ブロック暗号アルゴリズムCLEFIAの11段96階差分攻撃の高速化

五十嵐保隆[1]　金子敏信[2]　橋口陽介[1]　江口悠[1]　末吉隆太郎[1]　村井貴広[1]　福島誠治[1]　八野知博[1]

概要：CLEFIA は 2007 年に SONY の白井らが提案したブロック暗号アルゴリズムである。ブロック長は 128 ビットであり、鍵長は 128、192、256 ビットがサポートされている。データ攪拌部の段数は鍵長によって異なり、鍵長が 128、192、256 ビットの場合それぞれ 18 段、22 段、26 段となっている。これまでに、CLEFIA の 8 段目出力 128 ビット中の 64 ビットについては、その 96 階差分がゼロとなる特性が知られており、我々はこの特性を利用した CLEFIA の 11 段 96 階差分攻撃を報告している。この攻撃には選択平文数 $2^{98.3}$、暗号化計算量 2^{159} を要する。本稿では Ferguson らが提案した部分和法を用いて、解読時の中間データの mod2 頻度分布表を逐次導出することにより攻撃に要する計算量を削減する。さらに攻撃方程式の計算過程において、繰り返し計算ループの入れ子構造を採用し、入れ子の順序を適切に設定することにより計算量を削減し、11 段 96 階差分攻撃を高速化できることを報告する。結果としては従来よりも $2^{51.4}$ 倍高速化でき、選択平文数 $2^{98.3}$、平均暗号化計算量 $2^{107.6}$ で攻撃できることを示す。

キーワード：ブロック暗号, CLEFIA, 高階差分, 部分和法, 暗号解読

The improved 96th-order differential attack on reduced 11 rounds of a block cipher CLEFIA

Igarashi Yasutaka[1]　Kaneko Toshinobu[2]　Hashiguchi Yosuke[1]　Eguchi Yutaka[1]
Sueyoshi Ryutaro[1]　Murai Takahiro[1]　Fukushima Seiji[1]　Hachino Tomohiro[1]

Abstract: CLEFIA is a block cipher proposed by Shirai of SONY et al. in 2007. Its block size is 128 bits and its key size is 128, 192, or 256 bits. The number of round is depend on a key size, viz., it is 18, 22, or 26 rounds for 128, 192, or 256 bits of a key size, respectively. Such a characteristic of CLEFIA have been known that the 96th-order differential of 64 bits out of 128 bits of the 8th-round's output is zero. With this characteristic, we reported the 96th-order differential attack on CLEFIA of 11 rounds. This attack requires $2^{98.3}$ of plain text and 2^{159} of computational complexity. In this paper, we reduce this computational complexity by applying a partial sum technique proposed by Ferguson et al.. With the partial sum technique, we sequentially derive frequency distribution tables modulo 2 of intermediate data of cryptanalysis. We also reduce the complexity by introducing a nesting structure of iterative computations. As a result, we reduce the complexity to $2^{107.6}$.

Keywords: block cipher, CLEFIA, higher-order differential, partial sum technique, cryptanalysis

1. はじめに

CLEFIA は 2007 年に SONY の白井らが提案した一般化 Feistel 構造のブロック暗号アルゴリズムである [1]。ブロック長は 128 ビットであり、鍵長は 128、192、256 ビッ

[1] 鹿児島大学
Kagoshima University
[2] 東京理科大学
Tokyo University of Science

トがサポートされている。データ攪拌部の段数は鍵長によって異なり、鍵長が 128、192、256 ビットの場合それぞれ 18 段、22 段、26 段となっている。各段は同一の繰り返し構造であり、2 つの非線形関数が並列に配置された 4 系列一般化 Feistel 構造が採用されている。これまでに、CLEFIA のデータ攪拌部の 6 段目出力 128 ビット中の 24 ビットについては、その同期型 8 階差分がゼロとなる特性が知られている [2], [3]、また 8 段目出力 128 ビット中の 64 ビットについては、その 96 階差分がゼロとなる特性が知られており [4]、我々はこの特性を利用した CLEFIA の 11 段 96 階差分攻撃を報告している [5]。この攻撃には選択平文数 $2^{98.3}$、暗号化計算量 2^{159} を要する。本稿では節 2 で CLEFIA のデータ攪拌部の構造について述べる。節 3 では高階差分に関する知識を整理し、CLEFIA の 96 階差分攻撃の攻撃方程式を示す。節 4 では Ferguson らが提案した部分和法を用いて、解読時の中間データの mod2 頻度分布表を逐次導出することにより文献 [5] の攻撃に要する計算量を削減する。さらに攻撃方程式の計算過程において、繰り返し計算ループの入れ子構造を採用し、入れ子の順序を適切に設定することにより計算量を削減し、11 段 96 階差分攻撃を高速化できることを示す。結果としては従来よりも $2^{51.4}$ 倍高速化でき、選択平文数 $2^{98.3}$、平均暗号化計算量 $2^{107.6}$ で攻撃できることを示す。最後に節 5 でまとめる。尚、CLEFIA の高階差分攻撃に関して現在までに公表されている論文として最も成果を上げているのは、112 階差分を利用した 14 段 CLEFIA の攻撃であり、攻撃に要するデータ量は 2^{113}、計算量は $2^{244.5}$ である [10]。

2. 11 段構成の CLEFIA のデータ攪拌部

本節では本稿の理解に必要な CLEFIA のデータ攪拌部の構造について述べる。本研究には直接関わらない詳細や仕様については文献 [6], [7] を参照されたい。

図 1 に 11 段構成の CLEFIA のデータ攪拌部を示す。入出力はそれぞれ 128 ビットであり、X_i $(i = 0, 1, 2, 3)$ は 32 ビットの入力平文を表し、$C_i^{(11)}$ は 32 ビットの出力暗号文を表す。$C_i^{(j)}$ は j 段目 $(j = 1, 2, 3, \cdots, 11)$ の 32 ビット出力を表す。F_0 と F_1 は 32 ビット入出力の非線形関数を表し、WK_i と RK_ℓ $(\ell = 0, 1, 2, \cdots, 21)$ はそれぞれホワイトニング鍵、段鍵と呼ばれる 32 ビットの鍵を表す。RK_ℓ は F_0 と F_1 の関数内部で使用されている。$\oplus$ は排他的論理和を表す。図に示されるように 4 つの 32 ビットデータが非線形関数により変換、攪拌される構造を 4 系列一般化 Feistel 構造と呼ぶ。

図 2 に F_0 関数と F_1 関数を示す。x_i と y_i はそれぞれ 8 ビットの入力と出力データである。$RK_{\ell,i}$ は 8 ビット鍵であり、次の関係を満たす。$RK_\ell = RK_{\ell,0} \,||\, RK_{\ell,1} \,||\, RK_{\ell,2} \,||\, RK_{\ell,3}$. ここで $||$ はデータの連結を表す。S_0 と S_1 は入出力 8 ビットの非線形で全単射な置換表であり、その出力

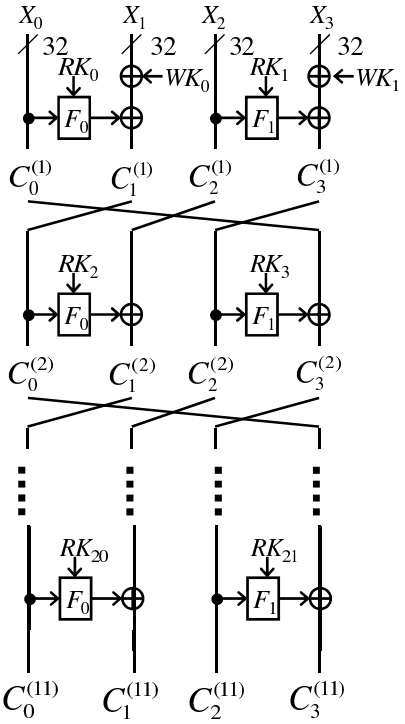

図 1 11 段構成の CLEFIA のデータ攪拌部.

を x_i' で表す。M_0 と M_1 は次式で定義される正則な 4×4 行列である。

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = M_i \begin{pmatrix} x_0' \\ x_1' \\ x_2' \\ x_3' \end{pmatrix}, \quad (i = 0, 1) \tag{1}$$

$$M_0 = \begin{pmatrix} 1 & 2 & 4 & 6 \\ 2 & 1 & 6 & 4 \\ 4 & 6 & 1 & 2 \\ 6 & 4 & 2 & 1 \end{pmatrix}, \; M_1 = \begin{pmatrix} 1 & 8 & 2 & a \\ 8 & 1 & a & 2 \\ 2 & a & 1 & 8 \\ a & 2 & 8 & 1 \end{pmatrix}. \tag{2}$$

ここで a は 16 進数である。式 (1) における行列とベクトルの乗算は特性多項式 $z^8 + z^4 + z^3 + z^2 + 1$ で定義される GF(2^8) 上の演算である。

3. 高階差分

本節では高階差分の定義を示し、高階差分の様々な性質のうち本稿に関係する事項及びその性質を利用した攻撃方程式について一般的に述べる。高階差分の詳細な性質に関しては文献 [8] を参照されたい。次に CLEFIA の 96 階差分特性 [4] とその特性を用いた攻撃方程式を示す [5]。

3.1 定義、性質及び攻撃方程式 [8]

図 3 の暗号回路モデルを例にして高階差分の定義、性質を示し、高階差分を利用した攻撃方程式を示す。

定義

E_1, E_2 はそれぞれ暗号化関数の構成要素を表し、$K_1 \in \mathrm{GF}(2)^{s_1}$, $K_2 \in \mathrm{GF}(2)^{s_2}$ はそれぞれ s_1 bit, s_2 bit の暗

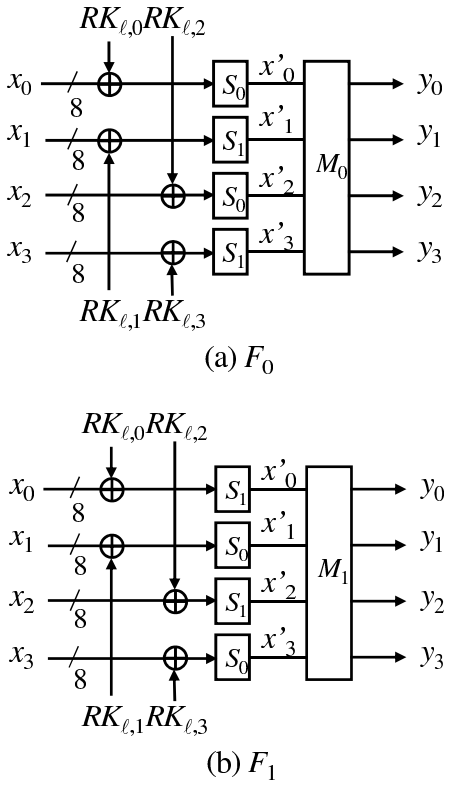

図 2 (a)F_0 関数と (b)F_1 関数.

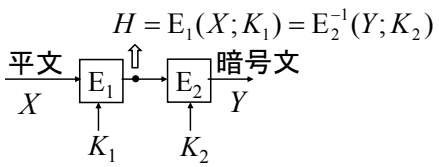

図 3 暗号回路モデル.

号化鍵を表す。$X = (x_1, x_2, \cdots, x_n) \in \mathrm{GF}(2)^n$, $H \in \mathrm{GF}(2)^m$, $Y = (y_1, y_2, \cdots, y_\ell) \in \mathrm{GF}(2)^\ell$ はそれぞれ E_1 への入力 n bit, E_1 の出力 m bit, E_2 の出力 ℓ bit を表し、$H = \mathrm{E}_1(X; K_1)$ とする。ここで $\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_i$ を $\mathrm{GF}(2)^n$ 上の 1 次独立な i 個のベクトルとするとき、これらのベクトルによって張られる $\mathrm{GF}(2)^n$ の i 次元部分空間 $V^{(i)}$ を入力差分という。そして次式で定義される $\Delta^{(i)}\mathrm{E}_1(X; K_1)$ を関数 $\mathrm{E}_1(X; K_1)$ の $V^{(i)}$ に関する i 階差分という。

$$\Delta^{(i)}\mathrm{E}_1(X; K_1) \equiv \sum_{A \in V^{(i)}}^{\oplus} \mathrm{E}_1(X \oplus A; K_1). \tag{3}$$

ここで $\sum^{\oplus}$ は XOR による総和を表す。

性質

今、$\mathrm{E}_1(X; K_1)$ の X に関するブール代数次数が N ならば、次式のように $N+1$ 階差分 $\Delta^{(N+1)}\mathrm{E}_1(X; K_1)$ は X と K_1 に依存せずにゼロになる性質をもつ。

$$\Delta^{(N+1)}\mathrm{E}_1(X; K_1) = 0. \tag{4}$$

さらに $\mathrm{E}_1(X; K_1)$ のブール展開式が x_t $(1 \le t \le n)$ の j 次項 (例えば $x_{t_1} x_{t_2} \cdots x_{t_j}$) を含まなければ、$j$ 個のベクトル $\{\boldsymbol{x}_{t_1}, \boldsymbol{x}_{t_2}, \cdots, \boldsymbol{x}_{t_j}\}$[*1]によって張られる $\mathrm{GF}(2)^n$ の部分空間 $V^{(j)}$ に関する j 階差分 $\Delta^{(j)}\mathrm{E}_1(X; K_1)$ は次式のように X と K_1 に依存せずにゼロになる性質を持つ。

$$\Delta^{(j)}\mathrm{E}_1(X; K_1) = 0. \tag{5}$$

攻撃方程式

今、例として式 (4) が成り立っているとする。この時、関数 E_2 の逆関数を E_2^{-1} とし、K_2 を推定して E_2 の出力 Y から E_1 の出力 H へと遡ることにより次式が成り立つ[*2]。

$$\Delta^{(N+1)}\mathrm{E}_2^{-1}(Y(X); K_2) \equiv \sum_{A \in V^{(N+1)}}^{\oplus} \mathrm{E}_2^{-1}(Y(X \oplus A); K_2) = 0. \tag{6}$$

ここで $Y(X)$ は平文 X に対応する暗号文を表す。式 (6) は暗号化鍵 K_2 の推定が正しい時、常に成立する。一方、推定が誤りの時はランダムに成立すると考えられる。従って推定した K_2 の真偽は式 (6) を用いて検査できる。このように式 (4) のような高階差分の性質を利用した攻撃を高階差分攻撃といい、式 (6) を攻撃方程式という。

3.2 CLEFIA の 96 階差分特性と攻撃方程式

ここでは図 1 を利用して CLEFIA の 96 階差分特性とそれを利用した攻撃方程式を示す。初めに入力平文 128bit の内、X_0 または X_2 を任意の値に固定し、残りの 3 つの X_i に対して 96 階差分を入力する。つまり 3 つの X_i の計 96 ビットに対してオールゼロからオール 1 までの全 2^{96} 通りのデータを入力する。この時、図 1 の $C_0^{(8)}$ 及び $C_2^{(8)}$ の 96 階差分がゼロになるという特性を報告された [4]。さらにこの性質を利用して式 (6) に対応する攻撃方程式を立てると次式となることを報告した [5]。

$$\sum_{A \in V^{(96)}}^{\oplus} \left\{ F_0(C_0^9(X \oplus A); RK_{16}) \oplus C_1^{(9)}(X \oplus A) \right\} = 0, \tag{7}$$

$$C_0^{(9)}(X \oplus A) = F_1(C_2^{(10)}(X \oplus A); RK_{19}) \oplus C_2^{(11)}(X \oplus A), \tag{8}$$

$$C_1^{(9)}(X \oplus A) = F_1(C_2^{(11)}(X \oplus A); RK_{21}) \oplus C_3^{(11)}(X \oplus A), \tag{9}$$

$$C_2^{(10)}(X \oplus A) = F_0(C_0^{(11)}(X \oplus A); RK_{20}) \oplus C_1^{(11)}(X \oplus A). \tag{10}$$

ここで 128 ビット入力平文 $X = X_0 \,||\, X_1 \,||\, X_2 \,||\, X_3$ であり (図 1 参照)、$C_i^{(j)}(X \oplus A)$ は入力平文 $X \oplus A$ に対応する $C_i^{(j)}$ の値を表す。式 (7) は 32 ビットサイズの 1 つの方程式であるが、これを 8 ビットサイズの 4 つの方程式に書き換えるために、図 4 に示すように 8 段目の F_0 関数を 4 並列の S_i 層と M_0 に分解して等価変形する。尚、M_0^{-1} は M_0

[*1] $\boldsymbol{x}_{t_1}$ は n bit のうち t_1 bit 目のみが 1 であり、残りは全てゼロである $\mathrm{GF}(2)^n$ 上のベクトルを表す。
[*2] 式 (5) が成り立っている場合も同様のことが成り立つ。

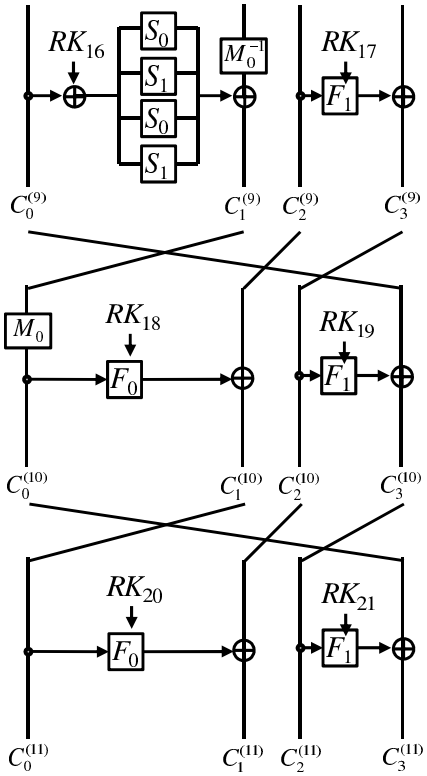

図 4 等価変形したデータ攪拌部.

の逆行列である。4 つの方程式に書き換える理由は攻撃計算量削減のためである。式 (7) を 4 つの方程式に書き換えると次式となる。尚、$C_i^{(j)} = C_{i0}^{(j)} \,||\, C_{i1}^{(j)} \,||\, C_{i2}^{(j)} \,||\, C_{i3}^{(j)}$ とする ($C_{im}^{(j)}$ は 8 ビット ($m = 0, 1, 2, 3$))。また式が見づらくなることを避けるために、$C_i^{(j)}$ や $C_{im}^{(j)}$ の直後にある引数 ($X \oplus A$) はこれ以降全て省略する。

$$\sum_{A \in V^{(96)}}^{\oplus} \left\{ S_0(C_{00}^{(9)}; RK_{16,0}) \oplus C_{10}^{(9)} \right\} = 0, \tag{11}$$

$$\sum_{A \in V^{(96)}}^{\oplus} \left\{ S_1(C_{01}^{(9)}; RK_{16,1}) \oplus C_{11}^{(9)} \right\} = 0, \tag{12}$$

$$\sum_{A \in V^{(96)}}^{\oplus} \left\{ S_0(C_{02}^{(9)}; RK_{16,2}) \oplus C_{12}^{(9)} \right\} = 0, \tag{13}$$

$$\sum_{A \in V^{(96)}}^{\oplus} \left\{ S_1(C_{03}^{(9)}; RK_{16,3}) \oplus C_{13}^{(9)} \right\} = 0. \tag{14}$$

ここで

$$C_1^{(9)} = M_0^{-1}\, C_0^{(10)}, \tag{15}$$

$$C_0^{(10)} = F_1(C_2^{(11)}; RK_{21}) \oplus C_3^{(11)} \tag{16}$$

である。4 つの方程式 (11)–(14) を計算するには $C_0^{(9)}$ と $C_1^{(9)}$ が必要である。$C_0^{(9)}$ と $C_1^{(9)}$ を得るには 4 つの式 (8), (10), (15), (16) を計算する必要がある。文献 [5] ではこれら全ての式 (8), (10), (11)–(16) を計算し、関係する段鍵 RK_{16}, RK_{19}, RK_{20}, RK_{21} の計 128 ビットを特定するために約 2^{159} 回の暗号化計算が必要であると報告している。また相異なる 96 階差分データを 5 組用意すれば計 4×5 組の 8 ビットサイズの攻撃方程式を導くことができる。8 ビットサイズの方程式がランダムに成立する確率は 2^{-8} であり、計 20 組の方程式がランダムに成立する確率は $2^{-8\times 20}$ となる。従って計 128 ビットの鍵候補パターン 2^{128} 通りの中から真の鍵を特定するには、5 組の 96 階差分データを用意すれば十分である。尚、1 組目の 96 階差分データから方程式の検査を実施し 2^{128} 通りの鍵候補パターンをふるいにかけるとパターン数は平均的に $1/2^{32}$ に減少し、2^{96} 通りとなる。2 組目の 96 階差分データからは 2^{96} 通りの鍵候補パターンをふるいにかけることになるので、1 組目のデータに比べて 2 組目のデータに対する計算量は約 $1/2^{32}$ となるので、無視することができる。同じ事が 3 組目以降のデータに対しても当てはまる。

4. mod2 頻度分布表の逐次導出による攻撃計算量の削減

本節では Ferguson らが提案した部分和法 [9] を利用して、式 (8), (10), (11)–(16) に関係する中間データの mod2 頻度分布表 (MFDT) を逐次導出することにより、これらの式の総計算量を削減できることを示す。MFDT とは頻度分布表の頻度を頻度の mod2(つまり 0 または 1) で置き換えた表である。例えばあるデータの頻度が奇数の場合は 1 で置き換え、偶数の場合は 0 で置き換える。MFDT を導出する理由は次の通りである。ある変数 x の偶数個の XOR は必ずゼロであり、奇数個の XOR は必ず x である。従って MFDT を求めておけば、それ以降の演算において変数 x の偶数個の XOR は不要となり、奇数個の XOR 演算結果も x で置き換えることができ、計算量を削減できる可能性があるからである。

次に図 4, 5, 6 を用いて MFDT 導出の手順を示す。図 5 は $C_0^{(j)}$ と $C_1^{(j)}$ を入力とし、$C_2^{(j-1)}$ へと 1 段さかのぼる等価回路であり、図 6 は $C_2^{(j)}$ と $C_3^{(j)}$ を入力とし、$C_0^{(j-1)}$ へと 1 段さかのぼる等価回路である。数字 (n) が四角で囲まれた記号は n 倍算を表す。

初めに式 (11) に着目すると、$C_{00}^{(9)}$ と $C_{10}^{(9)}$ の MFDT が分かれば、$RK_{16,0}$ を仮定して方程式を検査できることが分かる。$C_{00}^{(9)}$ の MFDT は、図 6 において j=10 とすれば、$C_{23}^{(10)}$ と $W_{20}^{(10)}$ の計 16 ビットデータの MFDT が分かれば鍵 $RK_{19,3}$ を仮定することにより導出できる。更に $C_{23}^{(10)}$ と $W_{20}^{(10)}$ の計 16 ビットデータの MFDT は、$C_{22}^{(10)}$, $C_{23}^{(10)}$, $W_{10}^{(10)}$ の計 24 ビットデータの MFDT が分かれば鍵 $RK_{19,2}$ を仮定することにより導出できる。同様にして $C_{22}^{(10)}, C_{23}^{(10)}, W_{10}^{(10)}$ の計 24 ビットデータの MFDT は $C_{21}^{(10)}$, $C_{22}^{(10)}$, $C_{23}^{(10)}$, $W_{00}^{(10)}$ の計 32 ビットデータの MFDT から導出でき、$C_{21}^{(10)}$, $C_{22}^{(10)}$, $C_{23}^{(10)}$, $W_{00}^{(10)}$ の計 32 ビットデータの MFDT は $C_{20}^{(10)}$, $C_{21}^{(10)}$, $C_{22}^{(10)}$, $C_{23}^{(10)}$, $C_{30}^{(10)}$(=$C_{20}^{(11)}$) の計 40 ビットデータの MFDT から導出できる。

$C_{20}^{(10)}$, $C_{21}^{(10)}$, $C_{22}^{(10)}$, $C_{23}^{(10)}$, $C_{30}^{(10)}$(=$C_{20}^{(11)}$) の計 40 ビットデータの MFDT は、図 5 において j=11 とすれば、$C_{03}^{(11)}$,

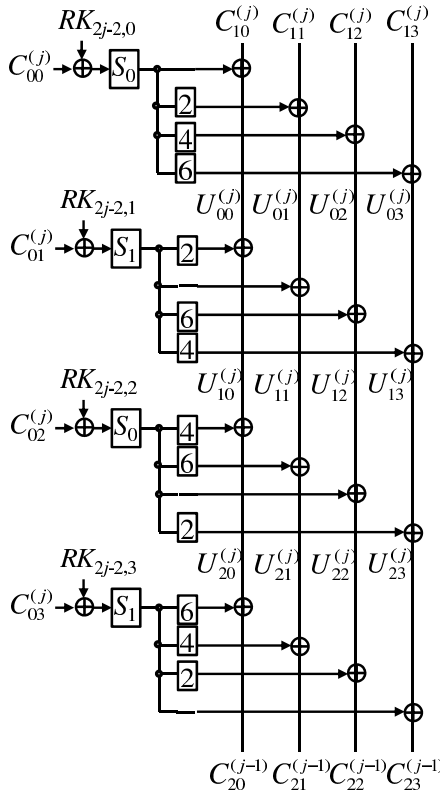

図 5 $C_0^{(j)}$ と $C_1^{(j)}$ を入力とし、$C_2^{(j-1)}$ へさかのぼる等価回路. $U_i^{(j)} = U_{i0}^{(j)} \,||\, U_{i1}^{(j)} \,||\, U_{i2}^{(j)} \,||\, U_{i3}^{(j)}$ とする.

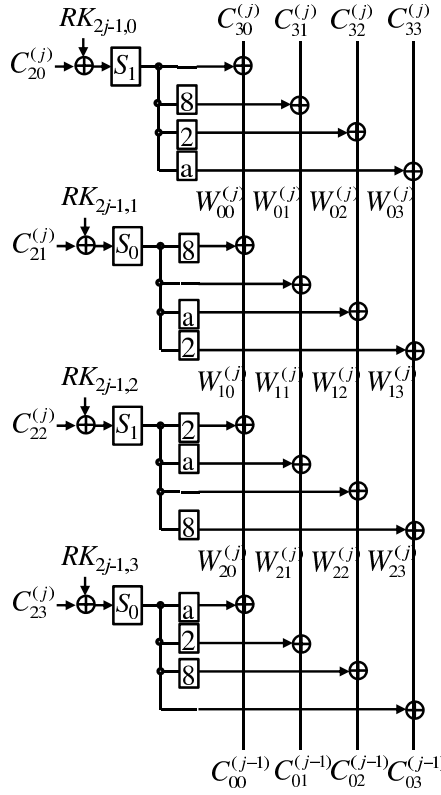

図 6 $C_2^{(j)}$ と $C_3^{(j)}$ を入力とし、$C_0^{(j-1)}$ へさかのぼる等価回路. $W_i^{(j)} = W_{i0}^{(j)} \,||\, W_{i1}^{(j)} \,||\, W_{i2}^{(j)} \,||\, W_{i3}^{(j)}$ とする.

$U_2^{(11)}$, $C_{20}^{(11)}$ の計 48 ビットデータの MFDT から導出できることが分かる。同様にして $C_{03}^{(11)}$, $U_2^{(11)}$, $C_{20}^{(11)}$ の計 48 ビットデータの MFDT は $C_{02}^{(11)}$, $C_{03}^{(11)}$, $U_1^{(11)}$, $C_{20}^{(11)}$ の計 56 ビットデータの MFDT から導出できる。$C_{02}^{(11)}$, $C_{03}^{(11)}$, $U_1^{(11)}$, $C_{20}^{(11)}$ の計 56 ビットデータの MFDT は $C_{01}^{(11)}$, $C_{02}^{(11)}$, $C_{03}^{(11)}$, $U_0^{(11)}$, $C_{20}^{(11)}$ の計 64 ビットデータの MFDT から導出でき、$C_{01}^{(11)}$, $C_{02}^{(11)}$, $C_{03}^{(11)}$, $U_0^{(11)}$, $C_{20}^{(11)}$ の計 64 ビットデータの MFDT は $C_0^{(11)}$, $C_1^{(11)}$, $C_{20}^{(11)}$ の計 72 ビットデータの MFDT から導出できる。

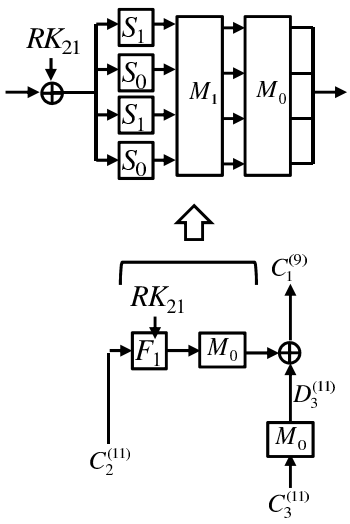

図 7 図 4 において $C_2^{(11)}$ と $C_3^{(11)}$ から $C_1^{(9)}$ へさかのぼる等価回路.

一方、式 (11) の $\sum^{\oplus} C_{10}^{(9)}$ の導出については、図 4 中の $C_2^{(11)}$ と $C_3^{(11)}$ から $C_1^{(9)}$ を逆算する回路を更に等価変形して考える (図 7 参照)。F_1 と M_0 の直列回路は、4 並列 S 層、M_1、M_0 の直列回路となっており、M_1 と M_0 の積は 16 進数表記で次式で与えられる。

$$M_0 M_1 = \begin{pmatrix} 25 & 2e & 22 & 28 \\ 2e & 25 & 28 & 22 \\ 22 & 28 & 25 & 2e \\ 28 & 22 & 2e & 25 \end{pmatrix} \tag{17}$$

従って、図 7 において、$C_2^{(11)}$ と $D_3^{(11)}$ から $C_1^{(9)}$ を計算する回路は図 6 において j=11 と設定し、$C_3^{(j)}$ を $D_3^{(j)}$ と置き換え、$C_0^{(j-1)}$ を $C_1^{(j-2)}$ と置き換え、倍数 1、8、2、a をそれぞれ 25、2e、22、28 と置き換えた回路と等価であることが分かる。

故に式 (11) の $\sum^{\oplus} C_{10}^{(9)}$ は $C_{23}^{(11)}$ と $W_{20}^{(11)}$ の計 16 ビットデータの MFDT から導出できる。$C_{23}^{(11)}$ と $W_{20}^{(11)}$ の計 16 ビットデータの MFDT は $C_{22}^{(11)}$, $C_{23}^{(11)}$, $W_{10}^{(11)}$ の計 24 ビットデータの MFDT から導出でき、$C_{22}^{(11)}$, $C_{23}^{(11)}$, $W_{10}^{(11)}$ の計 24 ビットデータの MFDT は $C_{21}^{(11)}$, $C_{22}^{(11)}$, $C_{23}^{(11)}$, $W_{00}^{(11)}$ の計 32 ビットデータの MFDT から導出でき、$C_{21}^{(11)}$, $C_{22}^{(11)}$, $C_{23}^{(11)}$, $W_{00}^{(11)}$ の計 32 ビットデータの MFDT は $C_{20}^{(11)}$, $C_{21}^{(11)}$, $C_{22}^{(11)}$, $C_{23}^{(11)}$, $D_{30}^{(11)}$ の計 40 ビットデータの MFDT から導出できる。

これまでに説明してきたことと同様の事が式 (12) の $C_{01}^{(9)}$ と $C_{11}^{(9)}$、式 (13) の $C_{02}^{(9)}$ と $C_{12}^{(9)}$、式 (14) の $C_{03}^{(9)}$ と $C_{13}^{(9)}$ に

ついても当てはまる。

以上の事を計 128 ビットの暗号文 $C_0^{(11)}$, $C_1^{(11)}$, $C_2^{(11)}$, $C_3^{(11)}$ から式 (11)–(14) を計算する攻撃アルゴリズムとして整理すると次のように与えられる。

攻撃アルゴリズム

(準備 1)96 階差分入力に対応する 2^{96} 個の暗号文 $C_0^{(11)}$, $C_1^{(11)}$, $C_2^{(11)}$, $C_3^{(11)}$ から次の 8 種類の MFDT を作成する。
- $C_0^{(11)}$, $C_1^{(11)}$, $C_{2m}^{(11)}(= C_{3m}^{(10)})$ の MFDT(計 72bit, 2^{96} 回, $m = 0, 1, 2, 3$)
- $C_2^{(11)}$, $C_3^{(11)}$ の MFDT(計 64bit, 2^{96} 回)

(準備 2)$C_2^{(11)}$, $C_3^{(11)}$ の MFDT と M_0 から $C_2^{(11)}$, $D_{3m}^{(11)}$ の MFDT(計 40bit) を作成 (高々 2^{64} 回, $m = 0, 1, 2, 3$)

(a) $RK_{21,0}$ を 1 つ仮定する (計 2^8 通り)
- $C_2^{(11)}$, $D_{3m}^{(11)}$ の MFDT と S_1(8bit) とそれに続く定数倍の計算から $C_{21}^{(11)}$, $C_{22}^{(11)}$, $C_{23}^{(11)}$, $W_{0m}^{(11)}$ の MFDT(計 32bit) を作成 (高々 2^{40} 回, $m = 0, 1, 2, 3$)

(b) $RK_{21,1}$ を 1 つ仮定する (計 2^8 通り)
- $C_{21}^{(11)}$, $C_{22}^{(11)}$, $C_{23}^{(11)}$, $W_{0m}^{(11)}$ の MFDT と S_0(8bit) とそれに続く定数倍の計算から $C_{22}^{(11)}$, $C_{23}^{(11)}$, $W_{1m}^{(11)}$ の MFDT(計 24bit) を作成 (高々 2^{32} 回, $m = 0, 1, 2, 3$)

(c) $RK_{21,2}$ を 1 つ仮定する (計 2^8 通り)
- $C_{22}^{(11)}$, $C_{23}^{(11)}$, $W_{1m}^{(11)}$ の MFDT と S_1(8bit) とそれに続く定数倍の計算から $C_{23}^{(11)}$, $W_{2m}^{(11)}$ の MFDT(計 16bit) を作成 (高々 2^{24} 回, $m = 0, 1, 2, 3$)

(d) $RK_{21,3}$ を 1 つ仮定する (計 2^8 通り)
- $C_{23}^{(11)}$, $W_{2m}^{(11)}$ の MFDT と S_0(8bit) とそれに続く定数倍の計算から $\sum^{\oplus} C_{1m}^{(9)}$ を導出 (高々 2^{16} 回, $m = 0, 1, 2, 3$)

(1) $RK_{20,0}$ を 1 つ仮定する (計 2^8 通り)
- $C_0^{(11)}$, $C_1^{(11)}$, $C_{3m}^{(10)}$ の MFDT と S_0(8bit) とそれに続く定数倍の計算から $C_{01}^{(11)}$, $C_{02}^{(11)}$, $C_{03}^{(11)}$, $U_0^{(11)}$, $C_{3m}^{(10)}$ の MFDT(計 64bit) を作成 (高々 2^{72} 回, $m = 0, 1, 2, 3$)

(2) $RK_{20,1}$ を 1 つ仮定する (計 2^8 通り)
- $C_{01}^{(11)}$, $C_{02}^{(11)}$, $C_{03}^{(11)}$, $U_0^{(11)}$, $C_{3m}^{(10)}$ の MFDT と S_1(8bit) とそれに続く定数倍の計算から $C_{02}^{(11)}$, $C_{03}^{(11)}$, $U_1^{(11)}$, $C_{3m}^{(10)}$ の MFDT(計 56bit) を作成 (高々 2^{64} 回, $m = 0, 1, 2, 3$)

(3) $RK_{20,2}$ を 1 つ仮定する (計 2^8 通り)
- $C_{02}^{(11)}$, $C_{03}^{(11)}$, $U_1^{(11)}$, $C_{3m}^{(10)}$ の MFDT と S_0(8bit) とそれに続く定数倍の計算から $C_{03}^{(11)}$, $U_2^{(11)}$, $C_{3m}^{(10)}$ の MFDT(計 48bit) を作成 (高々 2^{56} 回, $m = 0, 1, 2, 3$)

(4) $RK_{20,3}$ を 1 つ仮定する (計 2^8 通り)
- $C_{03}^{(11)}$, $U_1^{(11)}$, $C_{3m}^{(10)}$ の MFDT と S_1(8bit) とそれに続く定数倍の計算から $C_2^{(10)}$, $C_{3m}^{(10)}$ の MFDT(計 40bit) を作成 (高々 2^{48} 回, $m = 0, 1, 2, 3$)

(5) $RK_{19,0}$ を 1 つ仮定する (計 2^8 通り)
- $C_2^{(10)}$, $C_{3m}^{(10)}$ の MFDT と S_1(8bit) とそれに続く定数倍の計算から $C_{21}^{(10)}$, $C_{22}^{(10)}$, $C_{23}^{(10)}$, $W_{0m}^{(10)}$ の MFDT(計 32bit) を作成 (高々 2^{40} 回, $m = 0, 1, 2, 3$)

(6) $RK_{19,1}$ を 1 つ仮定する (計 2^8 通り)
- $C_{21}^{(10)}$, $C_{22}^{(10)}$, $C_{23}^{(10)}$, $W_{0m}^{(10)}$ の MFDT と S_0(8bit) とそれに続く定数倍の計算から $C_{22}^{(10)}$, $C_{23}^{(10)}$, $W_{1m}^{(10)}$ の MFDT(計 24bit) を作成 (高々 2^{32} 回, $m = 0, 1, 2, 3$)

(7) $RK_{19,2}$ を 1 つ仮定する (計 2^8 通り)
- $C_{22}^{(10)}$, $C_{23}^{(10)}$, $W_{1m}^{(10)}$ の MFDT と S_1(8bit) とそれに続く定数倍の計算から $C_{23}^{(10)}$, $W_{2m}^{(10)}$ の MFDT(計 16bit) を作成 (高々 2^{24} 回, $m = 0, 1, 2, 3$)

(8) $RK_{19,3}$ を 1 つ仮定する (計 2^8 通り)
- $C_{23}^{(10)}$, $W_{2m}^{(10)}$ の MFDT と S_0(8bit) とそれに続く定数倍の計算から $C_{0m}^{(9)}$ の MFDT(計 8bit) を作成 (高々 2^{16} 回, $m = 0, 1, 2, 3$)

(9) $RK_{16,0}$ を 1 つ仮定する (計 2^8 通り)
- $C_{00}^{(9)}$ の MFDT から $\sum^{\oplus} S_0(C_{00}^{(9)}; RK_{16,0})$ を導出し、式 (11) を検査 (高々 2^8 回)

(10)式 (11) が成立したとき (確率 $p = 1/256$) に限り、$RK_{16,1}$ を 1 つ仮定する (計 2^8 通り)
- $C_{01}^{(9)}$ の MFDT から $\sum^{\oplus} S_1(C_{01}^{(9)}; RK_{16,1})$ を導出し、式 (12) を検査 (高々 2^8 回)

(11)式 (12) が成立したとき (確率 $p = 1/256$) に限り、$RK_{16,2}$ を 1 つ仮定する (計 2^8 通り)
- $C_{02}^{(9)}$ の MFDT から $\sum^{\oplus} S_0(C_{02}^{(9)}; RK_{16,2})$ を導出し、式 (13) を検査 (高々 2^8 回)

(12)式 (13) が成立したとき (確率 $p = 1/256$) に限り、$RK_{16,3}$ を 1 つ仮定する (計 2^8 通り)
- $C_{03}^{(9)}$ の MFDT から $\sum^{\oplus} S_1(C_{03}^{(9)}; RK_{16,3})$ を導出し、式 (14) を検査 (高々 2^8 回)

手順の (a)–(d) は、式 (11)–(14) における $\sum^{\oplus} C_{1m}^{(9)}$ ($m = 0, 1, 2, 3$) を導出する過程であり、手順の (1)–(12) は $\sum^{\oplus} S_i(C_{0m}^{(9)}; RK_{16,m})$ ($i = 0$ or 1) を導出する過程である。手順の (準備 1) と (準備 2) は 1 回実施し、手順 (a)–(d) 及び (1)–(12) は繰り返し (for) ループの入れ子構造を利用して鍵候補パターン全通り 2^{128} の検査を実施する。

次に MFDT の作成コストや M_0 の計算コスト、S_i とそれに続く定数倍の計算コストを考える。初めに例としてビットサイズ 16(要素数 2^{16}) の MFDT を取り上げる。この MFDT に 1 回アクセスすることは互いに独立な 2 つの S_i の値を 1 回で (同時に) 計算していることと等価なので、S_i の 2 回分の計算量と見積もる。同様にして、ビットサイズ $8n$ の MFDT へのアクセスは S_i の n 回分の計算量と見積もる。またビットサイズ $8n$ の MFDT における 1 の要素数は高々 2^{8n} であり、データの出現頻度が一様ランダムであれば 1 の要素数は平均 2^{8n-1} である。

行列 M_0 については、これをテーブル化すると、その要

素数はビットサイズ 32 の MFDT の要素数と等しくなる。故に M_0 の 1 回の計算は S_i の 4 回分の計算量と等しいと見積もる。同様にして、S_i とそれに続く定数倍の 1 回の計算は S_i の 1 回分の計算量と等しいと見積もる。

ところで、攻撃アルゴリズム全体の計算量を削減するためには、計算量の多い手順は繰り返しループの入れ子構造の外側で計算し、計算量の少ない手順は入れ子構造の内側で計算する必要がある。攻撃アルゴリズムの各手順における計算量を考えると、外側から次の順序で入れ子構造を構成すると計算量が少なくなる。(1) → (2) → (3) → (4) → (5) → (a) → (6) → (b) → (7) → (c) → (8) → (d) → (9) → (10) → (11) → (12). この計算順序に従った攻撃アルゴリズム全体における S_i 関数の計算回数の最大値 (T_{max}) と平均値 ($\bar{T}$) は次式で与えられる。

$$T_{max} = T_0 + T_1 + T_2 \approx T_2 \approx 2^{136.0}, \tag{18}$$

$$T_0 = 9 \cdot 2^{96} \cdot 4 + 8 \cdot 2^{96}, \tag{19}$$

$$T_1 = 9 \cdot 2^{64} \cdot 4, \tag{20}$$

$$T_2 = 2^8(9 \cdot 2^{74} + 2^8(8 \cdot 2^{66} + 2^8(7 \cdot 2^{58} + 2^8(6 \cdot 2^{50} + T_3)))), \tag{21}$$

$$T_3 = 2^8(5 \cdot 2^{42} + 2^8(5 \cdot 2^{42} + 2^8(4 \cdot 2^{34} + 2^8(4 \cdot 2^{34} + T_4)))), \tag{22}$$

$$T_4 = 2^8(3 \cdot 2^{26} + 2^8(3 \cdot 2^{26} + 2^8(2 \cdot 2^{18} + 2^8(2^{18} + T_5)))), \tag{23}$$

$$T_5 = 2^8(2^8 + 2^8(2^8 + 2^8(2^8 + 2^8(2^8)))). \tag{24}$$

ここで T_0 は (準備 1) の計算量であり、T_1 は (準備 2) の計算量である。T_5 は手順 (9) → (10) → (11) → (12) の入れ子構造の計算量であり、T_4 は手順 (7) → (c) → (8) → ⋯ → (11) → (12) の入れ子構造の計算量である。同様に T_3 は手順 (5) → (a) → (6) → ⋯ → (11) → (12) の入れ子構造の計算量であり、T_2 は手順 (1) → (2) → (3) → ⋯ → (11) → (12) の入れ子構造の計算量である。

$$\bar{T} = T_0 + \bar{T_1} + \bar{T_2} \approx \bar{T_2} \approx 2^{114.0}, \tag{25}$$

$$\bar{T_1} = 9 \cdot 2^{63} \cdot 4, \tag{26}$$

$$\bar{T_2} = 2^8(9 \cdot 2^{73} + 2^8(8 \cdot 2^{65} + 2^8(7 \cdot 2^{57} + 2^8(6 \cdot 2^{49} + \bar{T_3})))), \tag{27}$$

$$\bar{T_3} = 2^8(5 \cdot 2^{41} + 2^8(5 \cdot 2^{41} + 2^8(4 \cdot 2^{33} + 2^8(4 \cdot 2^{33} + \bar{T_4})))), \tag{28}$$

$$\bar{T_4} = 2^8(3 \cdot 2^{25} + 2^8(3 \cdot 2^{25} + 2^8(2 \cdot 2^{17} + 2^8(2^{17} + \bar{T_5})))), \tag{29}$$

$$\bar{T_5} = 2^8(2^7 + p \cdot 2^8(2^7 + p \cdot 2^8(2^7 + p \cdot 2^8(2^7)))). \tag{30}$$

$\bar{T_i}$ ($i = 1, 2, 3, 4, 5$) は T_i の平均計算量である。11 段構成の CLEFIA には S_i 関数が 88 個含まれているので、T_{max} と $\bar{T}$ を暗号化関数の計算回数に変換するには、それぞれに 1/88 を乗算すればよい。従って、攻撃アルゴリズム全体における暗号化関数の計算回数の最大値 (T'_{max}) と平均値 ($\bar{T}'$) は次式で与えられる。

$$T'_{max} = T_{max}/88 \approx 2^{129.5}, \tag{31}$$

$$\bar{T}' = \bar{T}/88 \approx 2^{107.6}. \tag{32}$$

尚、攻撃に必要な選択平文数は従来 [5] と変わらず $2^{98.3}$ である。

5. おわりに

本稿ではブロック暗号アルゴリズム CLEFIA の 11 段 96 階差分攻撃の高速化について報告した。Ferguson らが提案した部分和法を用いて、解読時の中間データの mod2 頻度分布表を逐次導出することにより攻撃に要する計算量を削減した。さらに攻撃方程式の計算過程において、繰り返しループの入れ子構造を採用し、入れ子の順序を適切に設定することにより計算量を削減した。結果としては従来の攻撃では選択平文数 $2^{98.3}$、暗号化関数計算回数 2^{159} を要していたが、本攻撃では選択平文数 $2^{98.3}$、最大暗号化関数計算回数 $2^{129.5}$、平均暗号化関数計算回数 $2^{107.6}$ で済み、計算回数が $1/2^{51.4}$ に減少した。今回報告した計算量削減手法を文献 [2], [3] に適用することが今後の検討課題である。

参考文献

[1] T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, “The 128-Bit Blockcipher CLEFIA (Extended Abstract),” LNCS, Vol.4593, pp.181-195, Springer-Verlag, 2007.

[2] 芝山直喜, 金子敏信, 半谷精一郎, “CLEFIA の特殊な高階差分特性,” 暗号と情報セキュリティシンポジウム, No. 1C3-2, pp.1–6, 2012.

[3] Naoki Shibayama and Toshinobu Kaneko, “A Peculiar Higher Order Differential of CLEFIA,” International Symposium on Information Theory and its Applications, pp. 526-530, Oct. 2012.

[4] 角尾幸保, 辻原悦子, 久保博靖, 茂真紀, 川幡剛嗣, “一般化Ｆｅｉｓｔｅｌ構造の飽和特性,” 電子情報通信学会論文誌 A, Vol. J93-A(4), pp. 269–276, 2010.

[5] 芝山直喜, 五十嵐保隆, 金子敏信, 半谷精一郎, “共通鍵ブロック暗号 CLEFIA の飽和攻撃耐性評価,” 暗号と情報セキュリティシンポジウム, No. 2B1-4, pp.1–7, 2011.

[6] Sony Japan | CLEFIA, http://www.sony.co.jp/Products/cryptography/clefia/index.html

[7] Sony Corporation, “The 128-bit Blockcipher CLEFIA Algorithm Specification Revision 1.0,” June 1, 2007, http://www.sony.co.jp/Products/cryptography/clefia/download/data/clefia-spec-1.0.pdf

[8] CRYPTREC 応募暗号の高階差分及び補間攻撃耐性について (Ver.1), pp.3–7, 平成 13 年 1 月,http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/113_report.pdf

[9] N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting,“Improved Cryptanalysis of Rijndael,” LNCS, Vol. 1978, pp. 136-141, Springer, 2001.

[10] Yanjun Li, Wenling Wu, Lei Zhang, “Improved Integral Attacks on Reduced-Round CLEFIA Block Cipher,” LNCS, Vol. 7115, pp 28-39, Springer, 2012.