong subkeys after 2^{55} pairs is $2^{128}(1-2^{-48})^{2^{55}} = 2^{128}(1-2^{-48})^{2^{48}*2^7} \approx 2^{128} * e^{-128} < 1$. Thus, only the correct subkey value is expected to survive.

The attack costs 2^{36} CP; $2^{55} * 2^{64} * 2/5 \approx 2^{118}$ 5-round computations and $2^{128}/64 = 2^{122}$ blocks of memory.

6.3. ID Distinguishers of FOX128

There are several ID distinguishers for FOX128, following a similar pattern to those found for FOX64 in Sect. 6.1.. Consider 3-round FOX128, and a plaintext difference (Δ, Δ, Δ, Δ, Δ, Δ, Δ, Δ), where $|\Delta| = 16$ bits, and $\Delta \neq 0$. Tracing the propagation of this difference, the input difference to the first f64 function is $(0,0,0,0)$, and thus, its output difference is also $(0,0,0,0)$. Consequently, the first round output difference is (Δ, 0, Δ, Δ, Δ, 0, Δ, Δ), due to the two **or** mappings. This value is also the input difference to the second round, while the input difference to the second f64 is $(0,0,0,0)$.

Now, consider the ciphertext difference (Δ', Δ', 0, Δ', Δ', Δ', 0, Δ') after three rounds, where $\Delta' \neq 0$. Tracing the difference propagation upwards: before the **or** mappings, the difference becomes (0, Δ', 0, Δ', 0, Δ', 0, Δ') and thus, the input difference to the third f64 function is $(0,0,0,0)$, and the input difference to the third round is also (0, Δ', 0, Δ', 0, Δ', 0, Δ').

Before the second layer of **or** mappings, at the end of the second round, the difference becomes (Δ', 0, 0, Δ', Δ', 0, 0, Δ'). Combining the input and output differences of the second round, we conclude that the left 32-bit output difference of f64 is $(\Delta,0) \oplus (\Delta',0)$ and also $(0,\Delta) \oplus (0,\Delta')$, that is, $(\Delta \oplus \Delta', 0) = (0, \Delta \oplus \Delta')$, that is, $\Delta = \Delta'$. On the other hand, the right 32-bit output difference from f64 is $(\Delta,0) \oplus (\Delta',0)$ and also $(\Delta,\Delta) \oplus (0,\Delta')$, that is, $(\Delta \oplus \Delta', 0) = (\Delta, \Delta \oplus \Delta')$ i.e. $\Delta' = 0$, But, this last equality contradicts the fact that $\Delta' \neq 0$, by construction.

Thus, the difference (Δ, Δ, Δ, Δ, Δ, Δ, Δ, Δ) cannot cause the difference (Δ', Δ', 0, Δ', Δ', Δ', 0, Δ') after 3-round FOX128. This distinguisher is denoted (Δ, Δ, Δ, Δ, Δ, Δ, Δ, Δ) $\overset{3rounds}{\nrightarrow}$ (Δ', Δ', 0, Δ', Δ', Δ', 0, Δ'). Note that the contradiction works in the decryption direction, too.

6.4. ID Attacks on reduced-round FOX-128

First, we describe an attack on 4-round FOX128, recovering RK_0, similar to the attack on FOX64, but using the distinguisher in Sect. 6.3..

Choose a pool of 2^{68} plaintexts, $P = (P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8)$, $|P_i| = 16$ bits, which gives about 2^{135} text pairs. Collect about $2^{135}/(2^{16})^7 = 2^{135-112} = 2^{23}$ pairs whose ciphertext (denoted $C = (C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8)$) pairs satisfy $\Delta C_1 = \Delta C_2 = \Delta C_4 = \Delta C_5 = \Delta C_6 = \Delta C_8 \neq 0$, and $\Delta C_3 = \Delta C_7 = 0$. For each such pair, try all 2^{128} subkey value for RK_0 in the first f64 function, such that the first round output difference is $(\Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_5, \Delta_6, \Delta_7, \Delta_8)$ by partially encrypting one round. Collect about $2^{128}/(2^{16})^7 = 2^{16}$ subkey values satisfying $\Delta_1 = \Delta_2 = \Delta_3 = \Delta_4 = \Delta_5 = \Delta_6 = \Delta_7 = \Delta_8$. These subkeys are wrong because they lead to a pair that satifies the ID distinguisher of Sect. 6.3..

Each text pair defines about 2^{16} wrong subkeys. It is expected that after 2^{23} pairs, the number of wrong subkeys remaining is $2^{128}(1-2^{-16})^{2^{23}} = 2^{128}(1-2^{-16})^{2^{16}*2^7} \approx 2^{128} * e^{-128} < 1$, and the correct RK_0 can be uniquely identified.

The attack costs 2^{68} CP; $2^{128} * 2^{23}/4 = 2^{149}$ 4-round computations and about $2^{128}/128 = 2^{121}$ blocks of memory.

Next, an attack on 5-round FOX128, recovering both RK_0 and RK_4, using the distinguisher of Sect. 6.3.. It is similar to the attack on 5-round FOX64.

Choose a pool of 2^{52} plaintexts, denoted $P = (P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8)$, with $|P_i|$= 16 bits. This pool provides about 2^{103} text pairs. For each such pair:

(i) try all of the 2^{128} possible subkeys RK_0 of the first round, and partially encrypt each plaintext pair across one round. Keep those pairs leading to a difference (Δ_1, Δ_2, Δ_3, Δ_4, Δ_5, Δ_6, Δ_7, Δ_8) $\neq$ (0, 0, 0, 0, 0, 0, 0, 0) after one round. Collect about $2^{128}/(2^{16})^7 = 2^{16}$ 128-bit subkey values that lead to $\Delta_1 = \Delta_2 = \Delta_3 = \Delta_4 = \Delta_5 = \Delta_6 = \Delta_7 = \Delta_8$.

(ii) for each pair in (ii), try all 2^{128} subkey values RK_4 of the 5th round, and partially decrypt the last round for each text block of the pair. Keep the pairs that have difference (Δ_1, Δ_2, 0, Δ_3, Δ_4, Δ_5, 0, Δ_6). The number of pairs is a fraction of 2^{-32} due to the two 16-bit zero difference words: $2^{103}/2^{32} = 2^{71}$. Collect about $2^{128}/(2^{16})^5 = 2^{48}$ 128-bit subkey candidates for RK_4 that lead to $\Delta_1 = \Delta_2 = \Delta_3 = \Delta_4 = \Delta_5 = \Delta_6$.

Make a list of the 2^{64} 128-bit subkey values combining (i) and (ii). These subkeys are wrong because they lead to a pair that satisfies the ID distinguisher. Each such pair defines a list of about 2^{64} wrong 128-bit subkeys. The number of remaining wrong subkeys after 2^{71} pairs is $2^{128}(1-2^{-64})^{2^{71}} = 2^{128}(1-2^{-64})^{2^{64}*2^7} \approx 2^{128} * e^{-128} < 1$. Thus, only the single correct subkey value is expected to survive.

The attack costs 2^{52} CP; $2 * 2^{128} * 2^{71}/5 \approx 2^{198}$ 5-round computations and $2^{256}/128 = 2^{249}$ blocks of memory.

7. Conclusions

This paper presented ID attacks on 4-round and 5-round FOX ciphers, and some findings on information leakage on 2-round variants. We could derive the full subkeys of FOX64/k/2 and FOX128/k/2 using only two known plaintexts. The attacks described have the same effectiveness for either the encryption or the decryption schemes and is independent of the key schedule algorithms. Table 2 compare the complexity of known attacks reported on reduced-round FOX ciphers.

Furthermore, we could derive non-trivial internal cipher data from r-round FOX ciphers, for $r > 2$. This data leakage has already been observed in DES [NBS 1977, Davies and Murphy 1995], but not in IDEA, whose design inspired the FOX ciphers.

The information leakage detected in r-round FOX ciphers, $r > 2$, currently does not lead to attacks such as [Davies and Murphy 1995, Rijmen et al. 1997] since the round functions, f32 and f64, are both bijective mappings. It is an open question how to exploit this information leakage to further recover key bits, or other unknown text data.

A subject for further studies is the existence of dual FOX ciphers, as observed in the AES [Barkan and Biham 2002], since all internal operations in FOX are over GF(2^8).

Table 2. Attack complexities on reduced-round FOX ciphers.

Cipher	#Rounds	Time	Data	Memory	Source
FOX64/k/r	2	2^{32}	2 KP	—	Sect. 2.
	4	2^{41}	2^{40} CP	2^{34}	[WEWoRC 2005, p.98]
	4	$2^{45.4}$	2^{9} CP	2^{9}	[Wenling et al. 2005]
	4	2^{64}	$9 \cdot 2^{8}$ CP	2^{8}	[Junod and Vaudenay 2004]
	4	2^{84}	$2^{35.5}$ CP	2^{58}	Sect. 2.
	5	2^{105}	2^{40} CP	2^{46}	[WEWoRC 2005, p.98]
	5	$2^{109.4}$	2^{9} CP	2^{9}	[Wenling et al. 2005]
	5	2^{118}	2^{36} CP	2^{122}	Sect. 6.2.
	5	2^{128}	$17 \cdot 2^{8}$ CP	2^{8}	[Junod and Vaudenay 2004]
	6	$2^{173.4}$	2^{9} CP	2^{9}	[Wenling et al. 2005]
	6	2^{192}	$25 \cdot 2^{8}$ CP	2^{8}	[Junod and Vaudenay 2004]
FOX128/k/r	2	2^{64}	2 KP	—	Sect. 4.
	4	$2^{77.6}$	2^{9} CP	2^{9}	[Wenling et al. 2005]
	4	2^{128}	2^{40} CP	—	[Junod and Vaudenay 2004]
	4	2^{149}	2^{68} CP	2^{121}	Sect. 6.4.
	5	2^{198}	2^{52} CP	2^{249}	Sect. 6.4.
	5	2^{201}	2^{43} CP	2^{79}	[WEWoRC 2005, p.98]
	5	$2^{205.6}$	2^{9} CP	2^{9}	[Wenling et al. 2005]

References

Barkan, E. and Biham, E. (2002). In how many ways can you write Rijndael. In Zheng, Y., editor, *Adv. in Cryptology, Asiacrypt 2002*, LNCS 2501, pages 160–175. Springer-Verlag.

Biham, E., Biryukov, A., and Shamir, A. (1999). Miss-in-the-Middle Attacks on IDEA, Khufu and Khafre. In Knudsen, L., editor, *6th Fast Software Encryption Workshop*, LNCS 1636, pages 124–138. Springer-Verlag.

Davies, D. and Murphy, S. (1995). Pairs and Triplets of DES S-Boxes. *Journal of Cryptology*, 8(1):1–25.

Junod, P. and Vaudenay, S. (2004). FOX: a new family of block ciphers. In *11th Selected Areas in Cryptography (SAC) Workshop*, LNCS 3357, pages 114–129. Springer-Verlag.

Lai, X. (1995). *On the Design and Security of Block Ciphers*, volume 1 of *ETH Series in Information Processing*. Hartung-Gorre Verlag, Konstanz. J.L. Massey.

Lai, X., Massey, J., and Murphy, S. (1991). Markov Ciphers and Differential Cryptanalysis. In Davies, D., editor, *Advances in Cryptology, Eurocrypt'91*, LNCS 547, pages 17–38. Springer-Verlag.

NBS (1977). Data Encryption Standard (DES). FIPS PUB 46, Federal Information Processing Standards Publication 46, U.S. Department of Commerce.

Rijmen, V., Preneel, B., and Win, E. D. (1997). On weaknesses of non-surjective round functions. *Design, Codes and Cryptography*, 12(3):253–266.

Wenling, W., Wentao, Z., and Dengguo, F. (2005). Improved integral cryptanalysis of FOX block cipher.

WEWoRC (2005). Western European Workshop on Research in Cryptology.