コンピュータセキュリティ 11－7
（2000．9．8）

(入れ子型)SPN暗号の改良

大熊 建司† 村谷 博文† 佐野 文彦‡ 本山 雅彦† 川村 信一†
†(株) 東芝 研究開発センター 〒212-8582 川崎市幸区小向東芝町 1
{kenji.ohkuma,hirofumi.muratani,masahiko.motoyama,shinichi2.kawamura}@toshiba.co.jp
‡(株) 東芝 SI 技術開発センター 〒183–8512 東京都府中市片町 3-22
fumihiko.sano@toshiba.co.jp

あらまし 著者らは階層的に差分／線形解読法に対する安全性を保証する、2 階層入れ子型 (再帰的)SPN 構造を持った 128 ビット・ブロック暗号 "Hierocrypt" を提案した。その提案では、上位拡散層の設計に利用するガロア体の違いによって、Type-I (GF(2^{32})) と Type-II (GF(2^4)) の 2 種類があった。また、鍵スケジュールには、256 ビット幅の変形 Feistel 型構造を採用した。本報告では、両方式の実装検討結果に基づき、ソフト実装で有利であった Type-II 方式をベースに改良した改良版 Hierocrypt-3 を提案する。

A Revised Nested SPN Cipher

Kenji Ohkuma† Hirofumi Muratani† Fumihiko Sano‡ Masahiko Motoyama† Shinichi Kawamura†
† Computer & Network Systems Laboratory, Corporate Research & Development Center
1, Komukai Toshiba-cho, Saiwai-ku, Kawasaki, 212-8582, Japan
‡ System Integration Technology Center
3–22, Katamachi, Fuchu-shi, Tokyo 183–8512, Japan

Abstract The authors proposed two 128-bit block cipher algorithms Hierocrypt Type-I and Type-II based on a nested SPN strucuture where the upper-level S-box consists of the lower-level SP network hierarchically. The key scheduling parts are designed on a 256-bit modified Feistel structure. This paper proposes an improved version "Hierocrypt-3," based on the Type-II algorithm.

1 はじめに

筆者らは、2 階層入れ子型 SPN 構造を用いた 128 ビット・ブロック暗号 Hierocrypt(Type-I,-II) を提案した [7, 8]。2 階層入れ子型 SPN 構造とは、上位構造と下位構造がともに SPN 構造であり、上位構造の S-box が下位 SPN 構造で構成される再帰的階層構造である [7, 8]。

Type-I および II の実装を具体的に検討したところ、ローエンド環境での実装に問題があることが判明した。そこで、2 種類のうちソフト実装で有利だった Type-II をベースに改良を行ない、改良版 Hierocrypt-3 を設計した。(a)SQUARE 攻撃により強い上位拡散層への置き換え。(b) 実装を小さくするため下位拡散層を巡回型にした。(c)S-box を最大差分／線形確率が 8 ビット幅での最良の 2^{-6} にした。(d) 多項式表現の項数を基準に S-box と下位拡散層の適切な組合せを選択。(e)on-the-fly 実装の復号でも初期遅延が小さくなるよう、中間鍵スケジュールを折り返し型にした。

以下では、設計方針と安全性評価に述べ、アルゴリズムの仕様を付録とした。また、参考文献と Hierocrypt-3 の基本関数の仕様については、本報告 の次の報告「64bit 版 Hierocrypt の提案」を参照されたい [5] 。

2 設計方針

Hierocrypt-3 の設計に当たり、我々は次の点を重視した。

- 主要な攻撃法に対する十分な安全性
- スマートカードやミドルウェアでの高速性
- 実装の効率性
- 設計の透明性

以上の条件から、データ撹拌部には入れ子型 SPN 構造を [7, 8]、鍵スケジュール部には Feistel 構造を採用した。

2.1 データ撹拌部

データ撹拌部の設計では、Feistel 構造と並んで有力なブロック暗号の代表的な基本構造である SPN 構造を採用した。

SPN 構造の長所には以下のものがある。

- 符号理論に基づく設計手法が確立している
- Feistel 型と異なり、素通しのデータ経路が無い
- Feistel 型と比較して、自明な弱鍵が存在しにくい
- ハード実装で高速

一方、短所としては以下の通りである。

- 暗号化と復号が異なり、実装サイズが大きくなる
- 拡散層幅が Feistel 型の 2 倍で、計算コストが大きい

2.1.1 入れ子型 SPN 構造

入れ子型 SPN 構造とは、上位の SPN 構造の S-box の位置に下位の SPN 構造を埋め込んだ、再帰的階層構造である。SPN 構造の持つブランチ数などの安全性が、階層的に保証される。計算コストに関しては、下位の拡散層では拡散範囲が局所化され、上位拡散層では MDS 符号での語長が長く・語数が減るので、オリジナルの SPN 構造より有利になっている。我々は、一般の入れ子型 SPN 構造暗号設計に当たり、満たすべき条件を文献 [7, 8] で提案した。

(a) (下位)S-box のサイズは 8 ビット
(b) 2 階層の入れ子構造 (上位レベルと下位レベル)
(c) 下位レベルは 2 段 SPN 構造
(d) 上位／下位レベルとも、拡散層幅は S-box 4 個分

これらの条件の理由は次の通り。(a)S-box サイズの 8 ビットは、表引きでの実装が現実的な上限である。(b)3 階層以上では計算コストの面で不利なので、2 階層とした。(c) 活性 S-box 数が効率的に保証できる最小の段数は、2 段 SPN 構造 (SPS) である。(d)S-box の並列度は低い方が計算コストの面で望ましいので、上位／下位レベルとも S-box 4 個分とした。

入れ子型 SPN 構造は簡潔で、透明性が高い。また、構成要素をある程度独立に設計でき、暗号のブロック・サイズの変更にも柔軟に対応できる。

2.2 鍵スケジュール部

鍵スケジュール部の設計において、安全面では、拡大鍵間の単純な依存関係を避けて実質鍵長の低下を防ぐこと。実装面では、1段当たりの計算時間がデータ攪拌部より短く、on-the-fly の復号時の初期遅延を小さくするよう留意した。

Hierocrypt-3 では 256 ビット鍵まで利用可能なので、中間鍵は 256 ビットで段関数は全単射とした。

256 ビットを 128 ビット 2 組に分け、一方を繰り返し (全単射) 線形変換型、他方を Feistel 型とし、前者が後者の鍵に相当するデータを供給する。Feistel 構造の F 関数の拡散層は 64 ビット幅で、十分高速な動作が実現できる。

Hierocrypt-3 では中心付近の段で中間鍵生成の変換を逆転し、中間状態値が元に戻っていく折り返し型とした。この構造だと、on-the-fly の復号時でも、暗号化鍵から 1,2 回の変換で初段の中間鍵が生成でき、初期遅延は小さい。

拡大鍵は、中間鍵と生成途中の中間状態のデータを 64 ビット単位で排他的論理和することで生成する。この際、平文側と暗号文側で同じ拡大鍵が生成されず、単純な依存関係による弱鍵の出現しないよう注意した。

中間鍵に周期パターンが生じないよう、線形変換部で段数依存定数を加算した。設計の透明性のため、加算定数は、小さな整数の平方根を用いて作成した。段依存定数は、関連鍵攻撃の適用を困難にするため、鍵長依存とした。

3 設計基準

3.1 設計基準の細目

安全性について 最も基本的な安全性は暗号化鍵に対する全数探索であるが、これは暗号化鍵のビット長だけで決まる。暗号化鍵ビット長以外の主要な安全性の評価基準では、以下の攻撃に対する強度が重要である。(1a) 差分解読法; (1b) 線形解読法; (1c) 高階差分解読法; (1d) 補間攻撃法; (1e) SQUARE 攻撃; (1f) truncated 差分解読法; (1g) impossible 差分攻撃.

高速性について 暗号化および復号における高速性に関しては、以下の要素が重要である。(2a) データ攪拌部の速度; (2b) 鍵設定時間; (2c) on-the-fly 鍵スケジュールの速度.

実装効率 実装効率に関し、以下が重要である。(3a) オブジェクトコードが短い; (3b) RAM が小さい; (3c) ROM が小さい.

3.2 構成要素の設計

3.2.1 S-box

S-box の設計で、我々は以下の点を最も重要視した。(i) 最大差分確率と最大線形確率; (ii) 代数次数; (iii) 多項式表現での項数; (iv) 単純な代数構造の非存在.

最大差分確率 dp^s および最大線形確率 lp^s は 8 ビット入出力の最小値、$dp^s = lp^s = 2^{-6}$ となるものにした。$\mathrm{GF}(2^8)$ 上のべき乗関数でこの条件を満たすものがあるが、代数攻撃に対する耐性が低いので、入力側でのビット置換を挿入した。また、S-box の不動点の存在と統計的偏りを避けるため、出力側での定数和 (排他的論理和) を導入した。

$$s(x_{(8)}) = Add(Power(Perm(x_{(8)}))) ,$$
$$y_{(8)} = Perm(x_{(8)}) , \quad y_{i(1)} = x_{\pi[i](1)} ,$$

i	1	2	3	4	5	6	7	8
$\pi[i]$	3	7	5	8	6	2	4	1

$$Power : \mathrm{GF}(2^8) \rightarrow \mathrm{GF}(2^8) ,$$
$$Power\left(x_{(8)}\right) = {x_{(8)}}^{247} ,$$
$$Add\left(x_{(8)}\right) = x_{(8)} \oplus \mathrm{0x7} .$$

べき乗の指数 247 は、最大差分／線形確率が 2^{-6} になるもので、位数が最大なので選んだ。なお、原始多項式には、$z^8 + z^6 + z^5 + z + 1$ を用いた。定数加算値 0x7 は、入力と出力のハミング重みの組合せの分布が、ランダム関数の分布に近くなるように選んだ (相関計数は 0.093750)。$Perm$ は、x^{247} 自身とビット置換を挿入した関数を多項式表現で表わし、異なる項の個数が最多になるものを選んだ。

3.2.2 mds_{L}

以下を mds_{L} の設計基準とした。(i) 最大距離分離 (MDS) 行列; (ii) 巡回型; (iii) S-box と組み合わせたときの、多項式表現 (ビット単位) での項数が大きい.

(i) は xs が活性の場合、8 個の S-box のうち 5 個以上が活性に成ることを保証するための条件で、$\mathrm{GF}(2^8)$ 上の四則演算を利用して構成する。(ii) は実装の小型化に有効。

(iii) の条件は、補間攻撃や高階差分攻撃に対する耐性を高めるものである。S-box1 個とそれに接続する mds_{L} を一体とする入力 8 ビット・出力 32 ビットの変換に注目し、(i),(ii) の条件を満たし、出力ビットの重み付き項数和が最大のものを選んだ。重み付き項数和とは、次数 n の項が存在するとき、それを n 個として数える方式のことである。

条件 i)~iii) を満たす mds_{L} の設計手順を示す。

1. 条件 (iii) に関する $\mathrm{GF}(2^8)$ の元の上位候補で巡回型の行列を作る
2. MDS であれば次項へ。否なら前項に戻る。
3. 定数倍ごとの重み付き項数和の総和を MDS 行列とその逆行列に対して計算し和を取る。
4. 先頭に戻る
5. 項数和の上位候補から計算コストの低いものを選ぶ

この結果、最終的に次式の mds_{L} が得られた。

$$\begin{pmatrix} \mathrm{C4} & 65 & \mathrm{C8} & \mathrm{8B} \\ \mathrm{8B} & \mathrm{C4} & 65 & \mathrm{C8} \\ \mathrm{C8} & \mathrm{8B} & \mathrm{C4} & 65 \\ 65 & \mathrm{C8} & \mathrm{8B} & \mathrm{C4} \end{pmatrix}$$

ここで、行列要素は $\mathrm{GF}(2^8)$ の元を 16 進表現した。

3.2.3 MDS_{H}

MDS_{H} の設計基準として以下を重視した。(i) MDS 行列である, (ii) 巡回型である, (iii) バイト単位で見た複数経路性, (iv) バイト間結合数が出来るだけ少ない.

条件 (i) は、4 個の 32 ビット語を入出力とする MDS であるということである。ここで、$MDS(m,n)$ を m 個の n ビット語を入出力とする MDS 写像とする。文献 [7, 8] の Proposition 2 が示すように、$MDS(32,4)$ は $MDS(4,4)$ 8 個の組み合わせで構成できる。また、$MDS(4,4)$ が全て同じとき、MDS_{H} はバイト間の排他的論理和の組合せであり、16 行 16 列の行列で表わせる。

SPN 構造の代表的なものに、SQUARE と Rijndael があり、SQUARE 攻撃という専用攻撃法が 6 段まで有効である。我々は、ある段の S-box と結線で繋がっている 2 段前の S-box が、各 4 バイト・セットの中に 2 個以上ある性質を複数経路性と呼び、この性質を満たせば SQUARE 攻撃に強いことが示せる (後述)。条件 (iii) は、MDS_{H} が順逆両方向に対して複数経路性を満足することを要求する。複数経路性を満足するための必要十分条件は、$MDS(4,4)$ の全要素が $\{3,5,6,7,A,B,C,E\}$ に含まれることである。

具体的には、行列は以下の手順で選択する。

1. $\mathrm{GF}(2^4)$ の元の定数倍の候補で巡回型の行列を作る
2. MDS であれば次項へ。否なら前項に戻る

3. 逆行列を求め、行列要素が全部 GF(2^4) の候補に含まれているなら次項へ。否なら先頭に戻る。
4. 最終的に残った巡回行列を候補とする

上記の手順により、行内要素の回転と反転の自由度を除き、4種類の候補が求まった。その中の2個ずつは、互いに逆行列を行内で回転したものと一致する。ハード実装を考慮し、結線数が最も少ない次の行列を選んだ。

$$\begin{pmatrix} 5 & 5 & A & E \\ E & 5 & 5 & A \\ A & E & 5 & 5 \\ 5 & A & E & 5 \end{pmatrix}$$

ここで、行列要素は GF(2^4) の元を 16 進表現したものであり、原始多項式は z^4+z+1 である。

3.2.4 $P^{(n)}$

$P^{(n)}$ は鍵スケジュール部の拡散層として利用される。$P^{(n)}$ に対し、以下の設計基準を設けた。

(i) 1段あたりの計算時間がデータ攪拌部より短かい。(ii) 変換するデータのビット長に対するスケーラビリティが有る (ブロック・サイズの変更に柔軟に対応するため)。(iii) 拡散性が高い (拡大鍵から暗号化鍵の推定を困難にするため)。(iv) 全単射である (拡大鍵の持つ自由度の縮退を防ぐ)。(v) 最大距離分離 (MDS) 性は要求しない (高速性と安全性に対するバランスを考慮)。

上記の条件により、($4n$) ビット・データに対する次式の線形変換を選択した。

$$Y_{(4n)} = P^{(n)}\left(X_{(4n)}\right) .$$

$$\begin{pmatrix} y_{1(n)} \\ y_{2(n)} \\ y_{3(n)} \\ y_{4(n)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_{1(n)} \\ x_{2(n)} \\ x_{3(n)} \\ x_{4(n)} \end{pmatrix} .$$

この線形変換は、次式に示すように2種類の involution 型線形変換の合成関数になっている。

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} .$$

4 安全性評価

4.1 差分解読法および線形解読法

関数 f の最大差分確率を dp^f、最大線形確率を lp^f とする。

4.1.1 S-box の特性

Hierocrypt-3 の S-box 変換は、8-bit 入力に対し次の3段階の変換を順に行なうのと同値である。

(a) ビット置換
(b) GF(2^8) 上でのべき乗演算 x^{247}
(c) GF(2^8) 上での排他的論理和 $x \oplus 0x7$

変換 (a),(c) は差分／線形確率の分布を変えないので、S-box の最大差分／線形確率は (b) と一致し、ともに 2^{-6}。

$$dp^S = lp^S = 2^{-6} .$$

4.1.2 活性 S-box 数に基づく特性確率評価

mds_{L} と MDS_{H} が MDS 行列なので、連続する2段が含む活性 S-box の下限は、文献 [8, 7] の Proposition. 2 から $5 \times 5 = 25$ となる。よって、連続する2段の最大差分 (線形) 特性確率 $DP^{2\mathrm{R}}$($LP^{2\mathrm{R}}$) は次式の不等式を満足する。

$$DP^{2\mathrm{R}} \le (dp^S)^{25} = (2^{-6})^{25} = 2^{-150} .$$

$$LP^{2\mathrm{R}} \le (dp^S)^{25} = (2^{-6})^{25} = 2^{-150} .$$

これらの値は 2^{-128} より小さく、有効な特性確率の経路は存在しない。2段攻撃を仮定すれば、鍵長 128 ビットに対しては、$2+2=4$ で、4段有れば十分である。ただし、鍵長が 192／256 ビットの場合は探索鍵ビットも増やせるので段数を増やす必要がある。上位 S-box XS 1個の鍵の全数探索 (64 ビット) で解読可能段数を1段伸ばせると仮定すると、鍵長 128 ビットを基準段数としたとき、鍵長 192 ビットでは1段増やし、鍵長 256 ビットでは2段増やすのが、適当である。この結果を Table 1 に示す。

Table 1: 活性 S-box 数に基づく最小段数の評価

鍵長 (bit)	最小段数 R	$R-2$	特性確率
128	4 段	2 段	2^{-150}
192	5 段	3 段	2^{-180}
256	6 段	4 段	2^{-300}

4.1.3 証明可能安全性に基づく評価

FSE 2000 において、高麗大学の Hong らは、n 個の並列 S-box 2層を MDS 拡散層 (P) で直列に結合した SPS 構造の最大差分 (線形) 確率が、S-box に対する確率の n 乗を超えないことを証明した [2]。

この定理から、上位 S-box XS の最大差分 (線形) 確率 dp^{XS}(lp^{XS}) の上限値が評価できる。

$$dp^{XS} \le (dp^f)^4 = (2^{-6})^4 = 2^{-24} ,$$

$$lp^{XS} \le (lp^f)^4 = (2^{-6})^4 = 2^{-24} .$$

同様に、連続する2段に対し、次の不等式が成立する。

$$dp^{2\mathrm{R}} \le (dp^{XS})^4 = (2^{-24})^4 = 2^{-96} ,$$

$$lp^{2\mathrm{R}} \le (lp^{XS})^4 = (2^{-24})^4 = 2^{-96} .$$

2段での最大確率 2^{-96} は、全数探索より効率の良い解読の存在を否定しない。そこで、2段および XS に対する最大確率の上界値を用いて、3段以上の近似確率評価を行なう。2段攻撃を仮定し、偶数段に対しては2段の確率の積で、奇数段に対しては2段の確率の積と残り1段分の XS の確率の積で評価した結果を Table 2 に示す。鍵長の増加については、Table 1 のときと同じ仮定を用いた。

Table 2: 証明可能安全性に基づく最小段数の評価

鍵長 (bit)	最小段数 R	$R-2$	特性確率
–	4 段	2 段	2^{-96}
–	5 段	3 段	2^{-120}
128 ビット	6 段	4 段	2^{-192}
192 ビット	7 段	5 段	2^{-216}
256 ビット	8 段	6 段	2^{-288}

ここでの対差分／線形解読強度評価の近似精度は通常の最大特性確率よりも高い。

4.2 SQUARE 攻撃

SQUARE 攻撃は、基本攻撃と鍵推定による適用段数の拡張が提案されている [1]。Hierocrypt-3 と SUQARE 暗号の比較を行なう際の混乱を避けるため、以下では段数に代えて、S-box 層の数で定義する総数を用いる。

SQUARE 攻撃は、Hierocrypt-3 に対し、128 ビット鍵では4層まで、192 ビット鍵と 256 ビット鍵では5層までしか有効ではないことを確認した。

Hierocrypt-3 は、鍵長 128 ビット、192 ビット、256 ビットに対して、それぞれ、12層、14層、16層となるよう設計されているので、Hierocrypt-3 のいずれも SQUARE 攻撃に対して十分な安全性を有する。

4.2.1 Hierocrypt-3 への SQUARE 攻撃

基本攻撃 Hierocrypt-3 は入れ子型 SPN 構造を持つため、SQUARE 暗号の 2 層分が 1 段となっている。そこで、拡張 S-box の第 1 層目の鍵加算層から暗号化関数が始まるとした場合と第 2 層目の鍵加算層から暗号化関数が始まるとした場合に対応して、2 通りの基本攻撃を定義できる。それぞれ、基本攻撃 1、基本攻撃 2 と呼ぶ。

Hierocrypt-3 では、初層における Λ 集合は、S-box 層の数で数えて第 3 層目の S-box 層の入力においてすでに Λ 集合ではなく、第 3 層目の S-box 層の通過後の状態バイトは Λ 集合上でバランスしない。

従って、基本攻撃 1 と基本攻撃 2 が適用可能なのは、いずれも 3 層までであり、必要な既知平文数は 2^9 である。これは、必要な既知平文を n 個の Λ 集合とすると、推定された 1 バイトの鍵をユニークに特定するには、$(\frac{1}{256})^n \times 2^8 < 1$ を満足するしなければならないことによる。

タイプ 1 の拡張 基本攻撃の終層を 1 層拡張して鍵推定を行うことで、層数を増やして基本攻撃を適用できるようにする拡張をタイプ 1 の拡張と呼ぶ。

タイプ 2 の拡張 基本攻撃の初層に 1 層拡張して、最初の鍵加算の鍵推定を行うことで、層数を増やして基本攻撃を適用できるようにする拡張をタイプ 2 の拡張と呼ぶ。

各種拡張に対する同様の考察結果を表 3 に示す。Hierocrypt-3 に対する SQUARE 攻撃が有効な層数は、128 ビット鍵では 4 層まで、192 ビット鍵と 256 ビット鍵では 5 層までである。

Table 3: Hierocrypt-3 への SQUARE 攻撃

攻撃のタイプ	層数	平文数	演算数	メモリ量
基本 1	3 層	2^9	2^9	small
基本 1+ タイプ 1	4 層	2^{11}	2^{40}	small
基本 1+ タイプ 1×2	5 層	2^{13}	2^{168}	2^{13}
基本 1+ タイプ 2	4 層	2^{128}	2^{136}	2^{128}
基本 1+ タイプ 1 + タイプ 2	5 層	2^{128}	2^{168}	2^{128}
基本 2	3 層	2^9	2^9	small
基本 2+ タイプ 1	4 層	$\geq 2^{12}$	$\geq 2^{72}$	$\geq 2^{12}$
基本 2+ タイプ 1×2	5 層	$\geq 2^{13}$	$\geq 2^{208}$	$\geq 2^{13}$
基本 2+ タイプ 2	4 層	2^{32}	2^{40}	2^{32}
基本 2+ タイプ 2×2	5 層	2^{128}	2^{168}	2^{128}
基本 2+ タイプ 1 + タイプ 2	5 層	2^{32}	2^{168}	2^{32}

4.3 truncated 差分攻撃

バイト (8 ビット) 単位の truncated 差分攻撃では、差分をバイト単位での零／非零で分類する。バイト単位の演算の組み合わせで構成される Hierocrypt-3 の場合、truncated 差分攻撃による強度評価は不可欠である。

4.3.1 準備

Hierocrypt-3 では全ての演算がバイト単位の処理で書けるので、バイト単位の truncated 差分を定義する。$8m$ ビット・データの差分 $\Delta X_{(8m)}$ に対する truncated 差分 $\chi(\Delta X_{(8m)})$ を次式で定義する。

$$\chi(\Delta X_{(8m)}) = \delta(\Delta x_{1(8)}) \| \delta(\Delta x_{2(8)}) \| \cdots \| \delta(\Delta x_{m(8)}) ,$$

$$\delta(\Delta x_{(8)}) = \begin{cases} 1 , & \text{for } \Delta x_{(8)} \neq 0 , \\ 0 , & \text{for } \Delta x_{(8)} = 0 . \end{cases}$$

遷移 $\chi(\Delta X) \rightarrow \chi(\Delta Y)$ に対する truncated 差分確率を $\Pr(\chi(\Delta X) \rightarrow \chi(\Delta Y))$ と書く。

次に、経路探索の計算量削減のため、$\chi(\Delta X_{(32)})$ のハミング重みである truncated ハミング重み (以下、THD) $\eta(X_{(32)})$ とその特性確率 $\Pr(\cdots)$ を導入する。

Table 4: mds_{L} 関数の THD 確率 (確率の 2 べき近似を利用)

$\eta(\Delta X_{(32)})$ \ $\eta(\Delta Y_{(32)})$	0	1	2	3	4
0	1	0	0	0	0
1	0	0	0	0	1
2	0	0	0	2^{-8}	1
3	0	0	2^{-16}	2^{-8}	1
4	0	2^{-24}	2^{-16}	2^{-8}	1

$$\eta(\Delta X_{(32)}) = \sum_{i=1}^{4} \delta(\Delta x_{i(8)}) .$$

$$\Pr(\eta(\Delta X_{(32)}) \rightarrow \eta(\Delta Y_{(32)})) = \max_{\substack{\chi(\Delta X'_{(32)}),\ \eta(\Delta X'_{(32)}) = \eta(\Delta X_{(32)}), \\ \chi(\Delta Y'_{(32)}),\ \eta(\Delta Y'_{(32)}) = \eta(\Delta Y_{(32)})}} \Pr(\chi(\Delta X'_{(32)}) \rightarrow \chi(\Delta Y'_{(32)})) .$$

THD とその遷移確率は次式のように、$32m$ ビット・データに自然に一般化出来る。

$$\eta(\Delta X_{(32m)}) = \sum_{i=1}^{m} \eta(\Delta X_{i(32)}) 5^{m-1-i} ,$$

$$\Pr(\eta(\Delta X_{(32m)}) \rightarrow \eta(\Delta Y_{(32m)})) = \prod_{i=1}^{m} \Pr(\eta(\Delta X_{i(32)}) \rightarrow \eta(\Delta Y_{i(32)})) .$$

mds_{L} 関数と MDS_{L} 関数において、truncated 差分確率と THD 確率は一致する。

$$\Pr(\chi(\Delta X_{(32)}) \rightarrow \chi(\Delta Y_{(32)})) = \Pr(\eta(\Delta X_{(32)}) \rightarrow \eta(\Delta Y_{(32)})) ,$$

$$\Pr(\chi(\Delta X_{(128)}) \rightarrow \chi(\Delta Y_{(128)})) = \Pr(\eta(\Delta X_{(128)}) \rightarrow \eta(\Delta Y_{(128)})) .$$

4.3.2 構成要素の特性の検討

S-box: truncated 差分攻撃では、S-box はランダムな全単射写像であることが前提となる。Hierocrypt-3 の S-box は、最大差分／線形確率が最小値、代数次数が 7 次、多項式表現での項数が十分多いので、十分条件を満たす。

mds_{L} 関数: Hierocrypt-3 では MDS 写像なので、truncated 差分確率と THD は一致し、その値は松井の近似計算アルゴリズムにより、2 のべき乗で近似できる [9, 4]。表 4 に近似結果を示す。

MDS_{H} 関数: バイト単位の排他的論理和だけで構成されるので、松井による近似計算アルゴリズムによって、truncated 差分確率の 2 べき乗で表わされる近似評価が得られる [4]。

4.3.3 多段に対する評価

Hierocrypt-3 では、S-box と鍵加算を除くと、MDS_{H} 関数と MDS_{L} 関数が交互に並ぶ。両端が MDS_{L} 関数の段構造 (LHL$\cdots$HL) では、最大 truncated 差分特性確率は最大 THD 特性確率に一致し、MDS_{L} 関数と MDS_{H} 関数の THD 確率から計算できる。

THD を利用すると、従来の表サイズ約 2^{32} が、約 $5^8 (\simeq 2^{18.58})$ と大幅に節約できる。

以下に、多段に対する解析の手順を示す。

1. mds_{L} の THD 確率表を作成。
2. MDS_{H} の THD 確率表を作成。
3. LH(MDS_{L} 関数の後に MDS_{H} 関数を適用) の THD 確率表を作成。
4. (LH) の t 乗の確率表を順次作成。
5. 前項に L を掛けて、$(t+1)$ 段の THD 確率表を作成。
6. ランダム行列との差が無くなる段数を特定。

上記を実行の結果、Hierocrypt-3 では 3 段 (LHLHL) で truncated 差分特性確率がランダム行列と一致した。よって、2 段攻撃を仮定すると 5 段で十分安全と評価できる。

4.4 その他の攻撃

高階差分攻撃 高階差分攻撃法は、高階差分の性質を利用して連立方程式を立てて解くことで、鍵を特定する代数的攻撃法である [3]。

Hierocrypt-3 では、S-box の代数次数が 7 次であり、S-box と mds_{L} を組み合わせた合成関数の多項式表現の項数を最大化しているので、効率的な高階差分が存在する可能性は極めて低い。

補間攻撃 Hierocrypt-3 の場合、S-box には $\mathrm{GF}(2^8)$ 上のべき乗演算を使ってはいるものの、入力側でのビット置換を組み合わせているため代数構造が複雑化されている。さらに、下位拡散層と組合せたときの代数表現での項数が最大になるように選んであるので、補間攻撃の適用は困難と期待できる。

Impossible Differential 攻撃 MDS_{H} の複数経路性が有るので、Hierocrypt-3 は SQUARE や Rijndael より Impossible Differential は少なく、この攻撃に対して強いと推定できる。フルスペックの SQUARE や Rijndael が解かれたという報告も無いので、当攻撃に対して安全と考えられる。

Non-surjective 攻撃 Hierocrypt-3 では全部の構成要素が全単射なので、この攻撃は適用不可能である。

Mod n 攻撃 Hierocrypt-3 では全単射の構成要素のみで構成されるので、このような攻撃は適用不可能である。

Appendix

A Algorithm of Hierocrypt-3

A.1 Notations

An n-bit value is basically expressed with the subscript (n). For example, the value $X_{(n)}$ is an element of $\mathrm{GF}(2)^n$. A value expressed by a capital(s) describes an element of no less than 16 bits A value expressed by a small letter(s) describes an element of less than 16 bits.

We adopt the bigendian convention. When a value $X_{(mn)}$ is expressed as a concatenation of m pieces of n-bit length, each piece is expressed with the subscript $i(n)$ $(i = 1, 2, \ldots, m)$, that is $X_{(mn)} = X_{1(n)} \| X_{2(n)} \| \cdots \| X_{m(n)}$. Furthermore, $X_{(mn)}$ and $X_{i(n)}$ are expressed as the following concatenations: $X_{(mn)} = x_{1(1)} \| x_{2(1)} \| \cdots \| x_{mn(1)}$, $X_{i(n)} = x_{ni-n+1(1)} \| x_{ni-n+2(1)} \| \cdots \| x_{ni-n+n(1)}$. The following shows an example of concatenation expression of a 128-bit value $X_{(128)}$.

$$X_{(128)} = X_{1(32)} \| X_{2(32)} \| X_{3(32)} \| X_{4(32)} ,$$

$$X_{i(32)} = x_{4i-4+1(8)} \| x_{4i-4+2(8)} \| \cdots \| x_{4i-4+4(8)} , i = 1, 2, \ldots, 4,$$

$$X_{j(8)} = x_{8j-8+1(1)} \| x_{8j-8+2(1)} \| \cdots \| x_{8j-8+8(1)} , j = 1, 2, \ldots, 16.$$

Note that the LSB of the value $X_{i(n)}$ (i=1,2,...,m) is $x_{in(1)}$, which is the in-th MSB of $X_{(mn)}$.

B Structure of the algorithm

The structures of data randomization part and the key scheduling part are described in this section. Fundamental operations used there are described in the next section.

B.1 Encryption

The T-round encryption of Hierocrypt-3 consists of $(T-1)$ operations of round function ρ, an operation of XS-function, and the final key addition (AK) as shown below.

$$P_{(128)} \equiv X^{(0)}_{(128)} \overset{\rho}{\longmapsto} X^{(1)}_{(128)} \overset{\rho}{\longmapsto} \cdots \overset{\rho}{\longmapsto} X^{(T-1)}_{(128)} \overset{XS}{\longmapsto} X^{(T)}_{(128)} \overset{AK}{\longmapsto} C_{(128)}$$

T is 6,7, or 8 for 128-, 192-, or 256-bit, respectively.

The 128-bit value $X^{(i)}_{(128)}$ is the output of the i-th operation of round function ρ $(i = 1, 2, \ldots, T-1)$. The plaintext $P_{(128)}$ is assigned to the 0-th value $X^{(0)}_{(128)}$.

The value $X^{(t)}_{(128)}$ is the output of the t-th operation of ρ-function for the input $X^{(t-1)}_{(128)}$ and the round key $K^{(t)}_{(256)}$.

$$X^{(t)}_{(128)} = \rho(X^{(t-1)}_{(128)}, K^{(t)}_{(256)}), \quad t = 1, 2, \cdots, T-1.$$

Similarly, $X^{(T)}_{(128)}$ is the output of XS-function for the input $X^{(T-1)}_{(128)}$ and the final key $K^{(T)}_{(256)}$.

$$X^{(T)}_{(128)} = XS(X^{(T-1)}_{(128)}, K^{(T)}_{(256)}) .$$

The ciphertext $C_{(128)}$ is given as the addition (XOR, exclusive or) between the T-th round output $X^{(T)}_{(128)}$ and the first half of the final key $K^{(T+1)}_{1(128)}$

$$C_{(128)} = X^{(T)}_{(128)} \oplus (K^{(T+1)}_{1(64)} \| K^{(T+1)}_{2(64)}) .$$

B.2 decryption

The decryption of Hierocrypt-3 is the inverse of encryption, and consists of the final key addition, the inverse of XS-function (XS^{-1}), and $(T-1)$ inverse operations of round function (ρ^{-1}).

$$\begin{aligned}
X^{(T)}_{(128)} &= C_{(128)} \oplus (K^{(T+1)}_{1(64)} \| K^{(T+1)}_{2(64)}) , \\
X^{(T-1)}_{(128)} &= XS^{-1}(X^{(T)}_{(128)}, K^{(T)}_{(256)}) , \\
X^{(t-1)}_{(128)} &= \rho^{-1}(X^{(t)}_{(128)}, K^{(t)}_{(256)}) , \ t = T-1, \ldots, 2, 1.
\end{aligned}$$

The plaintext $P_{(128)}$ is given as the final output $X^{(0)}_{(128)}$.

$$P_{(128)} = X^{(0)}_{(128)} .$$

B.3 Key scheduling

The main part of key scheduling consists of the intermediate key generation part and the round key generation part, preceded by the intermediate key initialization. The intermediate key part recursively generates intermediate key outputs $Z^{(t)}_{(256)}$ $(t = 1, 2, \ldots, T+1)$, and the round key generation part gerates round keys $K^{(t)}_{(256)}$ $(t = 1, 2, \ldots, T+1)$ from the corresponding intermediate keys, as follows.

$$\begin{aligned}
&K_{(\mathrm{length})} \overset{\mathrm{padding}}{\longmapsto} Z^{(-1)}_{(256)} \overset{\sigma_0}{\longmapsto} Z^{(0)}_{(256)} \overset{\sigma}{\longmapsto} Z^{(1)}_{(256)} \\
&\overset{\sigma}{\longmapsto} Z^{(2)}_{(256)} \overset{\sigma}{\longmapsto} \cdots \overset{\sigma}{\longmapsto} Z^{(t_{\mathrm{turn}})}_{(256)} \\
&\overset{\sigma^{-1}}{\longmapsto} Z^{(t_{\mathrm{turn}}+1)}_{(256)} \overset{\sigma^{-1}}{\longmapsto} \cdots \overset{\sigma^{-1}}{\longmapsto} Z^{(T+1)}_{(256)}
\end{aligned}$$

The intermediate key $Z^{(t)}_{(128)}$ and the round key $K^{(t)}_{(128)}$ are divided into 4 pieces. $Z^{(t)}_{(128)}$ and $K^{(t)}_{(128)}$ are divided into 4 pieces.

$$\begin{aligned}
Z^{(t)}_{(256)} &= Z^{(t)}_{1(64)} \| Z^{(t)}_{2(64)} \| Z^{(t)}_{3(64)} \| Z^{(t)}_{4(64)} , \\
K^{(t)}_{(256)} &= K^{(t)}_{1(64)} \| K^{(t)}_{2(64)} \| K^{(t)}_{3(64)} \| K^{(t)}_{4(64)} .
\end{aligned}$$

To generate the intermediate keys, the σ-function is used for $1 \le t \le t_{\mathrm{turn}}$, and the σ^{-1}-function is used for $t_{\mathrm{turn}}+1 \le t \le T+1$, where $t_{\mathrm{turn}} = 4$ for the 128/192-bit key and where $t_{\mathrm{turn}} = 5$ for the 256-bit key. Under the recursion rule, the intermediate key values are symmetric with regard to the point $t = t_{\mathrm{turn}}$.

$$Z^{(t)}_{(256)} = Z^{(2t_{\mathrm{turn}}-t)}_{(256)} , \quad t_{\mathrm{turn}}+1 \le t \le T+1 .$$

B.3.1 round-dependent constants

To prevent periodic patterns from appearing in the intermediate key generation, and to improve resistance against the related key attack, we introduce round-dependent key additions to the intermediate key generation part. The round-dependent keys have been made by combining two from the four 32-bit values which are given as binary expansions of irrational numbers.

$$\begin{aligned}
H_0 &= \ \mathrm{0x5A827999} = \ trunc(\sqrt{2}/4), \\
H_1 &= \ \mathrm{0x6ED9EBA1} = \ trunc(\sqrt{3}/4), \\
H_2 &= \ \mathrm{0x8F1BBCDC} = \ trunc(\sqrt{5}/4), \\
H_3 &= \ \mathrm{0xCA62C1D6} = \ trunc(\sqrt{10}/4),
\end{aligned}$$

$$trunc(x) = \left\lfloor 2^{32} x \right\rfloor ,$$

$$\begin{aligned}
G_0(0) &= H_3 \| H_0 , \quad G_0(1) = H_2 \| H_1 , \\
G_0(2) &= H_1 \| H_3 , \quad G_0(3) = H_0 \| H_2 , \\
G_0(4) &= H_2 \| H_3 , \quad G_0(5) = H_1 \| H_0 .
\end{aligned}$$

B.3.2 Preprocessing

The intermediate key $Z^{(-1)}_{(256)}$ is made of the encryption key $K_{(\text{length})}$ (length = 128, 192, 256) with an initial operation. The intermediate key $Z^{(0)}_{(256)}$ is deriven from $Z^{(-1)}_{(256)}$ through the pre-whitening operation σ_0. The padding operation is done when length = 192, 256 where padded values are concatenations of the above mentioned 32-bit constants H_i. The padding operations are described as follows.

[128-bit key]

$$K_{1(64)} \| K_{2(64)} = K_{(128)} ,$$

$$Z^{(-1)}_{1(64)} = K_{1(64)} , \quad Z^{(-1)}_{2(64)} = K_{2(64)} ,$$

$$Z^{(-1)}_{3(64)} = K_{1(64)} , \quad Z^{(-1)}_{4(64)} = H_3 \| H_2 .$$

[192-bit key]

$$K_{1(64)} \| K_{2(64)} \| K_{3(64)} = K_{(192)} ,$$

$$Z^{(-1)}_{1(64)} = K_{1(64)} , \quad Z^{(-1)}_{2(64)} = K_{2(64)} ,$$

$$Z^{(-1)}_{3(64)} = K_{3(64)} , \quad Z^{(-1)}_{4(64)} = H_2 \| H_3 .$$

[256-bit key]

$$K_{1(64)} \| K_{2(64)} \| K_{3(64)} \| K_{4(64)} = K_{(256)} ,$$

$$Z^{(-1)}_{1(64)} = K_{1(64)} , \quad Z^{(-1)}_{2(64)} = K_{2(64)} ,$$

$$Z^{(-1)}_{3(64)} = K_{3(64)} , \quad Z^{(-1)}_{4(64)} = K_{4(64)} .$$

[pre-whitening](ρ_0)

The pre-whitening is done, before iterative operation by the σ-function. The pre-whitening operation ρ_0 is made from ρ by removing $P^{(32)}$.

$$Z^{(0)}_{(256)} = \sigma_0(Z^{(-1)}_{(256)}, G^{(0)}_{(64)}),$$

$$Z^{(0)}_{3(64)} = M_{5E}(Z^{(-1)}_{3(64)}) \oplus G^{(0)}_{(64)} ,$$
$$Z^{(0)}_{4(64)} = M_{5E}(Z^{(-1)}_{4(64)}) ,$$
$$Z^{(0)}_{1(64)} = Z^{(-1)}_{2(64)} ,$$
$$Z^{(0)}_{2(64)} = Z^{(-1)}_{1(64)} \oplus F_\sigma(Z^{(-1)}_{2(64)} \oplus Z^{(0)}_{3(64)}) .$$

As the round-dependent constant $G^{(0)}_{(64)}$, the following 64-bit concatenated value is used.

$$G^{(0)}_{(64)} = G_0(5) = H_1 \| H_0 .$$

B.3.3 Intermediate key update (σ-function)

The intermediate key $Z^{(t)}_{(256)}$ is generated by the operation σ up to $t = t_{\text{turn}}$, and afterwards by the inverse operation σ^{-1}. The sequence of intermediate keys is symmetric with respect to the point $t = t_{\text{turn}}$ for this round-trip-type scheduling.

$$Z^{(t)}_{(256)} = Z^{(2t_{\text{turn}}-t)}_{(256)} , \quad t_{\text{turn}} \le t \le T+1 .$$

We call the region: $(1 \le t \le t_{\text{turn}})$ as the plaintext side, and the other region: $(t_{\text{turn}}+1 \le t \le T+1)$ as the ciphertext side, corresponding to the position in the data randomizing part.

[Iteration of intermediate key(plaintext side)] $(1 \le t \le t_{\text{turn}})$

$$Z^{(t)}_{(256)} = \sigma(Z^{(t-1)}_{(256)} , \ G^{(t)}_{(64)}) ,$$
$$W^{(t-1)}_{1(64)} \| W^{(t-1)}_{2(64)} = P^{(32)}(Z^{(t-1)}_{3(64)} \| Z^{(t-1)}_{4(64)}) ,$$
$$Z^{(t)}_{3(64)} = M_{5E}(W^{(t-1)}_{1(64)}) \oplus G^{(t)}_{(64)} ,$$
$$Z^{(t)}_{4(64)} = M_{5E}(W^{(t-1)}_{2(64)}) ,$$
$$Z^{(t)}_{1(64)} = Z^{(t-1)}_{2(64)} ,$$
$$Z^{(t)}_{2(64)} = Z^{(t-1)}_{1(64)} \oplus F_\sigma(Z^{(t-1)}_{2(64)} \oplus Z^{(t)}_{3(64)}) .$$

[Iteration of intermediate key(ciphertext side)] $(t_{\text{turn}}+1 \le t \le T+1)$

$$Z^{(t)}_{(256)} = \sigma^{-1}(Z^{(t-1)}_{(256)}, G^{(t-1)}_{(64)}) ,$$
$$Z^{(t)}_{1(64)} = Z^{(t-1)}_{2(64)} \oplus F_\sigma(Z^{(t-1)}_{1(64)} \oplus Z^{(t-1)}_{3(64)}) ,$$
$$Z^{(t)}_{2(64)} = Z^{(t-1)}_{1(64)} ,$$
$$W^{(t)}_{1(64)} = M_{B3}(Z^{(t-1)}_{3(64)} \oplus G^{(t-1)}_{(64)}) ,$$
$$W^{(t)}_{2(64)} = M_{B3}(Z^{(t-1)}_{4(64)}) ,$$
$$Z^{(t)}_{3(64)} \| Z^{(t)}_{4(64)} = P^{(32)-1}(W^{(t)}_{1(64)} \| W^{(t)}_{2(64)}) .$$

B.3.4 Round key generation

The different rules are applied to generate a round key from the corresponding intermediate key for the plaintext side and the ciphertext side.

[Round key generation(plaintext side)] $(1 \le t \le t_{\text{turn}})$

$$V^{(t)}_{(64)} = F_\sigma(Z^{(t-1)}_{2(64)} \oplus Z^{(t)}_{3(64)}) ,$$
$$K^{(t)}_{1(64)} = Z^{(t-1)}_{1(64)} \oplus V^{(t)}_{(64)} , \quad K^{(t)}_{2(64)} = Z^{(t)}_{3(64)} \oplus V^{(t)}_{(64)} ,$$
$$K^{(t)}_{3(64)} = Z^{(t)}_{4(64)} \oplus V^{(t)}_{(64)} , \quad K^{(t)}_{4(64)} = Z^{(t-1)}_{2(64)} \oplus Z^{(t)}_{4(64)} .$$

[Round key generation(ciphertext side)] $(t_{\text{turn}}+1 \le t \le T+1)$

$$V^{(t)}_{(64)} = F_\sigma(Z^{(t-1)}_{1(64)} \oplus Z^{(t-1)}_{3(64)}) ,$$
$$K^{(t)}_{1(64)} = Z^{(t)}_{1(64)} \oplus Z^{(t-1)}_{3(64)} , \quad K^{(t)}_{2(64)} = W^{(t)}_{1(64)} \oplus V^{(t)}_{(64)} ,$$
$$K^{(t)}_{3(64)} = W^{(t)}_{2(64)} \oplus V^{(t)}_{(64)} , \quad K^{(t)}_{4(64)} = Z^{(t-1)}_{1(64)} \oplus W^{(t)}_{2(64)} .$$

Table 5: Key schedule for 128-bit key (6 rounds)

round key	t	operation	$G^{(t)}_{(64)}$
–	−1 (PAD)	–	$H_3 \| H_2$
–	0 (PW)	σ_0	$G_0(5)$
$K^{(1)}_{(256)}$	1	σ	$G_0(0)$
$K^{(2)}_{(256)}$	2	σ	$G_0(1)$
$K^{(3)}_{(256)}$	3	σ	$G_0(2)$
$K^{(4)}_{(256)}$	4	σ	$G_0(3)$
$K^{(5)}_{(256)}$	5	σ^{-1}	$G_0(3)$
$K^{(6)}_{(256)}$	6	σ^{-1}	$G_0(2)$
$K^{(7)}_{(256)}$	7	σ^{-1}	$G_0(1)$

Table 6: Key schedule for 192-bit key (7 rounds)

round key	t	operation	$G^{(t)}_{(64)}$
–	−1 (PAD)	–	$H_2 \| H_3$
–	0 (PW)	σ_0	$G_0(5)$
$K^{(1)}_{(256)}$	1	σ	$G_0(1)$
$K^{(2)}_{(256)}$	2	σ	$G_0(0)$
$K^{(3)}_{(256)}$	3	σ	$G_0(3)$
$K^{(4)}_{(256)}$	4	σ	$G_0(2)$
$K^{(5)}_{(256)}$	5	σ^{-1}	$G_0(2)$
$K^{(6)}_{(256)}$	6	σ^{-1}	$G_0(3)$
$K^{(7)}_{(256)}$	7	σ^{-1}	$G_0(0)$
$K^{(8)}_{(256)}$	8	σ^{-1}	$G_0(1)$

Table 7: Key schedule for 256-bit key (8 rounds)

round key	t	operation	$G^{(t)}_{(64)}$
–	−1 (PAD)	–	–
–	0 (PW)	σ_0	$G_0(5)$
$K^{(1)}_{(256)}$	1	σ	$G_0(4)$
$K^{(2)}_{(256)}$	2	σ	$G_0(0)$
$K^{(3)}_{(256)}$	3	σ	$G_0(2)$
$K^{(4)}_{(256)}$	4	σ	$G_0(1)$
$K^{(5)}_{(256)}$	5	σ	$G_0(3)$
$K^{(6)}_{(256)}$	6	σ^{-1}	$G_0(3)$
$K^{(7)}_{(256)}$	7	σ^{-1}	$G_0(1)$
$K^{(8)}_{(256)}$	8	σ^{-1}	$G_0(2)$
$K^{(9)}_{(256)}$	9	σ^{-1}	$G_0(0)$