修士論文要旨 (2013 年度)

ブロック暗号に対する中間一致攻撃の計算量削減に関する研究
On Complexity Reduction of Meet-in-the-Middle Attack

情報工学専攻　伊藤 祐樹
Yuki ITO

1　はじめに

中間一致攻撃は 1977 年に Diffie と Hellman によって提案された攻撃手法である [1]. 中間一致攻撃は, ブロック暗号を前半部と後半部に分割し, 双方から暗号化・復号を行い, 中間値の一致を確認することで鍵を絞り込む攻撃方法である.

本稿では中間一致攻撃における暗号化・復号による中間値の計算を分割して行うことで計算量を削減する. また改良した手法を KATAN に適用した結果を報告する.

第 2 節では中間一致攻撃の概要について説明する. 第 3 節では中間一致攻撃の手順の改良について述べる. そして第 4 節で KATAN に対する適用を述べ, 第 5 節でまとめとする.

2　中間一致攻撃

本節では中間一致攻撃の基本的なフレームワークについて述べる.

中間一致攻撃は大きく 2 つのパートに分割して考える. 中間値の一致の確認を行う Meet-in-the-Middle(MITM) ステージと, MITM ステージで出力された鍵候補を一つに絞り込む Key-Testing ステージである.

ブロック長 b ビット, l ビットの秘密鍵を持つ暗号を考える. $F1(k,x)$ を前半部の暗号化, $F2(k,y)$ を後半部の復号とする. また K_1 を $F1(k,x)$ での暗号化に使用する鍵とし, K_2 を $F2(k,y)$ での復号に使用する鍵とする. また K_1, K_2 の部分集合を次のように定義する.

$$A_0 = K_1 \cap K_2 \tag{1}$$
$$A_1 = K_1 \backslash K_1 \cap K_2 \tag{2}$$
$$A_2 = K_2 \backslash K_1 \cap K_2 \tag{3}$$

平文を p, その暗号文を c としたとき, MITM ステージの手順は次のようになる.

A_0 の全通りについて次を行う.

1. A_0 と A_1 の全てのパタンについて K_1 を計算し, $v = F1(K_1, p)$ を求める. K_1 を v で項目付けられたテーブル T に保存する.
2. A_0 と A_2 の全てのパタンについて K_2 を計算し, $u = F2(K_2, c)$ を求める. K_2 を v で項目付けられたテーブル T' に保存する.
3. $v = u$ を満たす K_1, K_2 のペアを T と T' からそれぞれ選び鍵候補として κ に加える.

$|a|$ を a のビット長とする. $|v| = |u| = m$ ビットと仮定した場合, MITM ステージにおける鍵候補は $2^{|A_0|} \cdot 2^{|A_1|+|A_2|} \cdot 2^{-m}$ となる. MITM ステージで得られた鍵候補の集合 κ を Key-Testing ステージで絞り込む. Key-Testing ステージは, 新たな平文・暗号文ペアを用意し鍵候補を用いて平文を暗号化する. そして正しい暗号文が出力されるかどうかを確認することで鍵を一つに絞り込む.

Key-Testing ステージで使用する平文・暗号文ペアを N 本とすると中間一致攻撃における計算量は $2^{|A_0|} \cdot (2^{|A_1|} + 2^{|A_2|}) + 2^{l-m} + 2^{l-2m} + \cdots + 2^{l-Nm}$ である.

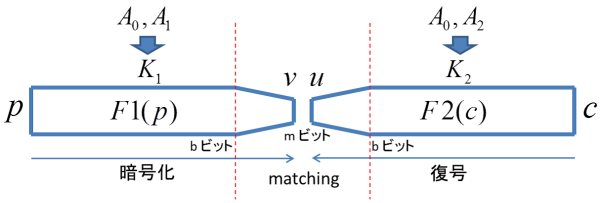

図 1　中間一致攻撃の概略

3　中間一致攻撃の計算量削減方法

本節では MITM ステージにおける計算量削減について示す.

3.1　分割による計算量削減

平文を p とし, 平文から前半部の暗号化処理によって求まる中間値を v とする. MITM ステージにおける前半分の $F1$ をさらに F_{f_1} と F_{f_2} に分割する.

$$g_1 = F_{f_1}(k_{f_1}, p) \tag{4}$$
$$v = F_{f_2}(k_{f_2}, g_1) \tag{5}$$

ただし, k_{f_1} と k_{f_2} はそれぞれ独立な鍵とする. p を k_{f_1} の全通りの鍵で暗号化を行い g_1 を求めた場合, k_{f_1} のパタンだけ g_1 が存在する. すると $|k_{f_1}| > |g_1|$ であった場合, 必ず衝突が存在することになる. k_{f_1}, k_{f_2} を使用して v を計算する場合を考えると, k_{f_1} と k'_{f_1} が同じ g_1 を出力するならば g_1 以降は同じ計算をすることになる. つまり衝突によって重複する計算を省くことで計算量を下げることができる.

復号方向も同様に, 暗号文を c, 暗号文から後半部を復号することによって得られる中間値を u とし,

$$g_2 = F_{b_1}(k_{b_1}, c) \tag{6}$$
$$u = F_{b_2}(k_{b_2}, g_2) \tag{7}$$

とする．

ここで計算量が最良となる分割方法について示す．$F1$ の暗号化処理で使用する鍵を k_f とすると $|k_f| = |k_{f_1}| + |k_{f_2}|(|k_{f_1}| > 0$ かつ $|k_{f_2}| > 0)$ である．k_f を総当たりして v を計算するときの計算量は $2^{|k_f|}$, 式 (4), (5) において分割した鍵を総当たりして v を計算するときの計算量は $2^{|k_{f_1}|} + 2^{|g_1|+|k_{f_2}|}$ である．このとき, $|k_f| > |g_1| + |k_{f_2}|$ であれば, 分割した場合の計算量は小さくなる．また, 分割した場合の計算量が最小となるのは $|k_{f_1}| = |g_1| + |k_{f_2}|$, すなわち $|k_{f_1}| = b + |k_{f_2}|$ の場合である．よって

$$|k_{f_1}| = \frac{|k_f| + b}{2}$$
$$|k_{f_2}| = \frac{|k_f| - b}{2} \tag{8}$$

と分割する場合, 計算量が最も小さくなる．

例えば, $|k_f| = 128$, $b = 64$ ならば, $|k_{f_1}| = 96$, $|k_{f_2}| = 32$ とした場合, 式 (5) の計算量は $2^{96} + 2^{96} = 2^{97}$ となる．

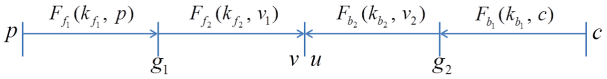

図 2　計算量の削減

3.2　アルゴリズム

本節では, 図 2 のように分割した中間一致攻撃の具体的な手順を示す．

MITM ステージ

1. 全通りの k_{f_1} に対して $g_1 = F_{f_1}(k_{f_1}, p)$ を計算し, テーブル T_1(表 1) を生成する．
2. 全通りの k_{b_1} に対して $g_1 = F_{b_1}(k_{b_1}, c)$ を計算し, テーブル T_2 を生成する．
3. g_1, k_{f_2} 全通り全ての組み合わせに対して $v = F_{f_2}(k_{f_2}, g_1)$ を計算し, テーブル T_v(表 2) を生成する．
4. g_2, k_{b_2} 全通り全ての組み合わせに対して $u = F_{b_2}(k_{b_2}, g_1)$ を計算し, テーブル T_u を生成する．

1 から 4 の合計計算量は全て同じラウンド数の処理とすると, $2^{|k_{f_1}|} + 2^{|k_{b_1}|} + 2^{|g_1|+|k_{f_1}|} + 2^{|g_2|+|k_{b_2}|}$ である．ここから中間ラウンドにおける一致の確認と鍵の絞り込みを行う．一致の確認はまず $v = u$ となる k_{f_1} と k_{b_2} を T_v, T_u から取り出す．さらに対応する g を用いて T_1, T_2 から一つそれぞれ取り出す．これにより得られた k_{f_1}, k_{f_2}, k_{b_1}, k_{b_2} を鍵候補として κ に加える．またテーブル T_1, T_2, T_v, T_u の具体的な中身を表 1, 2 に示す．

Key-Testing ステージ

Key-Testing ステージでは, MITM ステージにおいて構成された κ を別の平文・暗号文ペアを用いていテストする．これを正しい鍵一つに絞り込めるまで行う．

秘密鍵を l ビット, $|v| = |u| = b$ ビットと仮定した場合, κ の要素数は $2^{|k_{f_1}|+|k_{f_2}|+|k_{b_1}|+|k_{b_2}|-b} = 2^{l-b}$ である．したがって Key-Testing ステージにおける計算量は N 本の平文・暗号文ペアで一つに秘密鍵に絞り込めると考えた場合, $2^{l-b} + 2^{l-2b} + \cdots + 2^{l-Nb}$ である．

表 1　$T_1, T_2 (n = 2^{|k_{f_1}|-|g_1|})$

$g_1(g_2)$	$k_{f_1}(k_{b_1})$
0	$k_{f_1}^{(1)}, k_{f_1}^{(2)}, k_{f_1}^{(3)}, \ldots, k_{f_1}^{(n)}$
1	⋮
2	
3	
⋮	
$2^{\|g_1\|} - 1 (2^{\|g_2\|} - 1)$	

表 2　$T_v, T_u (n' = 2^{|g_1|+|k_{f_2}|-|v|})$

$v(u)$	$k_{f_2}(k_{b_2})$	$g_1(g_2)$
0	$f_{f_2}^{(1)}$	$g_1^{(1)}$
	$f_{f_2}^{(2)}$	$g_1^{(2)}$
	⋮	⋮
	$f_{f_2}^{(n')}$	$g_1^{(n')}$
1	⋮	⋮
2		
3		
⋮		
$2^{\|v\|} - 1 (2^{\|u\|} - 1)$		

4　KATAN に対する適用

本節では 3 節で考察した手法を KATAN に対して適用することを考える．

4.1　KATAN

KATAN は 2009 年に CHES で提案されたブロック暗号である [3]．KATAN は論理積, 排他的論理和, ビットシフトの組み合わせによって構成されている．ブロック長は 32, 48, 64 ビットから選択でき, それぞれ KATAN32, KATAN48, KATAN64 と呼ぶ．また鍵長はブロック長によらず 80 ビットでラウンド数は 254 ラウンドである．

KATAN は 2 つのレジスタ $L1$ と $L2$ から構成されており, ビットシフトにより最上位ビットは切り落とされ, f_a と f_b が挿入される．f_a, f_b は非線形関数であり, それぞれ次のように計算される．

$$f_a = L1[x_1] \oplus L1[x_2] \oplus (L1[x_3] \cdot L1[x_4]) \oplus (L1[x_5] \cdot IR) \oplus k_a \tag{9}$$
$$f_b = L2[y_1] \oplus L2[y_2] \oplus (L2[y_3] \cdot L2[y_4]) \oplus (L2[y_5] \cdot L2[y_6]) \oplus k_b \tag{10}$$

k_a, k_b は各ラウンドで使用される副鍵であり, R ラウンドで使用する鍵は k_{2R}, k_{2R+1} である．$L1_a$ は $L1$ の a ビット目を表しており, $L1_0$ は $L1$ の最上位ビットである．また, IR (irregular update rule) は

初期状態を 255 とした 8 ビットの LFSR の出力であり, 規約多項式は $x^8+x^7+x^5+x^3+1$ である. 鍵スケジュールは初期状態を秘密鍵の 80 ビットとし, $k_i = k_{i-80} \oplus k_{i-61} \oplus k_{i-50} \oplus k_{i-13}$ によって生成される. x, y は表 3 のビット位置である. KATAN の全体図を図 3 に示す.

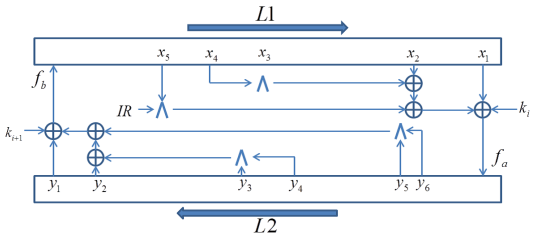

図 3　KATAN の全体図

表 3　KATANn のパラメータ

n	$\|L1\|$	$\|L2\|$	x_1	x_2	x_3	x_4	x_5
32	13	19	12	7	8	5	3
48	19	29	18	12	15	7	6
64	25	39	24	15	20	11	9

n	y_1	y_2	y_3	y_4	y_5	y_6
32	18	7	12	10	8	3
48	28	19	21	13	15	6
64	38	25	33	21	14	9

4.2　KATAN32 に対する Multi-Dimensional 中間一致攻撃

Multi-Dimensional 中間一致攻撃は暗号化アルゴリズムを n 個の区間に分割し, それぞれの区間で中間一致攻撃を適用する手法である (nD-MITM と呼ぶ). 本節では文献 [2] で示された 175 ラウンドに縮小した KATAN32 に対する適用について説明する.

i ラウンドの中間値を s_i とする. すなわち平文・暗号文をそれぞれ p, c とすると, $p = s_0$, $c = s_{175}$ である. 175 ラウンドの KATAN32 を次の 3 つの区間に分割する.

1. 1 から 63(一致ラウンドは 39 ラウンド)
2. 64 から 87(一致ラウンドは 79 ラウンド)
3. 88 から 175(110 ラウンドで Partial-Matching)

区間 1 における暗号化･復号処理を $F_{f_1}(k,x)$, $F_{b_1}(k,y)$ 区間 2 は $F_{f_2}(k,x)$, $F_{b_2}(k,y)$, 区間 3 は $F_{f_3}(k,x)$, $F_{b_3}(k,y)$ とする. 区間 3 では Partial-Matching を行う. s_{110} で一致させるビットは $L2_{16}$, $L2_{18}$ の 2 ビットである. アルゴリズムの詳細は次の通りである.

1. $k_{f_1} = k_{0\ldots77}$ の全通りについて $s_{39} = F_{f_1}(k_{f_1}, s_0)$ を計算する. $k'_{80\ldots125}$ を $k_{80\ldots125}$ に含まれている k_{78}, k_{79} を除いた鍵とし, $k'_{80\ldots125}$ を k_{f_1} から鍵スケジュールを用いて求める. そして $k_{0\ldots77}$ を s_{39} と $k'_{80\ldots125}$ で項目付けられたテーブル T_1 に保存する.
2. $k_{b_3} = k_{272\ldots349}$ の全通りについて $s'_{110} = F_{b_3}(k_{b_3}, s_{175})$ のうち 2 ビットを計算する. 2 ビットで項目付けられた T'_3 に $k_{272\ldots349}$ を保存する.
3. s_{87} と $k_{f_3} = k_{174\ldots219}$ の全通り・全組み合わせについて $s_{110} = F_{f_3}(k_{f_3}, s_{87})$ を計算する. そして s_{87} と k_{f_3} で項目付けられたテーブル T_3 に s_{110} を格納する. このステップ後に T_3 と T'_3 から積集合を構成する.
4. $g_1(= s_{63})$ の全ての値に対して次を行う.
   (a) $k_{b_1} = k_{78\ldots125}$ の全通りについて $s'_{39} = F_{b_1}(k_{b_1}, s_{63})$ を計算する. さらに k_{78} と k_{79} を $k_{80\ldots125}$ から排除することで $k'_{80\ldots125}$ を計算する. 計算した s'_{39} と $k'_{80\ldots125}$ を用いて T_1 から $k_{0\ldots77}$ を取り出し, $k_{0\ldots79}$ を得る. 次に $k_{0\ldots79}$ をもとに $k_{126\ldots173}$ を計算し, $k_{126\ldots173}$ で項目付けられたテーブル S に保存する. この時点で S の各項目には平均 1 個の要素が存在する.
   (b) $k_{f_2} = k_{126\ldots157}$ の全通りについて $s_{79} = F_{f_2}(k_{f_2}, s_{63})$ を計算する. そして s_{79} で項目付けられたテーブル T_2 に $k_{126\ldots157}$ を保存する.
   (c) $g_2(= s_{87})$ の全ての値に対して次を行う.
      i. $k_{b_2} = k_{158\ldots173}$ の全通りについて $s'_{79} = F_{b_2}(k_{b_2}, s_{87})$ を計算する. s'_{79} を用いて T_2 から一致する $k_{126\ldots157}$ を見つけ, $k_{126\ldots173}$ を得る. 次に, $k_{126\ldots173}$ を用いて S から $k_{0\ldots79}$ を取り出す. 最後に $k_{0\ldots79}$ を用いて $k_{174\ldots219}$ と $k_{272\ldots349}$ を計算する. そして T_3 と T'_3 の直積集合に求めた副鍵が存在するかを確認し, 存在した場合は鍵候補として出力する.

出力した鍵候補を別の平文・暗号文ペアを用いて鍵候補を一つに絞り込む. この攻撃の合計計算量は次の通りである.

$$2^{78}\cdot\frac{39}{175}+2^{78}\cdot\frac{65}{175}+2^{32+46}\cdot\frac{23}{175}+2^{32}\cdot\left(2^{48}\cdot\frac{24}{175}+2^{32}\cdot\frac{16}{175}+2^{32+16}\cdot\frac{8}{175}\right)+2^{80-2}\approx 2^{79.30}. \quad (11)$$

また MITM ステージにおける計算量は $2^{78.54}$ で, 3 本の既知平文・暗号文を用いることにより鍵を一つに絞り込むことができる.

4.3　3D-MITM に対する分割の適用

文献 [2] で示された 3D-MITM 攻撃の区間 1 と区間 3 に, 3 節で示した分割手法を適用することで, 計算量が削減できることを示す. 区間 2 はラウンド数が少なく, 効率化が見込めないため除外する.

4.3.1　適切な分割ラウンド　最初に区間 1 における暗号化方向で分割するラウンドについて示す. 分割するラウンドは, 式 (8) より, $k_f = 78$, $b = 32$ であるから $\frac{78+32}{2} = 55$ ビット目, すなわち F_{f_1} を 28 ラウンドで分割すると計算量が最小になる. 同様の方法で各関数における適切な分割を考える. すると区間 1 の F_{b_1}

では 43 ラウンド, 区間 3 は F_{b_3} では 107 ラウンドで分割する. F_{b_3} では 147 ラウンドで分割する. イメージを図 4, 5 に示す.

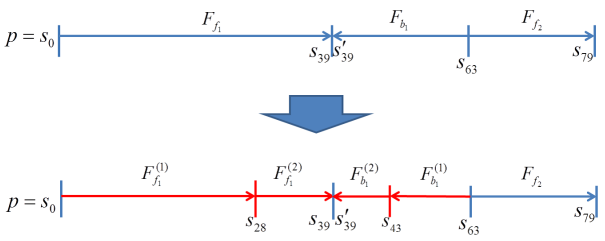

図 4　区間 1 における分割

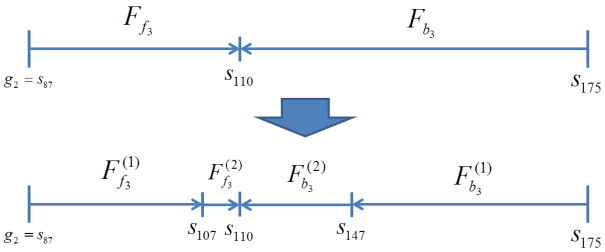

図 5　区間 3 における分割

4.3.2　解析手順　本節では文献 [2] の 3D-MITM に対して 4.3.1 節の分割を適用した場合の解析手順を示す. ただしテーブル T_1, T_3, T'_3, S は 4.2 節で述べられているテーブルと同じものとする.

1. $k^{(1)}_{f_1} = k_{0\ldots55}$ の全通りについて $s_{28} = F^{(1)}_{f_1}(k^{(1)}_{f_1}, s_0)$ を計算する. 計算結果を s_{28} で項目付けられたテーブル $T_{s_{28}}$ に入れる. s_{28}, $k^{(2)}_{f_1} = k_{56\ldots77}$ の組み合わせ全通りに対して $s_{39} = F^{(2)}_{f_1}(k^{(2)}_{f_1}, s_{28})$ を計算し, 結果を s_{39} で項目付けられたテーブル $T_{s_{39}}$ に保存する. $T_{s_{28}}$, と $T_{s_{39}}$ から k_{f_1} を取り出し, $k'_{80\ldots125}$ を計算することでテーブル T_1 を構成する.
2. $k^{(1)}_{b_3} = k_{294\ldots349}$ の全通りについて $s_{147} = F^{(1)}_{b_3}(k^{(1)}_{b_3}, s_{175})$ を計算する. 計算結果を s_{147} で項目付けられたテーブル $T_{s_{147}}$ に入れる. s_{147}, $k^{(2)}_{b_3} = k_{272\ldots293}$ の組み合わせ全通りについて $s_{136} = F^{(2)}_{b_3}(k^{(2)}_{b_3}, s_{147})$ を計算し, 結果を s_{136} で項目付けられたテーブル $T_{s_{39}}$ に保存する. $T_{s_{147}}$ と $T_{s_{136}}$ から k_{b_1} を取り出し, T'_3 を計算する.
3. s_{87} の全ての値に対して次を行う.
   $k^{(1)}_{f_3} = k_{172\ldots211}$ の全通りについて $s_{107} = F^{(1)}_{f_3}(k^{(1)}_{f_3}, s_{87})$ を計算する. 計算結果を s_{32} で項目付けられたテーブル $T_{s_{32}}$ に入れる. s_{107}, $k^{(2)}_{f_3} = k_{212\ldots219}$ の組み合わせ全通りについて $s_{110} = F^{(2)}_{f_3}(k^{(2)}_{f_3}, s_{107})$ を計算し, 結果を s_{110} で項目付けられたテーブル $T_{s_{110}}$ に保存する. $T_{s_{32}}$ と $T_{s_{110}}$ から k_{f_3} を取り出し, T_3 を構成する.
4. $g_2(= s_{63})$ の全ての値に対して次を行う.
   (a) $k^{(1)}_{b_1} = k_{78\ldots117}$ の全通りについて $s_{43} = F^{(1)}_{b_1}(k^{(1)}_{b_1}, s_{63})$ を計算する. 計算結果を $T_{s_{43}}$ に保存する. さらに s_{43}, $k^{(2)}_{b_1} = k_{118\ldots125}$ の全通りの組み合わせを用いて $s'_{39} = F^{(2)}_{b_1}(k^{(2)}_{b_1}, s_{43})$ を計算し, 結果を $T_{s'_{39}}$ に保存する. $T_{s_{43}}$ と $T_{s'_{39}}$ から k_{b_1} を取り出し, 4.2 節と同じ手法で S を構成する.
   (b), (c) については 4.2 節の同じ手順のため省略する.

以上の手順で出力した候補鍵を, Key Testing で別の平文・暗号文ペアを用いてテストすることで鍵を一つに絞り込む. 計算量は

$$2^{73.66} + 2^{75.24} + 2^{69.39} + 2^{73} + 2^{75.55} + 2^{80-2} \approx 2^{78.49}$$

である. また Key-Testing ステージを除いた計算量は $2^{76.72}$ である.

5　まとめ

KATAN に適用した結果, 中間一致攻撃にかかる計算量を $2^{79.30}$ から $2^{78.49}$ に下げることができた. 今回考察した内容は中間一致攻撃の MITM ステージにおける計算量を削減するものである. MITM ステージのみで比較すると $2^{78.54}$ から $2^{76.72}$ に削減できている.

表 4　KATAN に対する適用の結果

	全体の計算量	MITM ステージにおける計算量	文献
削減前	$2^{79.30}$	$2^{78.54}$	[2]
削減後	$2^{78.49}$	$2^{76.72}$	本稿

謝辞

本研究を進めるにあたり，適切な御指導を頂いた中央大学理工学部 趙晋輝教授, 富士通研究所 下山武司氏, NEC 研究所 洲崎智保氏, 峯松一彦氏, 角尾幸保氏に深く感謝致します.

関連発表

1. 伊藤祐樹, 洲崎智保, 峯松一彦, 角尾幸保, 趙晋輝, “中間一致攻撃の計算量削減法の提案”, SCIS 2014.

参考文献

[1] Diffie, W., Hellman, M.,“Exhaustive cryptanalysis of the nbs data encryption standard”, Computer 10 , pp74-84, 1977.

[2] Bo Zhu and Guang Gong,“Multidimensional Meet-in-the-Middle Attack and Its Applications to KATAN/32/48/64”, eprint, 2011.

[3] Canniere, C.D, Dunkelman, O., Knezevic, M.,“Katan and ktantan - a family of small and efficient hardware-oriented block ciphers.”, CHES 2009, LNCS 5747, pp272-288, Springer 2009.

[4] A. Bogdanov and C. Rechberger, “A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN”, SAC 2010, LNCS 6544, pp229-240, Springer 2011.