選択平文攻撃及びサイドチャネル攻撃に対するブロック暗号の耐性評価手法

防衛大学校理工学研究科後期課程

電子情報工学系専攻　情報知能メディア学

小菅　悠久

平成 ２９年 １２月

目　　次

第 1 章

序論

1.1 研究の位置づけ

インターネットを利用する企業や個人レベルの取引がますます発展し，安全な通信環境を保証する暗号は重要度を増している．また，IoT 技術の普及に伴い暗号の用途が多様化している．暗号は用途に合わせた安全性や軽量性を考えて設計し，評価される必要がある．

暗号は大きく共通鍵暗号と公開鍵暗号に分けられ，共通鍵暗号にはブロック暗号とストリーム暗号の 2 種類がある．公開鍵暗号は鍵配送や署名といった用途に用いることができるが，実行時間が大きいという欠点がある．これに対し，共通鍵暗号は暗号化処理を高速に行えるため，大量データの暗号化に用いられる．ブロック暗号はデータを一定の大きさに区切って暗号化するのに対し，ストリーム暗号は擬似乱数列を生成してビットまたはバイト単位で暗号化を行う．本論文ではブロック暗号に注目する．ブロック暗号の概要については付録 C に，本論文で扱う具体的なブロック暗号を付録 D から G に示す．

暗号の安全性評価はアルゴリズム (暗号の設計) 及びその実装でそれぞれ行う．アルゴリズムの安全性評価では，対象のアルゴリズムが安全性の要件を満たしているかを確認する．公開鍵暗号では安全性の根拠となる問題への帰着及びその問題が効率的に解けないことを示す．これに対し，共通鍵暗号ではショートカット法 (第 1.1.1 項参照) である既知の攻撃に対する耐性を確認する．本論文ではショートカット法のひとつである選択平文攻撃に注目する．

実装の安全性評価では，サイドチャネル攻撃耐性 (第 1.1.2 項参照) を評価する．

暗号の実装にはハードウェア実装とソフトウェア実装があり，用途に応じて選択されるため，方針や実装要件が大きく異なる．そのため，サイドチャネル攻撃は対象となる実装等に応じて有効な手法が異なる．暗号を実装したデバイス (暗号デバイス) に有効な攻撃を考慮し，安全性を評価することが重要である．

1.1.1 選択平文攻撃

ブロック暗号は段関数とよばれる関数を繰り返し使用して構成される．段関数の繰り返し回数を段数という．ブロック暗号では，ショートカット法によってアルゴリズムの安全性を評価する．ショートカット法とは，暗号の脆弱性を利用して鍵の全数探索より少ない計算量で鍵を推定する手法のことをいう．ショートカット法には以下の攻撃モデルが存在する [56].

暗号文単独攻撃 暗号文のみを手に入れて行う攻撃.

既知平文攻撃 与えられた平文とそれに対応する暗号文を手に入れて行う攻撃.

選択平文攻撃 攻撃者が任意に選択した平文とそれに対応する暗号文を手に入れて行う攻撃.

選択暗号文攻撃 攻撃者が任意に選択した暗号文とそれに対応する平文を手に入れて行う攻撃.

なお，鍵は攻撃を実行している間に変更されず，未知の定数として扱う．攻撃対象で使用されている鍵を推定し (鍵の推定)，これが正しい場合に攻撃が成功するという．また，アルゴリズムは既知とする．これにより，鍵が未知であれば安全性が保証される暗号を構成することができる．

アルゴリズムの安全性の基準として最も一般的なのは選択平文攻撃に対する耐性である．ブロック暗号は選択平文攻撃によって鍵の推定が困難であるように構成される．その前提において，鍵の推定が可能である段数 (攻撃可能段数) を評価する．選択平文攻撃による攻撃可能段数の最大値が重要な安全性の指標となる．ブロック暗号に対する具体的な攻撃法として，差分攻撃，不能差分攻撃及び

Integral 攻撃等があるが [6][7][25]，その中で最も攻撃可能段数が大きいものが最大値となる．

本論文では選択平文攻撃のひとつである Integral 攻撃に注目する．Integral 攻撃は Knudsen らが提案した攻撃手法で，Integral 特性を用いる [25]．準備段階で鍵に依らず成立する Integral 特性を探索する．攻撃段階では Integral 特性を得られる選択平文を攻撃対象に入力し，得られた暗号文を利用して鍵を推定する．Integral 特性探索と鍵の推定という 2 つの段階があるが，本論文では安全性評価の上でより重要な指標である Integral 特性探索を対象とする．

1.1.2 サイドチャネル攻撃

第 1.1.1 項で示したショートカット法の仮定では，攻撃者はブロック暗号の入出力の情報は得られるが，その他の情報は得られない (ブラックボックス仮定)．しかし，Kocher が提案した差分電力解析攻撃 [27] の登場により，従来のブラックボックス仮定による評価のみでは安全性の評価は十分に行えないことが明らかになった．Kocher は暗号デバイスの途中処理の消費電力を測定することで，鍵が推定可能なことを示した．このような暗号デバイス動作時に攻撃者が得られる情報 (サイドチャネル情報) は多岐にわたり，電磁波 [19] や CPU のキャッシュメモリ [47] 等が存在する．サイドチャネル情報から鍵を推定する攻撃をサイドチャネル攻撃という．

サイドチャネル攻撃耐性評価では物理的な実験が一般的である．対象である暗号デバイスに対して既知である様々な攻撃を適用し，これらを防ぐのに必要な対策を行なう．しかし，この手法には未知のサイドチャネル攻撃に対する対策が行えないという欠点がある．

未知攻撃への対策として形式的評価があり [15][22][38][40][49]，本論文はこれを対象とする．形式的評価ではサイドチャネル情報をモデル化 (漏洩モデル) し，攻撃が実行困難となる条件 (測定回数等) を理論的に導出する．なお，形式的評価は特定の暗号デバイスを対象としないため，一般的な評価が可能である．本論文

では Bogdanov らが提案したランダム漏洩モデル及び Dinur らが提案した 1 ビットアクセス漏洩モデルに注目し，これらの漏洩モデルにおいて有効な差分バイアス攻撃及びサイドチャネル Cube 攻撃を研究する [9][15].

1.2 研究の目的

本論文の目的は Integral 攻撃，差分バイアス攻撃及びサイドチャネル Cube 攻撃に対する耐性評価手法を提案することである．耐性評価手法を適用することで，安全性に関するパラメータを得る．設計段階においては，このパラメータを反映することで安全なアルゴリズム及びその実装を設計することができる．また，すでに発表されている暗号に対しては，このパラメータによって安全性を確認できる．

選択平文攻撃においてはパラメータとして攻撃可能段数の最大値を導出する．これを上回る段数を設定することで，既知の攻撃に対する安全性を保証できる．また，余分に段関数を加えることで未知の攻撃に対する対策とする．本論文では攻撃者にとって優位な Integral 特性を高速に探索する手法を提案する．得られた特性から Integral 攻撃における攻撃可能段数が得られる．

サイドチャネル攻撃に対する形式的評価では，対象のデバイスの実装環境に応じて現実的な漏洩モデルを全て選択する．選択した漏洩モデルにおいて攻撃が実行困難となる条件をパラメータとして導出する．その後，評価対象のデバイスにおいてこのパラメータを満たすように対策を行なう．これにより，漏洩モデルで想定される未知のサイドチャネル攻撃に対して対策を行うことができる．本論文では差分バイアス攻撃及びサイドチャネル Cube 攻撃において既存手法よりも強力な攻撃手法を提案し，攻撃が困難となるパラメータを厳密に導出する耐性評価手法を提案する．

第 2 章

Integral 攻撃に対する耐性評価手法

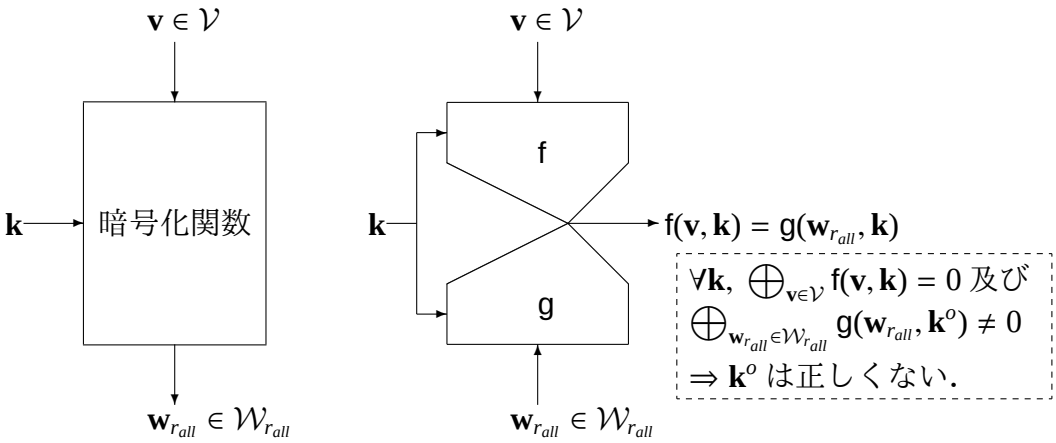
本論文ではアルゴリズムの安全性評価として，選択平文攻撃のひとつである Integral 攻撃に注目する．なお，選択平文攻撃の仮定を付録 H に示す．Integral 攻撃は Knudsen らが提案した攻撃手法で，以下の Integral 特性を用いる [25].

定義 2.1 (Integral 特性)**.** 暗号化処理の中間値をブール関数 f で $\mathsf{f}(\mathbf{v}, \mathbf{k})$ と表現する．“任意の鍵 $\mathbf{k}$ に対し $\bigoplus_{\mathbf{v}\in\mathcal{V}} \mathsf{f}(\mathbf{v}, \mathbf{k}) = 0$”または“任意の鍵 $\mathbf{k}$ に対し $\bigoplus_{\mathbf{v}\in\mathcal{V}} \mathsf{f}(\mathbf{v}, \mathbf{k}) = 1$”ならば，選択平文集合 $\mathcal{V}$ に対し中間値 $\mathsf{f}(\mathbf{v}, \mathbf{k})$ において Integral 特性が成立するという．

ここで，ブール関数 f は平文及び鍵から中間値を求めるものであり，暗号化処理の一部に相当する．また，ブール関数の積分を以下に定義する．

定義 2.2 (ブール関数の積分)**.** ブール関数 $\mathsf{f} : \mathbb{F}_2^n \to \mathbb{F}_2$ の定義域の部分集合 $\mathcal{X} \subset \mathbb{F}_2^n$ の全要素に対して $\bigoplus_{\mathbf{x}\in\mathcal{X}} \mathsf{f}(\mathbf{x})$ を計算することをブール関数の積分とよぶ.

定義 2.1 はブール関数の積分値が鍵の値に依らず定数となることを意味する．Integral 特性は付録 J 及び K に示す全単射及びブール関数の特性から探索できる．Integral 特性は鍵に依らないため，Integral 特性探索はあらかじめ暗号化関数を解析して行う．Integral 特性探索の後，図 2.1 に示す鍵の推定を行う．図 2.1 (a) では選択平文集合 $\mathcal{V}$ に含まれる平文の暗号化処理を行い，暗号文多重集合 $\mathcal{W}_{r_{all}}$ を得ている．Integral 特性探索により，中間値 $\mathsf{f}(\mathbf{v}, \mathbf{k})$ における $\bigoplus_{\mathbf{v}\in\mathcal{V}} \mathsf{f}(\mathbf{v}, \mathbf{k}) = 0$ という Integral 特性が既知であるとする．図 2.1 (b) のように，この中間値は暗号文 $\mathbf{w}_{r_{all}}$ から，$\mathsf{f}(\mathbf{v}, \mathbf{k}) = \mathsf{g}(\mathbf{w}_{r_{all}}, \mathbf{k})$ で求まるとする．なお，f が平文及び鍵から中間値を求めるのに対し，g は暗号文及び鍵から同様の値を求めるものであり，復号処理の一部に相当する．鍵 $\mathbf{k}$ の o 番目の候補を $\mathbf{k}^o$ とし，暗号文を用いて $\mathsf{g}(\mathbf{w}_{r_{all}}, \mathbf{k}^o)$ の積分を行う．もし $\bigoplus_{\mathbf{w}_{r_{all}}\in\mathcal{W}_{r_{all}}} \mathsf{g}(\mathbf{w}_{r_{all}}, \mathbf{k}^o) \neq 0$ ならば，既知の Integral 特性に矛盾する



(a) 暗号文多重集合の導出　　(b) 鍵の推定

図 2.1　Integral 攻撃における鍵の推定の概要

ため $\mathbf{k}^o$ は候補から除外できる．最終的に残った候補数が十分に小さければ，攻撃者は総当りで正しい鍵を推定できる．

鍵の推定については手順が確立されているため ([17] 及び [45] 参照)，Integral 特性によって攻撃可能段数は決定される．そのため，本論文では Integral 特性探索を対象とし，全単射の特性を利用した Integral 特性探索手法 IPS-BP (Integral Property Search with Bijective Property) を提案する．IPS-BP は Knudsen らの探索手法 [25] IPS-PM (Integral Property Search - Previous Method) の改良となる．第 2.1 節で Integral 特性とその探索法について述べた後，第 2.2 節で IPS-PM を第 2.3 節で IPS-BP について示す．

2.1　Integral 特性

まず，ブール関数の特性と Integral 特性の関係を示す．その際，変数及び定数ビットを指定する方法 (付録 H 参照) で選択平文集合を定義し，変数ビットの添字集合が $\mathcal{I}$ である選択平文集合を $\mathcal{V}_{\mathcal{I}}$ とする ($|\mathcal{I}| = n_{var}$)．ここで，本論文では選択平文集合を入力して暗号化処理を行う際，平文の値によって変化する値を“変数”，一定である値を“定数”とよぶ (鍵は定数)．平文を $\mathbf{v}$ を変数 $\mathbf{v}_{var}$ 及び定数 $\mathbf{v}_{const}$ に分割すると，$\mathcal{V}_{\mathcal{I}} = \{\mathbf{v} | \mathbf{v}_{var} \in \mathbb{F}_2^{n_{var}}, \mathbf{v}_{const} = const\}$ となる．ブール関数の特性

から，関数 f で求められる中間値の積分値は

$$\bigoplus_{\mathbf{v}\in\mathcal{V}_{\mathcal{I}}} \mathsf{f}(\mathbf{v},\mathbf{k}) = \begin{cases} 0 & a_{\mathcal{I}} = 0 \text{ の場合} \\ 1 & a_{\mathcal{I}} = 1 \text{ の場合} \end{cases} \tag{2.1}$$

のように，$\mathsf{f}_{\mathcal{I}}$ の最大次数項の係数 $a_{\mathcal{I}}$ (付録 K 参照) に従う．係数を関数 $a_{\mathcal{I}} = \mathsf{a}_{\mathcal{I}}(\mathbf{v}_{const},\mathbf{k})$ とみなすとき，以下の 3 つの場合が存在する．

case 1： 任意の $\mathbf{v}_{const}$ 及び $\mathbf{k}$ に対し， $\mathsf{a}_{\mathcal{I}}(\mathbf{v}_{const},\mathbf{k}) = 0$

case 2： 任意の $\mathbf{v}_{const}$ 及び $\mathbf{k}$ に対し， $\mathsf{a}_{\mathcal{I}}(\mathbf{v}_{const},\mathbf{k}) = 1$

case 3： その他 (2.2)

case 1 及び **case 2** においては，積分値が定数になるため Integral 特性が成立する．Integral 特性が成立する中間値を平衡ビットとよぶ．**case 3** では積分値が定数にならないため Integral 特性が成立しない．このような中間値を非平衡ビットとよぶ．

次に，全単射の特性と Integral 特性の関係を示す．鍵 $\mathbf{k}$ が定数の場合に全単射 $\mathsf{F}:\mathbb{F}_2^{n_{var}} \to \mathbb{F}_2^{n_{var}}$ が平文 n_{var} ビットから中間値 n_{var} ビットを出力すると仮定する．全単射の特性が F において成立する場合，$\bigoplus_{\mathbf{v}_{var}\in\mathbb{F}_2^{n_{var}}} \mathsf{F}(\mathbf{v}_{var}) = 0$ が成立する．したがって，F の出力は全て **case 1** であり平衡ビットとなる．つまり，全単射の特性を利用することでブール関数の特性を考慮することなく平衡ビットの判定が可能となる．IPS-PM 及び IPS-BP ではこれを利用する．

変数ビットの数が多い程より多くの中間値が平衡ビットとなる．これを変数ビットの添字集合が $\mathcal{I}'$ 及び $\mathcal{I}$ $(\mathcal{I}' \supset \mathcal{I})$ である選択平文集合 $\mathcal{V}_{\mathcal{I}'}$ 及び $\mathcal{V}_{\mathcal{I}}$ を用いて示す．もし，$\mathcal{V}_{\mathcal{I}}$ に対し **case 1** または **case 2** が成立するならば，

$$\bigoplus_{\mathbf{v}\in\mathcal{V}_{\mathcal{I}'}} \mathsf{f}(\mathbf{v},\mathbf{k}) = \bigoplus_{\mathcal{V}_{\mathcal{I}}|\mathcal{I}\subset\mathcal{I}'} \bigoplus_{\mathbf{v}\in\mathcal{V}_{\mathcal{I}}} \mathsf{f}(\mathbf{v},\mathbf{k}) = 0 \tag{2.3}$$

が成り立つ．なお，$\{\mathcal{V}_{\mathcal{I}}|\mathcal{I}\subset\mathcal{I}'\}$ の要素 $\mathcal{V}_{\mathcal{I}}$ は，$\mathcal{I}'\backslash\mathcal{I}$ なる添字集合の値がそれぞれ異

なる (要素数は $2^{|\mathcal{I}'|-|\mathcal{I}|}$). 仮定より $\bigoplus_{\mathbf{v}\in\mathcal{V}_\mathcal{I}} \mathbf{f}(\mathbf{v},\mathbf{k})$ が定数であり，かつ，$\{\mathcal{V}_\mathcal{I}|\mathcal{I}\subset\mathcal{I}'\}$ の要素数は偶数なので，積分値は確定的に 0 となる．

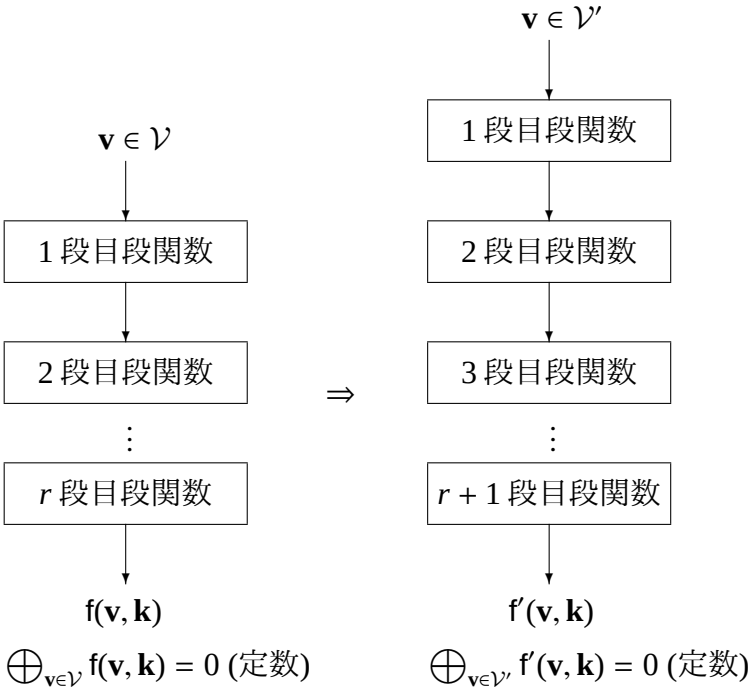
Integral 特性探索では選択平文集合 $\mathcal{V}$ を変化させつつ，平衡ビットを探索する．実際に式 (2.1) を十分な数の鍵候補に対し検証すれば探索が可能である．しかし，変数ビット数 n_{var} に対して $2^{n_{var}}$ 回の暗号化処理を繰り返す必要があるため，n_{var} が大きい場合は計算機による処理が困難となる．

これに対し，Integral 特性を形式的に見積もる手法が存在し，全単射とブール関数の特性のいずれかもしくは両方を利用する．IPS-PM は全単射の特性のみを利用し [25]，全単射とブール関数の特性を考慮した Division 特性を用いる手法を Todo が提案している [44][46]．Division 特性にはビット単位とワード単位の特性が存在するが，本論文ではまとめて IPS-DP (Integral Property Search with Division Property) とよぶ．

2.2 Integral 特性探索手法の先行研究 [25]

IPS-PM は暗号化関数における演算の単位であるワードを単位として実行される．そのため，PRESENT(付録 E，図 E.1，[8]) のように，ビット単位の処理が必要となるブロック暗号に対してはこの手法は適用できない．選択平文集合はワード単位で変数を選択する (付録 H 参照)．IPS-PM では積分値が任意の鍵 $\mathbf{k}$ に対し 0 となるワードを探索する．なお，積分値が 0 となるワードを平衡ワードとよぶ (平衡ワードに含まれるビットは全て平衡ビット).

図 2.2 に IPS-PM の概要を示す．図 2.2 (a) では，変数ワードが 1 個の選択平文集合 $\mathcal{V}$ を仮定して複数段処理における Integral 特性を探索する (第 2.2.1 項に示す)．図 2.2 (b) では，$\mathcal{V}$ から得られた Integral 特性が 1 段分多く段関数処理をしても成立する選択平文集合 $\mathcal{V}'$ を求める．その際，選択平文集合の変数ワードの数は増加し，データ量が増加する ($|\mathcal{V}| < |\mathcal{V}'|$)．この処理を Integral 特性の入力側への拡張とよび，第 2.2.2 項に示す．



(a) 1 変数ワードの選択平文集合　(b) Integral 特性の入力側への拡張による Integral 特性探索

図 2.2　Integral 特性探索の先行手法 (IPS-PM) の概要

2.2.1　1 変数ワードの選択平文集合による Integral 特性探索

平衡ワードは中間値の多重集合で決定される．中間値の多重集合を $\mathcal{W}_{r,i} = \{\mathbf{w}_{r,i}^{o} | \mathbf{w}_{r,i}^{o} = \mathsf{F}(\mathbf{v}^{o}, \mathbf{k}) \in \mathbb{F}_2^{n_{word}},\ \mathbf{v}^{o} \in \mathcal{V}_{\mathcal{J}}\}$ とする．ここで，$\mathcal{V}_{\mathcal{J}}$ は選択平文集合であり，$\mathbf{v}^{o}$ は $\mathcal{V}_{\mathcal{J}}$ に含まれる o 番目の平文を意味する．変数ワードが 1 個のとき ($|\mathcal{J}| = 1$)，中間値の状態を以下に分類することで平衡ワードの探索を容易にする．

Constant (C): $\mathcal{W}_{r,i}$ に含まれる任意の $\mathbf{w}_{r,i}^{o}$ 及び $\mathbf{w}_{r,i}^{o'}$ が同じ値である．

All (A): $\mathcal{W}_{r,i}$ に含まれる任意の $\mathbf{w}_{r,i}^{o}$ 及び $\mathbf{w}_{r,i}^{o'}$ が異なる．

Balanced (B): C 及び A 状態でなく，かつ，積分値が 0 である ($\bigoplus_{\mathbf{w}_{r,i}^{o} \in \mathcal{W}_{r,i}} \mathbf{w}_{r,i}^{o} = 0$)．

Unknown (U): その他

中間値 $\mathbf{w}_{r,i}$ の状態を $\mathtt{W}_{r,i} = \mathtt{A}$ のように記号 $\mathtt{W}_{r,i}$ (平文は $\mathtt{V}_i$) を用いて表現する．中間値の状態の判定法を以下に示す．

C 判定: 変数ワードによって値が変化しない中間値は C 状態と判定できる．

A 判定: 鍵 $\mathbf{k}$ 及び平文の変数ワード $\mathbf{v}_{i'}$ ($\mathcal{J} = \{i'\}$) 以外を定数とした $\mathbf{w}_{r,i} = \mathsf{R}(\mathbf{v}_{i'})$ なる R が全単射ならば，全単射の特性より $\mathbf{w}_{r,i}$ は A 状態と判定できる．

B 判定: 鍵 $\mathbf{k}$ 及び平文の変数ワード $\mathbf{v}_{i'}$ ($\mathcal{J} = \{i'\}$) 以外を定数としたとき，中間値 $\mathbf{w}_{r,i}$ が $\mathbf{w}_{r,i} = \bigoplus_j \mathsf{R}_j(\mathbf{v}_{i'})$ のように，全単射 R_j の出力 (A 状態) の XOR で表現できるとする．A 状態の中間値の積分値は 0 なので，$\mathbf{w}_{r,i}$ の積分値は

$$\bigoplus_{\mathbf{v}\in\mathcal{V}_{\mathcal{J}}} \bigoplus_j \mathsf{R}_j(\mathbf{v}_{i'}) = \bigoplus_j \bigoplus_{\mathbf{v}\in\mathcal{V}_{\mathcal{J}}} \mathsf{R}_j(\mathbf{v}_{i'}) = \bigoplus_j 0 = 0 \tag{2.4}$$

となる．積分値が確定的に 0 となるため，$\mathbf{w}_{r,i}$ は B 状態と判定できる．

U 判定: その他の場合 U 状態と判定する．

まず，平文の変数ワードは A 状態，定数ワードは C 状態とし，全中間値ワードの状態は初期状態として U 状態とする．その後，表 2.1 に示す状態の判定表を用いて入力側から出力側に各中間値の状態を更新する．表 2.1 において，“$\bigoplus_{\mathbf{w}^o_{r,i}\in\mathcal{W}_{r,i}} \mathbf{w}^o_{r,i}$” は積分を意味し，0 であれば平衡ワードである．“$\mathsf{R}(\mathbf{w}_{r,i})$” は任意の全単射 R に入力されたときの出力の状態を示す．ここで，A 状態は全単射の特性から状態が変化しないため，B 状態よりも攻撃者にとって有利な状態となる．“$\oplus\mathbf{w}_{r',i'}$” は他の中間値 $\mathbf{w}_{r',i'}$ との XOR による変化を示している．

実行例: 表 2.1 を用いた Integral 特性探索例を図 2.3 に示す Type-II Feistel 構造 [55] の段関数を有するブロック暗号を用いて例示する．この段関数は以下の Feistel 構造の関数 FS を 2 個並列処理している．

$$(\mathbf{y}_1, \mathbf{y}_2) = \mathsf{FS}(\mathbf{x}_1, \mathbf{x}_2) = (\mathsf{R}(\mathbf{x}_1) \oplus \mathbf{x}_2, \mathbf{x}_1) \tag{2.5}$$

なお，R は $\mathbb{F}_2^{n_{ward}} \to \mathbb{F}_2^{n_{ward}}$ なる任意の全単射である．

表 2.1　1 変数ワードの選択平文集合による Integral 特性探索における中間値の状態の判定表

		$\bigoplus_{\mathbf{w}^o_{r,i}\in\mathcal{W}_{r,i}} \mathbf{w}^o_{r,i}$	$\mathsf{R}(\mathbf{w}_{r,i})$	$\oplus\mathbf{w}_{r',i'}$			
				C	A	B	U
$\mathbf{w}_{r,i}$	C	0	C	C	A	B	U
	A	0	A	A	B	B	U
	B	0	U	B	B	B	U
	U	未知	U	U	U	U	U

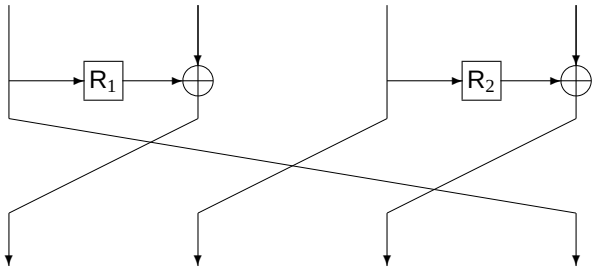


図 2.3　Type-II Feistel 構造の段関数

図 2.4 に図 2.3 の段関数を複数回処理した際の Integral 特性の探索例を示す．平文入力において，2 番目のワードのみを変数ワードとし，その他は定数ワードとする．表 2.1 を用いて各中間値の状態を入力側から出力側に逐次的に判定する．7 段目出力では全ワードが U 状態と判定されるので，Integral 特性は 6 段目出力まで存在する．ここで，$\mathcal{W}_r^{\mathcal{J}'}$ は r 段目出力の中間値多重集合であり，添字集合 $\mathcal{J}'$ で示すワードが平衡ワードとなることを示す．選択平文集合 $\mathcal{V}_{\mathcal{J}}$ に対する積分値を計算した際，r 段目出力において $\mathcal{J}'$ で示すワードが平衡ワードとなるという状況を関数 $\mathsf{IP}_r : \mathbb{F}_2^{n_{all}\cdot 2^{n_{var}}} \to \mathbb{F}_2^{n_{all}\cdot 2^{n_{var}}} (\mathcal{V}_{\mathcal{J}} \mapsto \mathcal{W}_r^{\mathcal{J}'})$ で表現する．r は Integral 特性が得られる最大段数を示し，IP_r は選択平文集合を r 段文暗号化して中間値多重集

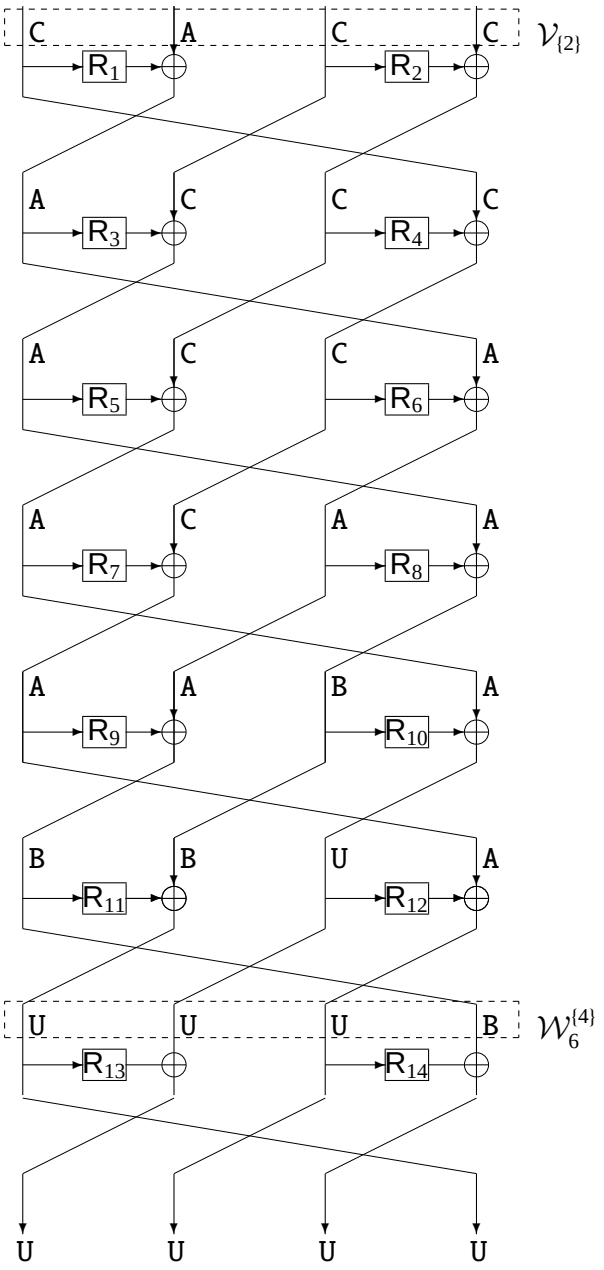


図 2.4　表 2.1 を用いた Integral 特性の探索例

合を出力する．図 2.4 の例では以下が示されている．

$$\mathcal{W}_6^{\{4\}} = \mathsf{IP}_6(\mathcal{V}_{\{2\}}) \tag{2.6}$$

2.2.2 Integral 特性の入力側への拡張

1 変数ワードの選択平文集合による Integral 特性探索の結果を拡張し，変数ワード数は増えるが 1 段分多く段関数処理をしても成立する Integral 特性を得る．なお，変数ワードの数が多いほど，より多くの平衡ワードが得られる (第 2.1 節参照)．Integral 特性の拡張が可能な理由を示す．1 段目出力において，1 ワードのみが変数でその他が定数である中間値多重集合 $\mathcal{W}_{\mathcal{J}_1}$ が得られると仮定する．なお，$\mathcal{J}_r = \{(r, i) | i \in \mathcal{J}\}$ とする (r:段数)．ここで，$\mathcal{J}_1 \subset \mathcal{J}_1'$ なる中間値多重集合 $\mathcal{W}_{\mathcal{J}_1'}$ を仮定する．$\mathcal{W}_{\mathcal{J}_1}$ に対し平衡ワードとなる中間値 $\mathbf{w}_{r,i} = \mathsf{F}(\mathbf{w}_1, \mathbf{k})$ において，

$$\bigoplus_{\mathbf{w}_1 \in \mathcal{W}_{\mathcal{J}_1'}} \mathsf{F}(\mathbf{w}_1, \mathbf{k}) = \bigoplus_{\mathcal{W}_{\mathcal{J}_1} | \mathcal{J}_1 \subset \mathcal{J}_1'} \bigoplus_{\mathbf{w}_1 \in \mathcal{W}_{\mathcal{J}_1}} \mathsf{F}(\mathbf{w}_1, \mathbf{k}) = 0 \tag{2.7}$$

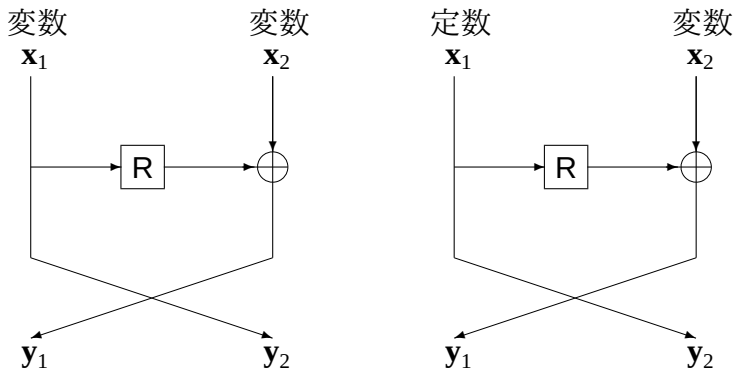
が式 (2.3) と同様の理由により成り立つ．

次に，$\mathbf{w}_{r,i} = \mathsf{F}'(\mathbf{v}, \mathbf{k}) = \mathsf{F}(\mathbf{w}_1, \mathbf{k})$ なる関数 F' を考える．ここで，$\mathbf{w}_1 = \mathsf{G}(\mathbf{v}, \mathbf{k})$ なる G は段関数であり，$|\mathcal{J}'| = |\mathcal{J}_1'|$ なる選択平文集合 $\mathcal{V}_{\mathcal{J}'}$ を入力する．鍵を定数としたとき，G が全単射 $\mathbf{v}_{\mathcal{J}'} \mapsto \mathbf{w}_{\mathcal{J}_1'}$ と定数入出力の関数を並列に処理する場合，G で処理した 1 段目出力の多重集合は $\mathcal{W}_{\mathcal{J}_1'}$ となる．したがって，

$$\bigoplus_{\mathbf{v} \in \mathcal{V}_{\mathcal{J}'}} \mathsf{F}'(\mathbf{v}, \mathbf{k}) = \bigoplus_{\mathbf{v} \in \mathcal{V}_{\mathcal{J}'}} \mathsf{F}(\mathsf{G}(\mathbf{v}, \mathbf{k}), \mathbf{k}) = \bigoplus_{\mathbf{w}_1 \in \mathcal{W}_{\mathcal{J}_1'}} \mathsf{F}(\mathbf{w}_1, \mathbf{k}) \tag{2.8}$$

が成立する．$\mathcal{V}_{\mathcal{J}}$ で得られる Integral 特性は全て $\mathcal{W}_{\mathcal{J}_1'}$ からも得られるので，これに 1 段分追加した特性が $\mathcal{V}_{\mathcal{J}'}$ から得られる．

実行例: 第 2.2.1 項の実行例の続きとして，式 (2.6) で示した結果を用いて Integral 特性の拡張について例示する．付録 J に示す理由から，FS では図 2.5 に示す全単射



(a) $\mathbb{F}_2^{2\cdot n_{ward}} \to \mathbb{F}_2^{2\cdot n_{ward}}$ なる全単射　(b) $\mathbb{F}_2^{n_{ward}} \to \mathbb{F}_2^{n_{ward}}$ なる全単射

図 2.5　Feistel 構造を有する関数 FS から全単射を定義する方法

が定義できる．図 2.5 (a) では全入力が変数であり，関数 FS 全体で $\mathbb{F}_2^{2\cdot n_{ward}} \to \mathbb{F}_2^{2\cdot n_{ward}}$ なる全単射となる．図 2.5 (b) では $\mathbf{x}_2$ のみ変数とし，$\mathbf{x}_2 \mapsto \mathbf{y}_1$ が $\mathbb{F}_2^{n_{ward}} \to \mathbb{F}_2^{n_{ward}}$ なる全単射となることを示している．前者を考慮すると，$\mathbf{w}_{1,1}$ が出力に含まれる全単射 $(\mathbf{v}_3, \mathbf{v}_4) \mapsto (\mathbf{w}_{1,1}, \mathbf{w}_{1,2})$ が段関数内に存在する．$\mathbf{w}_{1,1}$ 及び $\mathbf{w}_{1,2}$ が変数ワードである $\mathcal{W}_{\{(1,1),(1,2)\}}$ では式 (2.6) が得られるので，

$$\mathcal{W}_7^{\{4\}} = \mathsf{IP}_7(\mathcal{V}_{\{3,4\}}) \tag{2.9}$$

が得られる．さらに，同様の処理を繰り返すことでより多くの段関数処理の後に成立する Integral 特性を得ることができる．その際，データ量の制限 (付録 H 参照) から，全平文ワードを変数に指定することはできない．したがって，全平文ワードが変数となったならば探索を終了し，最後の拡張以前の結果を出力する．

2.3　全単射の特性を利用した Integral 特性探索手法の提案

第 2.2 節で示した IPS-PM[25] を改良し，より多くの平衡ワードを探索する手法 IPS-BP を提案する．Algorithm 2.1 に IPS-BP を示す．全単射の特性をより多くの中間値において適用することで IPS-PM よりも多くの平衡ワードを探索できる．IPS-PM では複数変数ワードの選択平文集合は 1 変数ワードの選択平文集合から拡張される．そのため，拡張によって定義されない選択平文集合から得られ

Algorithm 2.1 全単射の特性を利用した Integral 特性探索 (IPS-BP)

input 変数ワードの添字集合 $\mathcal{J}$.
全ワードの状態集合 $\check{\mathcal{S}}_{\mathcal{J}}$ を初期化 $\forall \check{\mathtt{W}}_{r,j} \in \check{\mathcal{S}}_{\mathcal{J}}$, $\check{\mathtt{W}}_{r,j} \leftarrow \check{\mathtt{U}}$.
$\check{\mathtt{V}}_j \leftarrow \check{\mathtt{A}}$ $(j \in \mathcal{J})$, $\check{\mathtt{V}}_j \leftarrow \check{\mathtt{C}}$ $(j \notin \mathcal{J})$
$(\check{\mathcal{S}}_{\mathcal{J}}, \mathcal{J}_{bij}) \leftarrow \mathsf{BijFuncSearch}(\check{\mathcal{S}}_{\mathcal{J}}, \mathcal{J})$ ▷ Algorithm 2.2 参照
$\check{\mathcal{S}}_{\mathcal{J}} \leftarrow \mathsf{InitSearch}(\check{\mathcal{S}}_{\mathcal{J}}, \mathcal{J}_{bij})$ ▷ Algorithm 2.3 参照
$\check{\mathcal{S}}_{\mathcal{J}} \leftarrow \mathsf{AddSearch}(\check{\mathcal{S}}_{\mathcal{J}}, \mathcal{J}_{bij})$ ▷ Algorithm 2.4 参照
return $\check{\mathcal{S}}_{\mathcal{J}}$

る Integral 特性が未知となる．これに対し，IPS-BP は任意の選択平文集合に対して Integral 特性探索が行える．なお，IPS-BP は IPS-PM と同様にワード単位で実行されるため，ビット単位の処理が必要なブロック暗号には適用できない．

第 2.3.1 項で多変数ワードの選択平文集合における中間値の多重集合の状態の定義を行う．第 2.3.2 項から第 2.3.4 項で IPS-BP の各関数を示す．

2.3.1 多変数ワードの選択平文集合による Integral 特性探索

IPS-BP では第 2.2 節で示したものとは異なる以下の 5 つの状態を用いる．

Constant ($\check{\mathtt{C}}$): $\mathcal{W}_{r,i}$ に含まれる任意の $\mathbf{w}^o_{r,i}$ 及び $\mathbf{w}^{o'}_{r,i}$ が同じ値である．

All equal ($\check{\mathtt{A}}$): $\mathbb{F}_2^{n_{word}}$ の任意の要素が同じ個数ずつ $\mathcal{W}_{r,i}$ に含まれる．

Even ($\check{\mathtt{E}}$): $\check{\mathtt{C}}$ 及び $\check{\mathtt{A}}$ 状態でなく，かつ，$\mathbb{F}_2^{n_{word}}$ の任意の要素が偶数個ずつ $\mathcal{W}_{r,i}$ に含まれる．

Balanced ($\check{\mathtt{B}}$): $\check{\mathtt{C}}$，$\check{\mathtt{A}}$ 及び $\check{\mathtt{E}}$ 状態でなく，かつ，積分値が 0 である ($\bigoplus_{\mathbf{w}^o_{r,i} \in \mathcal{W}_{r,i}} \mathbf{w}^o_{r,i} = 0$).

Unknown ($\check{\mathtt{U}}$): その他

なお，多変数ワードの選択平文集合を入力した際の平文及び中間値の状態を $\check{\mathtt{W}}_{r,i}$ (平文は $\check{\mathtt{V}}_i$) で表す．$\check{\mathcal{S}}_{\mathcal{J}}$ ($\check{\mathtt{V}}_i$,$\check{\mathtt{W}}_{r,i} \in \check{\mathcal{S}}_{\mathcal{J}}$) を状態集合とし，Algorithm 2.1 で示した IPS-BP においてはこれを各関数で更新する．

表 2.2 に中間値が処理された場合の多重集合の状態の判定表を示す．$\check{\mathtt{E}}$ の状態を除くと，表 2.1 とほぼ同様の結果が得られることが分かる．$\check{\mathtt{E}}$ の状態の多重集

表 2.2　多変数ワードの選択平文集合による Integral 特性探索における中間値の状態の判定表

		$\bigoplus_{\mathbf{w}^o_{r,j}\in\mathcal{W}_{r,j}} \mathbf{w}^o_{r,j}$	$\mathsf{R}(\mathbf{w}_{r,j})$	$\oplus\mathbf{w}_{r',j'}$				
				Č	Ă	Ĕ	Ă	Ŭ
$\mathbf{w}_{r,j}$	Č	0	Č	Č	Ă	Ĕ	Ă	Ŭ
	Ă	0	Ă	Ă	Ă	Ă	Ă	Ŭ
	Ĕ	0	Ĕ	Ĕ	Ă	Ă	Ă	Ŭ
	Ă	0	Ŭ	Ă	Ă	Ă	Ă	Ŭ
	Ŭ	未知	Ŭ	Ŭ	Ŭ	Ŭ	Ŭ	Ŭ

合を全単射 R に入力した際の出力の多重集合において，各値の含まれる個数は入力と同様に偶数となる (状態が変化しない)．これにより，Ĕ の状態は Ă と同様に Ă よりも攻撃者にとって有利な状態となる．

IPS-BP では各関数で暫定的に判定した状態を更新する．つまり，暫定的に判定した状態と異なる状態が判定される可能性がある．その際，表 2.3 を用いて暫定的な状態を更新する．表 2.3 では 2 つの状態からより特殊な状態を選択している．例えば，暫定的に Ă 状態であるが，これと矛盾しない Ă 状態とも判断できる場合は Ă 状態へ変更する．このように状態を変更することを“新たな判定”とよぶ．なお，Ă 状態であれば積分値は 0 であるので，Ă は Ă の特殊な状態であるといえる．

中間値の状態の判定法を示す準備として，以下を定義する．

定義 2.3 (中間値の多重集合 $\mathcal{W}_{\mathcal{J}_{mid}}$ において Ă 状態)**.** 複数の平文または中間値のワードを $\mathbf{w}_{\mathcal{J}_{mid}}$ とする．なお，$(r,i)\in\mathcal{J}_{mid}$ は段数 r とワードの添字 i を要素とする集合とし，$\mathbf{w}_{\mathcal{J}_{mid}}$ は $\mathcal{J}_{mid}$ に含まれる中間値ワードの全ビットを成分とする．選択平文集合を入力した際に得られる $\mathbf{w}_{\mathcal{J}_{mid}}$ の値の多重集合を $\mathcal{W}_{\mathcal{J}_{mid}}$ とする ($|\mathcal{W}_{\mathcal{J}_{mid}}| = 2^{n_{var}}$)**.**

表 2.3　暫定的な中間値の状態の新たな判定を踏まえた更新の方法

暫定的な状態	新たな判定			
	Ǎ	Ě	B̌	Ǔ
Ǎ	Ǎ	Ǎ	Ǎ	Ǎ
Ě	Ǎ	Ě	Ě	Ě
B̌	Ǎ	Ě	B̌	B̌
Ǔ	Ǎ	Ě	B̌	Ǔ

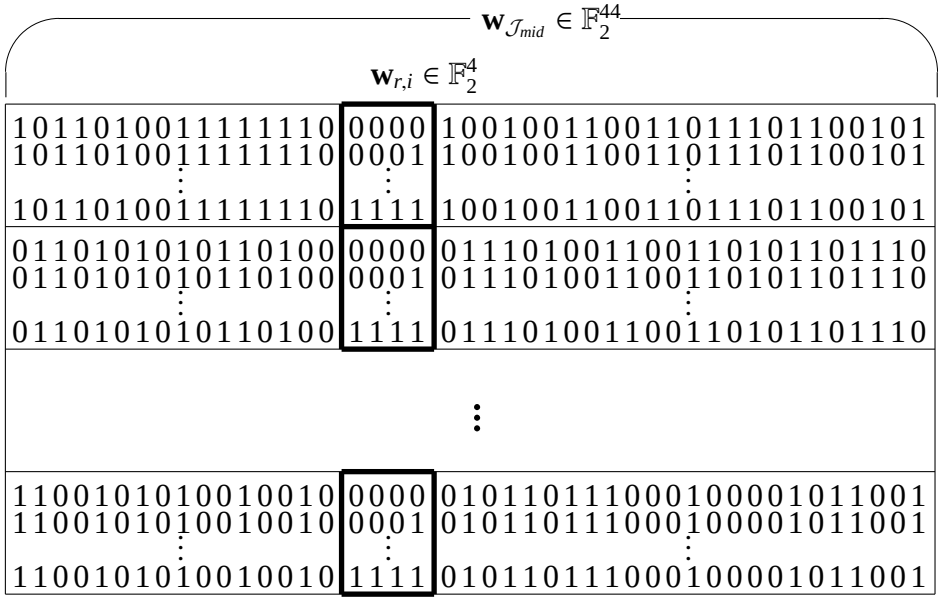
$\mathbf{w}_{\mathcal{J}_{mid}}$ を $\mathbf{w}_{r,i}$ 及び $\mathbf{w}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}$ に分割して考える．$\mathcal{W}_{\mathcal{J}_{mid}}$ において，$\mathcal{W}_{r,i}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}]$ をある値 $\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}$ によって定まる中間値 1 ワードの多重集合

$$\mathcal{W}_{r,i}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}] = \{\mathbf{w}^{o}_{r,i} | (\mathbf{w}^{o}_{r,i}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}) \in \mathcal{W}_{\mathcal{J}_{mid}}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}} = const\} \tag{2.10}$$

とする．任意の $\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}$ に対して $\mathcal{W}_{r,i}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}]$ に $\mathbb{F}_2^{n_{word}}$ の任意の要素が同じ個数ずつ含まれるとき，$\mathbf{w}_{r,i}$ は $\mathcal{W}_{\mathcal{J}_{mid}}$ において Ǎ 状態であると表現する．

簡単のため，概要を図 2.6 に示す．図 2.6 全体が多重集合 $\mathcal{W}_{\mathcal{J}_{mid}}$ であり，各行がそれぞれ $\mathcal{W}_{\mathcal{J}_{mid}}$ の要素を示す．各行 (44 ビット) はそれぞれ $\mathbf{w}_{\mathcal{J}_{mid}}$ の値を示し，縦の太線で区切られている 4 ビットは $\mathbf{w}_{r,i}$ の値を示す．$\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}$ ($\mathbf{w}_{r,i}$ を除いた 40 ビットの値) に対し，$\mathcal{W}_{r,i}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}]$ は太線で囲われた 4 ビットの値の集合である．これが $\mathbb{F}_2^4$ の任意の要素を同じ個数有するならば，$\mathbf{w}_{r,i}$ は $\mathcal{W}_{\mathcal{J}_{mid}}$ において Ǎ 状態である．

多変数ワードの選択平文集合から得られる中間値の状態は定義 2.3 を用いて以下のように判定される．ただし，中間値 $\mathbf{w}_{r,i}$ は鍵 $\mathbf{k}$ 及び Č 状態のワードを定数とした関数 F への入力 $\mathbf{w}_{\mathcal{J}_{mid}}$ で一意に定まると仮定し，その多重集合は $\mathcal{W}_{r,i} = \{\mathbf{w}^{o}_{r,i} | \mathbf{w}^{o}_{r,i} = \mathsf{F}(\mathbf{w}^{o}_{\mathcal{J}_{mid}}), \mathbf{w}^{o}_{\mathcal{J}_{mid}} \in \mathcal{W}_{\mathcal{J}_{mid}}\}$ とする．



多重集合 $\mathcal{W}_{\mathcal{J}_{mid}}$

図 2.6　中間値の多重集合 $\mathcal{W}_{\mathcal{J}_{mid}}$ において Ă 状態であるワードの例

Č 判定: C 状態と同様．変数ワードから影響を受けない全ての中間値は Č 状態と判定できる．

Ă 判定: 選択平文集合を入力して得られる中間値の多重集合 $\mathcal{W}_{\mathcal{J}_{mid}}$ において $\mathbf{w}_{r',i'}$ が Ă 状態であるとする．$\mathcal{J}_{mid}$ に含まれる $\mathbf{w}_{r',i'}$ 以外のワードの値を定数とみなした $\mathbf{w}_{r,i} = \mathsf{R}(\mathbf{w}_{r',i'})$ なる R が全単射ならば，$\check{\mathbf{W}}_{r,i} = \check{\mathtt{A}}$ と判定できる．なお，$\mathcal{W}_{\mathcal{J}_{mid}}$ において Ă 状態である任意の $\mathbf{w}_{r,i}$ $((r,i) \in \mathcal{J}_{mid})$ は Ă 状態と判定できる．

理由: まず，$\mathcal{W}_{\mathcal{J}_{mid}}$ において Ă 状態である任意の $\mathbf{w}_{r',i'}$ は Ă 状態と判定できることを示す．$\mathcal{W}_{r',i'}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\backslash\{(r',i')\}}] = \{\mathbf{w}^{o}_{r',i'} | (\mathbf{w}^{o}_{r',i'}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\backslash\{(r',i')\}}) \in \mathcal{W}_{\mathcal{J}_{mid}}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\backslash\{(r',i')\}} = const\}$ とする．定義 2.3 より，任意の $\mathbf{w}^{o'}_{\mathcal{J}_{mid}\backslash\{(r',i')\}}$ に対し，$\mathcal{W}_{r',i'}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\backslash\{(r',i')\}}]$ には $\mathbb{F}_2^{n_{ward}}$ の任意の要素が同じ個数ずつ含まれる．したがって，$\mathcal{W}_{r',i'}$ には $\mathbb{F}_2^{n_{ward}}$ の任意の要素が同じ個数ずつ含まれるので，$\check{\mathbf{W}}_{r',i'} = \check{\mathtt{A}}$ と判定できる．

次に，$\mathcal{J}'_{mid} = \mathcal{J}_{mid}\backslash\{(r',i')\} \cup \{(r,i)\}$ とする．全単射の特性から $\mathcal{W}_{\mathcal{J}'_{mid}}$ において $\mathbf{w}_{r,i}$ は Ă 状態である．したがって，$\check{\mathbf{W}}_{r,i} = \check{\mathtt{A}}$ と判定できる．

$\check{\mathtt{E}}$ 判定: $\mathbf{w}_{r',i'}$ によって $\mathbf{w}_{r,i}$ が変化しないとする．$\mathbf{w}_{r',i'}$ が $\mathcal{W}_{\mathcal{J}_{mid}}$ において $\check{\mathtt{A}}$ 状態であるとき，$\check{\mathtt{W}}_{r,i} = \check{\mathtt{E}}$ と判定できる．

理由: $\mathcal{W}_{r',i'}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r',i')\}}] = \{\mathbf{w}^{o}_{r',i'} | (\mathbf{w}^{o}_{r',i'}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r',i')\}}) \in \mathcal{W}_{\mathcal{J}_{mid}}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r',i')\}} = const\}$ とする．定義 2.3 より，任意の $\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r',i')\}}$ に対し，$\mathcal{W}_{r',i'}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r',i')\}}]$ には $\mathbb{F}_2^{n_{ward}}$ の任意の要素が同じ個数ずつ含まれる．ここで，$\mathcal{W}_{r,i}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}] = \{\mathbf{w}^{o}_{r,i} = \mathsf{F}(\mathbf{w}^{o}_{r',i'}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r',i')\}}) | (\mathbf{w}^{o}_{r',i'}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r',i')\}}) \in \mathcal{W}_{\mathcal{J}_{mid}}, \mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}} = const\}$ とする．$\mathbf{w}_{r',i'}$ によって $\mathbf{w}_{r,i}$ の値は変化しないので，任意の $\mathbf{w}^{o'}_{\mathcal{J}'_{mid}\setminus\{(r',i')\}}$ に対して $\mathcal{W}_{r,i}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}]$ は $\mathbb{F}_2^{n_{ward}}$ の単一の要素から成る．また，$\mathcal{W}_{r',i'}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r',i')\}}]$ には $\mathbb{F}_2^{n_{ward}}$ の任意の要素が同じ個数ずつ含まれることから，$\mathcal{W}_{r,i}[\mathbf{w}^{o'}_{\mathcal{J}_{mid}\setminus\{(r,i)\}}]$ の要素数は $2^{n_{ward}}$ の整数倍である．したがって，$\mathbb{F}_2^{n_{ward}}$ の任意の要素が $\mathcal{W}_{r,i}$ に含まれる個数は $2^{n_{ward}}$ の整数倍となり，これは偶数であるので $\check{\mathtt{W}}_{r,i} = \check{\mathtt{E}}$ と判定できる．

$\check{\mathtt{B}}$ 判定: 中間値 $\mathbf{w}_{r,i}$ が複数の $\check{\mathtt{A}}$ または $\check{\mathtt{E}}$ 状態の中間値の XOR で求まるならば，$\check{\mathtt{W}}_{r,i} = \check{\mathtt{B}}$ と判定できる．

$\check{\mathtt{U}}$ 判定: その他の場合 $\check{\mathtt{W}}_{r,i} = \check{\mathtt{U}}$ と判定する．

2.3.2 複数ワードを入出力とする全単射の探索

Algorithm 2.2 に全単射の探索を行う BijFuncSearch について示す．Integral 特性探索の際，まず複数ワードを入出力とする全単射を探索する．全単射の探索は各段で逐次的に行う．

まず，段関数を解析し，部分的な全単射とその成立条件を確認する．例えば，関数 FS (式 (2.5) 参照) には図 2.5 で示した全単射が存在する．解析によって得られた全単射を $\mathsf{F}_1, \mathsf{F}_2, ..., \mathsf{F}_{m_{bij}}$ とする．

段関数の入力における変数ワードの添字集合を $\mathcal{J}_{in}$ とする．全単射 $\mathsf{F}_j : \mathbb{F}_2^{m \cdot n_{ward}} \rightarrow \mathbb{F}_2^{m \cdot n_{ward}}$ の入力ワードがすべて $\mathcal{J}_{in}$ に含まれるならば，$\mathbb{F}_2^{m \cdot n_{ward}}$ の任意の要素が入力される．全単射の特性から，F_j から $\mathbb{F}_2^{m \cdot n_{ward}}$ の任意の要素が出力される．したがって，この出力ワードは次の段関数入力において変数ワードとみなせる．そのため，次の段関数入力の変数ワードとなる添字集合 $\mathcal{J}_{out}$ に加える．

Algorithm 2.2 全単射の探索

function BijFuncSearch($\check{\mathcal{S}}_{\mathcal{J}}, \mathcal{J}$)
段関数に含まれる全単射を $\mathsf{F}_1, \mathsf{F}_2, ..., \mathsf{F}_{m_{bij}}$ とする．
$\mathcal{J}_{in} \leftarrow \mathcal{J}$
for $r = 1 \rightarrow r_{all}$ **do**
for $j = 1 \rightarrow m_{bij}$ **do**
if F_j の入力ワードが全て $\mathcal{J}_{in}$ に含まれる． **then**
F_j の出力ワードの添字を $\mathcal{J}_{out}$ に加える．
else
F_j の入力ワードと $\mathcal{J}_{in}$ に含まれるワードの添字を i とする．
$\mathcal{J}_{bij} \leftarrow \mathcal{J}_{bij} \cup \{(r-1, i)\}$
end if
end for
for $j \in \mathcal{J}_{out}$ **do**
$\check{\mathtt{W}}_{r,j} \leftarrow \check{\mathtt{A}}$
end for
$\mathcal{J}_{in} \leftarrow \mathcal{J}_{out}$
end for
return $(\check{\mathcal{S}}_{\mathcal{J}}, \mathcal{J}_{bij})$
end function

もし，全単射 F_j の入力ワードがすべて $\mathcal{J}_{in}$ に含まれないならば，全単射の特性を利用できない．そこで，F_j の入力ワードの添字集合と $\mathcal{J}_{in}$ の共通部分を $\mathcal{J}_{bij}$ という添字集合に加える．

上記で示した BijFuncSearch の手順で

$$\mathbf{w}_{\mathcal{J}_{out} \cup \mathcal{J}_{bij}} = \mathsf{G}(\mathbf{v}_{\mathcal{J}}) \tag{2.11}$$

なる全単射 G が探索できることを帰納的に示す．なお，平文 $\mathbf{v}_{\mathcal{J}}$ は添字が $\mathcal{J}$ に含まれるワードが変数であり，G において鍵等は定数として扱われる．初期状態において $\mathcal{J}_{in}$ は平文の変数ワードの添字集合である．$\mathbf{w}_{\mathcal{J}_{out} \cup \mathcal{J}_{bij}} = (\mathbf{w}_{\mathcal{J}_{out}}, \mathbf{v}_{\mathcal{J}_{bij}})$ とすれば，式 (2.11) の関数 F は全単射 $\mathsf{F}_1, \mathsf{F}_2, ..., \mathsf{F}_{m_{bij}}$ の一部と恒等関数 $\mathbf{v}_{\mathcal{J}_{bij}} \mapsto \mathbf{v}_{\mathcal{J}_{bij}}$ の並

列処理を行うので，全単射である．次に，r 段目出力において，式 (2.11) の全単射が探索されていると仮定する．ここで，$\mathbf{w}_{\mathcal{J}_{out}\cup\mathcal{J}_{r+1}} = \mathsf{G}_{r+1}(\mathbf{w}_{\mathcal{J}_{in}})$ なる関数 G_{r+1} を考える．なお，$\mathcal{J}_{r+1}$ は $r+1$ 段目出力で新たに $\mathcal{J}_{bij}$ の要素となるワードの添字集合とする．G_{r+1} は全単射 $\mathsf{F}_1, \mathsf{F}_2, ..., \mathsf{F}_{m_{bij}}$ の一部と恒等関数 $\mathbf{w}_{\mathcal{J}_{r+1}} \mapsto \mathbf{w}_{\mathcal{J}_{r+1}}$ の並列処理を行うので全単射である．G_{r+1} と r 段目出力で既に探索された全単射の合成は全単射である．したがって，$r+1$ 段目出力においても，全単射が探索される．

BijFuncSearch の手順を繰り返すことで $|\mathcal{J}_{in}| = |\mathcal{J}_{out}| = 0$ 及び $|\mathcal{J}_{bij}| = |\mathcal{J}|$ という状態になり，$\mathbf{w}_{\mathcal{J}_{bij}} = \mathsf{G}_{bij}(\mathbf{v}_{\mathcal{J}})$ なる全単射 G_{bij} が探索される．図 2.3 に示したブロック暗号を用いた BijFuncSearch の実行例を図 2.7 に示す．黒及び灰色のワードがそれぞれ $\mathcal{J}_{in}$ 及び $\mathcal{J}_{bij}$ に含まれるワードである．これらのワードが逐次的に出力側に更新される．$\mathcal{J}_{in}$ に含まれるワードの中で，いずれかの全単射の入力となるワードについては，その出力ワードの添字集合 $\mathcal{J}_{out}$ が次の段における $\mathcal{J}_{in}$ となる．また，全単射の入力とならないものは $\mathcal{J}_{bij}$ に含める．一番右側の図では $\mathcal{J}_{in}$ に含まれるワードがなくなることで $\mathcal{J}_{bij}$ が確定するので終了する．

途中処理で得られた全単射 (G_{bij} も含む) の出力 $\mathbf{w}_{\mathcal{J}_{out}\cup\mathcal{J}_{bij}}$ の多重集合 $\mathcal{W}_{\mathcal{J}_{out}\cup\mathcal{J}_{bij}}$ において任意の $\mathbf{w}_{r,i}$ $((r,i) \in \mathcal{J}_{out} \cup \mathcal{J}_{bij})$ は $\check{\mathtt{A}}$ 状態である (定義 2.3 参照)．したがって，これらの中間値は $\check{\mathtt{A}}$ 状態と判定でき，かつ，これを利用して第 2.3.3 項に示す関数 InitSearch において平衡ワードを探索する．

2.3.3 Integral 特性の初期探索

Algorithm 2.3 に初期探索を行う InitSearch を示す．第 2.3.2 項で示した BijFuncSearch から得られた全単射が存在する中間値の添字集合 $\mathcal{J}_{bij}$ (第 2.3.2 項参照) を用いて $\check{\mathcal{S}}_{\mathcal{J}}$ を更新する．多重集合 $\mathcal{W}_{\mathcal{J}_{bij}}$ において任意の $\mathbf{w}_{r,i}$ $((r,i) \in \mathcal{J}_{bij})$ は $\check{\mathtt{A}}$ 状態であり，第 2.3.1 項で示した $\check{\mathtt{A}}$ 及び $\check{\mathtt{E}}$ の判定に利用できる．まず，$\mathcal{J}_{bij}$ の任意の要素 (r', i') に対し，第 2.2.1 項で示した 1 変数ワードの選択平文集合による Integral 特性探索を行う．この特性探索は $\mathbf{w}_{r',i'}$ のみを変数ワード，$\mathcal{J}_{bij}$ に含まれるその他のワードは定数ワードとして実行する．1 変数ワードの選択平文集合による Integral

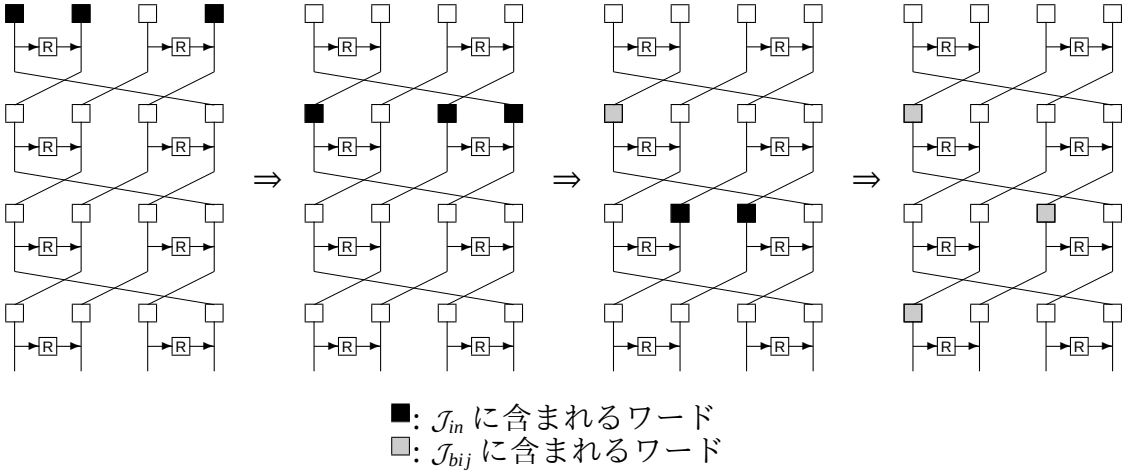


図 2.7　BijFuncSearch(Algorithm 2.2) の実行例

特性探索結果から，以下のとおり $\check{\mathtt{A}}$ 及び $\check{\mathtt{E}}$ の判定を行う (第 2.3.1 項の各状態の判定法を参照).

$\check{\mathtt{A}}$ 判定: $\mathtt{W}_{r,i} = \mathtt{A}$ なる $\mathbf{w}_{r,i}$ では，$\mathtt{A}$ 状態の定義から $\mathbf{w}_{r,i} = \mathsf{R}(\mathbf{w}_{r',i'})$ なる全単射 R が存在する．$\mathcal{W}_{\mathcal{J}_{bij}}$ において $\mathbf{w}_{r',i'}$ は $\check{\mathtt{A}}$ 状態なので，$\check{\mathtt{W}}_{r,i} = \check{\mathtt{A}}$ と判定できる．

$\check{\mathtt{E}}$ 判定: $\mathtt{W}_{r,i} = \mathtt{C}$ であるため，$\mathbf{w}_{r,i}$ は $\mathbf{w}_{r',i'}$ によって変化しない．$\mathbf{w}_{r',i'}$ が $\mathcal{W}_{\mathcal{J}_{bij}}$ において $\check{\mathtt{A}}$ 状態であるため，$\check{\mathtt{W}}_{r,i} = \check{\mathtt{E}}$ と判定できる．

$\mathcal{J}_{bij}$ に含まれる全てのワードについて $\check{\mathtt{A}}$ 及び $\check{\mathtt{E}}$ の判定を行った後，$\check{\mathcal{S}}_{\mathcal{J}}$ を更新する．具体的には，初期値である $\check{\mathtt{U}}$ 状態のままである中間値について，$\check{\mathtt{B}}$ または $\check{\mathtt{C}}$ 状態かどうかを表 2.2 及び 2.3 を用いて確認する．

2.3.4　Integral 特性の追加探索

Algorithm 2.4 に追加探索を行う AddSearch を示す．ここでは，$\check{\mathcal{S}}_{\mathcal{J}}$ の更新を行うが，対象は $\check{\mathtt{B}}$ 及び $\check{\mathtt{U}}$ 状態の中間値である．これらが $\check{\mathtt{A}}$ 状態として判定できるかを確認する．まず，対象となる中間値 $\mathbf{w}_{r,i}$ について，

$$\mathbf{w}_{r,i} = \mathsf{F}''(\mathbf{w}_{\mathcal{J}_{mid}}, \mathbf{w}_{\overline{\mathcal{J}_{mid}}}) \tag{2.12}$$

Algorithm 2.3 Integral 特性の初期探索

function InitSearch($\check{\mathcal{S}}_{\mathcal{J}}, \mathcal{J}_{bij}$)
  for $(r', i') \in \mathcal{J}_{bij}$ **do**
    全ワードの状態集合 $\mathcal{S}_{\mathcal{J}}$ を初期化 $\forall \mathtt{W}_{r,i} \in \mathcal{S}_{\mathcal{J}}$, $\mathtt{W}_{r,i} \leftarrow \mathtt{U}$.
    $\mathtt{W}_{r',i'} \leftarrow \mathtt{A}$, $\mathtt{W}_{r',i''} \leftarrow \mathtt{C}$ $(i'' \in \{1, 2, ..., m_{all}\} \backslash \{i'\})$
    $\mathcal{S}_{\mathcal{J}}$ を表 2.1 を用いて更新する.
    for $r = (r' + 1) \rightarrow r_{all}$ **do**
      for $i = 1 \rightarrow m_{all}$ **do**
        if $\mathtt{W}_{r,i} = \mathtt{A}$ 及び $\check{\mathtt{W}}_{r,i} \neq \check{\mathtt{A}}$ **then**
          $\check{\mathtt{W}}_{r,i} \leftarrow \check{\mathtt{A}}$
        end if
        if $\mathtt{W}_{r,i} = \mathtt{C}$ 及び $\check{\mathtt{W}}_{r,i} \neq \check{\mathtt{C}}, \check{\mathtt{A}}$ **then**
          $\check{\mathtt{W}}_{r,i} \leftarrow \check{\mathtt{E}}$
        end if
      end for
    end for
  end for
  $\check{\mathcal{S}}_{\mathcal{J}}$ を表 2.2 を用いて更新する.
  return $\check{\mathcal{S}}_{\mathcal{J}}$
end function

と求まり，かつ，$\mathbf{w}_{\overline{\mathcal{J}_{mid}}}$ によって $\mathbf{w}_{r,i}$ が変化しないと仮定する (鍵等は定数としている)．なお，$\mathbf{w}_{\mathcal{J}_{mid}}$ 及び $\mathbf{w}_{\overline{\mathcal{J}_{mid}}}$ は r' 段目出力の中間値とし，$\mathcal{J}_{mid} \cup \overline{\mathcal{J}_{mid}} = \{(r', i) | i \in \{1, 2, ..., m_{all}\}, \mathcal{J}_{mid} \cap \overline{\mathcal{J}_{mid}} = \emptyset\}$ を満たす．この r' 段目出力の中間値で，$\mathcal{J}_{mid}$ に含まれるワードについて，1 変数ワードの選択平文集合を用いた Integral 特性探索を行う．このとき，$\mathbf{w}_{r',i'}$ $(i' \in \mathcal{J}_{mid})$ のみを変数ワード，その他を定数ワードとして実行する．実行結果から，対象である $\mathbf{w}_{r,i}$ が $\mathtt{A}$ 状態であることが分かったならば，中間値の多重集合 $\mathcal{W}_{\mathcal{J}_{mid}}$ において $\mathbf{w}_{r',i'}$ が $\check{\mathtt{A}}$ 状態であるかを確認する．このとき，以下を用いる．

命題 2.1. もし，下記の 2 つの条件を同時に満たす $(r', i') \in \mathcal{J}_{mid}$ 及び $(r'', i'') \in \mathcal{J}_{bij}$ が存在するならば，$\mathcal{W}_{\mathcal{J}_{mid}}$ において $\mathbf{w}_{r',i'}$ が $\check{\mathtt{A}}$ 状態である．

Algorithm 2.4 Integral 特性の追加探索

function AddSearch($\check{\mathcal{S}}_{\mathcal{J}}, \mathcal{J}_{bij}$)
  for $\check{\mathtt{W}}_{r,i} = \check{\mathtt{B}}$ または $\check{\mathtt{U}}$ なる (r, i) **do**
    for $r' = r \to 0$ **do**
      $\mathbf{w}_{r,i}$ に対し変数となる r' 段目ワードの添字集合を $\mathcal{J}_{mid}$ とする．
      for $i' \in \mathcal{J}_{mid}$ **do**
        全ワードの状態集合 $\mathcal{S}_{\mathcal{J}}$ を初期化，全ワードに $\mathtt{U}$ を代入．
        $\mathtt{W}_{r',i'} \leftarrow \mathtt{A}$，$\mathtt{W}_{r',i''} \leftarrow \mathtt{C}$ $(i'' \in \{1, 2, ..., m_{all}\} \backslash \{i'\})$
        $\mathcal{S}_{\mathcal{J}}$ を表 2.1 を用いて更新する．
        if $\mathtt{W}_{r,i} = \mathtt{A}$ **then**
          if InjCheck($\mathcal{J}_{bij}, \mathcal{J}_{mid}, \check{\mathcal{S}}_{\mathcal{J}}$) = 1 **then**
            for $(r'', i'') \in \mathcal{J}_{bij}$ **do**
              if InjCheck($\mathcal{J}_{bij} \backslash \{(r'', i'')\}, \mathcal{J}_{mid} \backslash \{(r', i')\}, \check{\mathcal{S}}_{\mathcal{J}}$) = 1 **then**
                $\check{\mathtt{W}}_{r,i} = \check{\mathtt{A}}$
              end if
            end for
          end if
        end if
      end for
    end for
  end for
  $\check{\mathcal{S}}_{\mathcal{J}}$ を表 2.2 を用いて更新する．
  return $\check{\mathcal{S}}_{\mathcal{J}}$
end function

(i) $\mathbf{w}_{\mathcal{J}_{mid}} = \mathsf{G}_{inj}(\mathbf{w}_{\mathcal{J}_{bij}})$ なる単射 G_{inj} が存在する．

(ii) $\mathbf{w}_{\mathcal{J}_{mid} \backslash \{(r',i')\}} = \mathsf{G}'_{inj}(\mathbf{w}_{\mathcal{J}_{bij} \backslash \{(r'',i'')\}})$ なる単射 G'_{inj} が存在する．

証明． G_{bij} が全単射なので，$\mathbf{W}_{\mathcal{J}_{bij} \backslash \{(r'',i'')\}}$ は $2^{n_{var}-n_{ward}}$ 個の異なる値を有する多重集合である．したがって，単射 G'_{inj} で求まる $\mathbf{w}_{\mathcal{J}_{mid} \backslash \{(r',i')\}}$ の多重集合 $\mathcal{W}_{\mathcal{J}_{mid} \backslash \{(r',i')\}}$ も同様に $2^{n_{var}-n_{ward}}$ 個の異なる値を有する．

ここで，$\mathcal{W}_{r',i'}[\mathbf{w}^{o'}_{\mathcal{J}_{mid} \backslash \{(r',i')\}}] = \{\mathbf{w}^{o}_{r',i'} | (\mathbf{w}^{o}_{r',i'}, \mathbf{w}^{o'}_{\mathcal{J}_{mid} \backslash \{(r',i')\}}) \in \mathcal{W}_{\mathcal{J}_{mid}}, \mathbf{w}^{o'}_{\mathcal{J}_{mid} \backslash \{(r',i')\}} = const\}$ とする．任意の $\mathbf{w}^{o'}_{\mathcal{J}_{mid} \backslash \{(r',i')\}}$ に対して，$\mathcal{W}_{r',i'}[\mathbf{w}^{o'}_{\mathcal{J}_{mid} \backslash \{(r',i')\}}]$ が l_i $(i \in \{1, 2, ..., 2^{n_{var}-n_{ward}}\})$

個の異なる値から成ると仮定する．ここで，$\mathbf{w}_{r',i'}$ は $\mathbb{F}_2^{n_{word}}$ の要素なので，$l_i \leq 2^{n_{word}}$ である．単射 G_{inj} で求まる $\mathcal{W}_{\mathcal{J}_{mid}}$ は $2^{n_{var}}$ 個の互いに異なる値を有する．したがって，$\sum_{i=1}^{2^{n_{var}-n_{word}}} l_i = 2^{n_{var}}$ が成り立つ．任意の i に対し，$l_i = 2^{n_{word}}$ であれば $\sum_{i=1}^{2^{n_{var}-n_{word}}} l_i$ は最大値 $2^{n_{var}}$ となるので，$l_i = 2^{n_{word}}$ である．したがって，$\mathcal{W}_{\mathcal{J}_{mid}}$ において $\mathbf{w}_{r',i'}$ は $\check{\mathtt{A}}$ 状態である． □

$\mathcal{W}_{\mathcal{J}_{mid}}$ において $\mathbf{w}_{r',i'}$ が $\check{\mathtt{A}}$ 状態であり，かつ，$\mathbf{w}_{r,i}$ が $\mathbf{w}_{r',i'}$ を唯一の変数ワードとしたときに $\mathtt{A}$ 状態であることを確認できれば，$\mathbf{w}_{r,i}$ は $\check{\mathtt{A}}$ 状態であると判定できる (第 2.3.3 項の $\check{\mathtt{A}}$ 判定を参照)．ここで，命題 2.1 における単射が存在するかを確認する InjCheck を用いる．InjCheck については，第 2.3.5 項で後述する．

任意の $\check{\mathtt{B}}$ または $\check{\mathtt{U}}$ なる中間値について，$\check{\mathtt{A}}$ として判定できるかを確認したのち，最終的には InitSearch (Algorithm 2.3) と同様に状態集合 $\check{\mathcal{S}}_{\mathcal{J}}$ の更新を行う．この出力が Algorithm 2.1 で示した IPS-BP の出力となる．

2.3.5 単射の確認手順

Algorithm 2.5 に InjCheck を示す．この関数は $\mathbf{w}_{\mathcal{J}_{out}} = \mathsf{G}(\mathbf{w}_{\mathcal{J}_{in}})$ なる G が単射であるかを確認する．ここでは単射の定義の対偶を用いて，

$$\text{任意の } \mathbf{w}^{o}_{\mathcal{J}_{in}} \text{及び } \mathbf{w}^{o'}_{\mathcal{J}_{in}} \text{に対し，} \ \mathbf{w}^{o}_{\mathcal{J}_{out}} = \mathbf{w}^{o'}_{\mathcal{J}_{out}} \text{ならば } \mathbf{w}^{o}_{\mathcal{J}_{in}} = \mathbf{w}^{o'}_{\mathcal{J}_{in}} \tag{2.13}$$

を確認する．中間値 1 ワード及び複数ワードの差分をそれぞれ $\Delta\mathbf{w}_{r,i}$ 及び $\Delta\mathbf{w}_{\mathcal{J}_{mid}}$ とする．なお，$\Delta\mathbf{w}_{r,i} = \mathbf{w}^{o}_{r,i} \oplus \mathbf{w}^{o'}_{r,i}$ とし，$(\mathbf{w}^{o}_{r,i}, \mathbf{w}^{o'}_{r,i})$ は平文組を入力した際に得られる中間値組とする．式 (2.13) の成立の有無を示す際，$\Delta\mathbf{w}_{r,i} \in \mathbb{F}_2^{n_{word}}$ が確定的に 0 となるかが重要となる．そこで，確定的に 0 となる場合は Δ_0，それ以外は $\Delta_?$ と表現し，これらを“差分の状態”とよぶ $(\Delta\mathbf{w}_{r,i} \in \{\Delta_0, \Delta_?\})$.

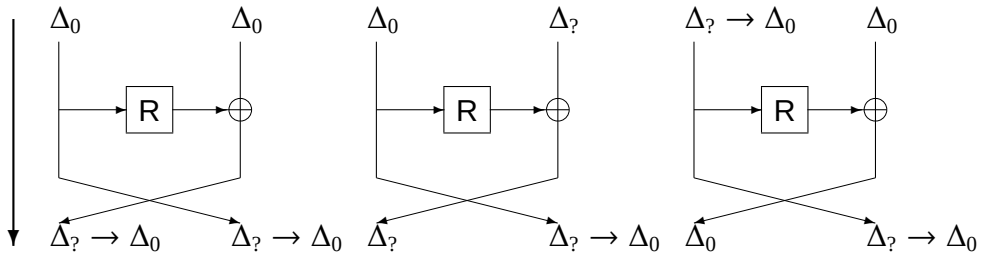
まず，$\Delta\mathbf{w}_{\mathcal{J}_{out}} = \Delta_0$ であると仮定する．この仮定から，入力の差分が確定的に 0 となるかを確認する．なお，すでに $\check{\mathtt{C}}$ 状態であると判定されている中間値では確定的に差分が 0 であるので，Δ_0 を代入する．Δ_0 が代入されていない，つまり確定的に差分が 0 であると確認されていない任意の中間値には $\Delta_?$ を代入する．

Algorithm 2.5 単射の確認手順

function InjCheck($\mathcal{J}_{in}, \mathcal{J}_{out}, \check{\mathcal{S}}_{\mathcal{J}}$)
段関数に含まれる全単射を $\mathsf{F}_1, \mathsf{F}_2, ..., \mathsf{F}_{m_{bij}}$ とする．
任意の (r, i) に対し $\Delta\mathbf{w}_{r,i} \leftarrow \Delta_?$．
任意の $(r, i) \in \mathcal{J}_{out}$ に対し $\Delta\mathbf{w}_{r,i} \leftarrow \Delta_0$．
任意の $\check{\mathcal{C}}$ 状態の中間値 $\mathbf{w}_{r,i}$ に対し $\Delta\mathbf{w}_{r,i} \leftarrow \Delta_0$．
do
for $r = r_{all} - 1 \rightarrow 0$ **do**
for $j = 1 \rightarrow m_{bij}$ **do**
F_j^{-1} の入出力で確定的に差分がゼロとなるワードに Δ_0 を代入．
end for
end for
for $r = 1 \rightarrow r_{all}$ **do**
for $j = 1 \rightarrow m_{bij}$ **do**
F_j の入出力で確定的に差分がゼロとなるワードに Δ_0 を代入．
end for
end for
while Δ_0 が 1 度でも代入された．
if 任意の $(r, i) \in \mathcal{J}_{in}$ に対し $\Delta\mathbf{w}_{r,i} = \Delta_0$ **then** **return** 1
else return 0
end if
end function

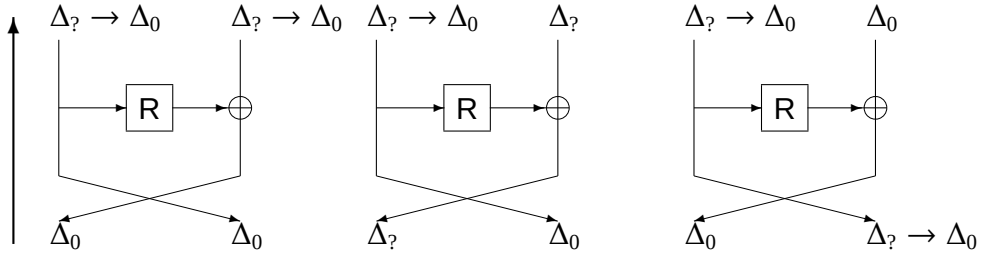
$\Delta\mathbf{w}_{\mathcal{J}_{in}} = \Delta_0$ を証明するために，$\mathbf{w}_{\mathcal{J}_{out}} = \mathsf{G}(\mathbf{w}_{\mathcal{J}_{in}})$ の途中処理である中間値について，Δ_0 が代入できるかを確認する．その際，出力側から入力側及び入力側から出力側の 2 つの方向で確認を行う．また，段関数に含まれる全単射 $\mathsf{F}_1, \mathsf{F}_2, ..., \mathsf{F}_{m_{bij}}$ 及びその逆関数について，入出力の差分の状態から確定的に Δ_0 が代入できるワードが存在するか確認する．

例として，図 2.8 及び 2.9 に関数 FS (式 (2.5) 参照) 及びその逆関数 FS^{-1} による差分の状態の変化について示し，これらをまとめたものを表 2.4 に示す．図 2.8 及び 2.9 は入出力が (a) から (c) のサブタイトルのような差分の状態のとき，$\Delta_?$ で



(a) $(\Delta_0, \Delta_0) \mapsto (\Delta_?, \Delta_?)$ (b) $(\Delta_0, \Delta_?) \mapsto (\Delta_?, \Delta_?)$ (c) $(\Delta_?, \Delta_0) \mapsto (\Delta_0, \Delta_?)$

図 2.8 関数 FS (入力から出力) による差分の状態の変化



(a) $(\Delta_0, \Delta_0) \mapsto (\Delta_?, \Delta_?)$ (b) $(\Delta_?, \Delta_0) \mapsto (\Delta_?, \Delta_?)$ (c) $(\Delta_0, \Delta_?) \mapsto (\Delta_?, \Delta_0)$

図 2.9 関数 FS の逆関数 FS^{-1} (出力から入力) による差分の状態の変化

表 2.4 関数 FS 及びその逆関数による差分の状態の変化

入力	出力	FS		FS^{-1}	
		入力	出力	入力	出力
(Δ_0, Δ_0)	$(\Delta_?, \Delta_?)$	(Δ_0, Δ_0)	(Δ_0, Δ_0)	(Δ_0, Δ_0)	(Δ_0, Δ_0)
$(\Delta_0, \Delta_?)$	$(\Delta_?, \Delta_?)$	$(\Delta_0, \Delta_?)$	$(\Delta_?, \Delta_0)$	-	-
$(\Delta_?, \Delta_0)$	$(\Delta_0, \Delta_?)$	(Δ_0, Δ_0)	(Δ_0, Δ_0)	-	-
$(\Delta_?, \Delta_0)$	$(\Delta_?, \Delta_?)$	-	-	$(\Delta_?, \Delta_0)$	$(\Delta_0, \Delta_?)$
$(\Delta_0, \Delta_?)$	$(\Delta_?, \Delta_0)$	-	-	(Δ_0, Δ_0)	(Δ_0, Δ_0)

あるワードに Δ_0 が代入される ($\Delta_? \to \Delta_0$) ことを示している．表 2.4 では FS 及び FS^{-1} による入出力の差分の状態の変化を示している．成立する理由が自明でない表 2.4 の 3 及び 5 行目 (図 2.8(c) 及び 2.9(c)) について示す．$(\mathbf{y}_1, \mathbf{y}_2) = \mathsf{FS}(\mathbf{x}_1, \mathbf{x}_2)$ において $\mathbf{x}_2^o \oplus \mathbf{x}_2^{o'} = 0$ 及び $\mathbf{y}_1^o \oplus \mathbf{y}_1^{o'} = \mathsf{R}(\mathbf{x}_1^o) \oplus \mathbf{x}_2^o \oplus \mathsf{R}(\mathbf{x}_1^{o'}) \oplus \mathbf{x}_2^{o'} = 0$ から，$\mathsf{R}(\mathbf{x}_1^o) = \mathsf{R}(\mathbf{x}_1^{o'})$ である必要がある．R が全単射なので，$\mathbf{x}_1^o = \mathbf{x}_1^{o'}$ であり，差分が 0 である必要がある．したがって，$\Delta\mathbf{x}_1$ 及び $\Delta\mathbf{y}_2$ に Δ_0 が代入される．

上記の処理を複数回繰り返すと，これ以上 Δ_0 が代入できる中間値が存在しなくなる．この段階で $\Delta\mathbf{w}_{\mathcal{J}_{in}} = \Delta_0$ かどうかを確認し，真であれば単射であると決定し，これを示す 1 を出力する．

2.3.6 計算量の評価

変数ワードが $m_{all} - 1$ 個である選択平文集合を入力した際の計算量について評価する．その際，他の演算と比べて実行時間の大きいテーブル参照 (TLU) の回数の上界を評価する．表 2.5 に各関数の TLU の回数及び使用するテーブルサイズの上界を示す．m_{F} は全単射 F_j の入力ワード数の最大値であり，関数 FS (式 (2.5) 参照) から成る段関数では $m_{\mathsf{F}} = 2$ である．

BijFuncSearch: 高々 $r_{all} \cdot m_{all}$ 個の中間値に対して全単射を探索する．その際，図 2.5 のように入力ワードが変数または定数である場合の出力の状態を表として用いる．これは，m_{F} に対してテーブルサイズは高々 $2^{m_{\mathsf{F}}}$ である．

InitSearch: 任意の $\mathcal{J}_{bij}$ ($|\mathcal{J}_{bij}| < m_{all}$) に含まれる中間値に対し，1 変数ワードの選択平文集合による Integral 特性探索を行う．1 変数ワードの選択平文集合による Integral 特性探索は高々 $r_{all} \cdot m_{all}$ 個の中間値に対して表 2.1 を適用して行う．したがって，高々 $r_{all} \cdot (m_{all})^2$ 回 TLU を実行する．また，最後に表 2.2 を用いて $r_{all} \cdot m_{all}$ 個の中間値の状態の更新を行う．表 2.2 が表 2.1 よりもサイズが大きいため，テーブルサイズの上界は 25 となる．

表 2.5　各関数の TLU の回数及びテーブルサイズの上界

関数	TLU の回数の上界	テーブルサイズ
BijFuncSearch	$r_{all} \cdot m_{all}$	$2^{m_{\mathsf{F}}}$
InitSearch	$r_{all} \cdot (m_{all})^2 + r_{all} \cdot m_{all}$	25 (表 2.1)
InjCheck	$2 \cdot (r_{all} \cdot m_{all})^2$	$2^{2 \cdot m_{\mathsf{F}}}$
AddSearch	$2 \cdot (r_{all})^4 \cdot (m_{all})^5 + 2 \cdot (r_{all} \cdot m_{all})^4$ $+(r_{all} \cdot m_{all})^3 + r_{all} \cdot m_{all}$	$\max\{2^{2 \cdot m_{\mathsf{F}}}, 25\}$

m_{F}:全単射 F_i の入力ワード数の最大値

InjCheck: 任意の $r_{all} \cdot m_{all}$ 個の差分が確定的に Δ_0 となるかを確認する．一回の試行で少なくとも 1 個の中間値に Δ_0 が代入されるので，高々 $r_{all} \cdot m_{all}$ 回の試行で停止する．したがって，$2 \cdot (r_{all} \cdot m_{all})^2$ 回 TLU を実行する (表 2.4 のような表をテーブルとする)．全単射の入力及び出力のワード数 $2 \cdot m_{\mathsf{F}}$ に対してテーブルサイズが $2^{2 \cdot m_{\mathsf{F}}}$ 必要になる．

AddSearch: 任意の $\check{\mathtt{B}}$ または $\check{\mathtt{U}}$ 状態である高々 $r_{all} \cdot m_{all}$ 個の中間値の状態の更新を行う．対象の中間値 $\mathbf{w}_{r,i}$ の入力である r' 段目出力 ($r' \in \{1, 2, ..., r_{all}\}$) の中間値 $\mathbf{w}_{\mathcal{J}_{mid}}$ において，$\mathbf{w}_{r',i'}$ $((r', i') \in \mathcal{J}_{mid})$ に対し 1 変数ワードの選択平文集合による Integral 特性探索及び InjCheck の呼び出しを $m_{all} + 1$ 回行う．最後に，表 2.2 を用いて $r_{all} \cdot m_{all}$ 個の中間値の状態の更新を行う．したがって，

$$\begin{aligned} & r_{all} \cdot m_{all} \cdot \Big(r_{all} \cdot \Big(\big(r_{all} \cdot m_{all} + (m_{all} + 1) \cdot \big(2 \cdot (r_{all} \cdot m_{all})^2\big)\big)\Big)\Big) + r_{all} \cdot m_{all} \\ & = 2 \cdot (r_{all})^4 \cdot (m_{all})^5 + 2 \cdot (r_{all} \cdot m_{all})^4 + (r_{all} \cdot m_{all})^3 + r_{all} \cdot m_{all} \end{aligned} \tag{2.14}$$

と上界が定まる．

本論文では暗号化関数の処理回数として計算量を定義している．暗号化関数の処理はおおよそ $r_{all} \cdot m_{all}$ 回の TLU 処理と同等と考えられる．したがって，計算量は表 2.5 の $1/(r_{all} \cdot m_{all})$ 程度と見積もられる．これに対し，IPS-PM では任意の $r_{all} \cdot m_{all}$ 個の中間値に対して TLU を行い第 2.2.1 項及び第 2.2.2 項の処理を行うので，線形時間で実行できる．IPS-BP の計算量は IPS-PM よりも大きいが，多項式時間で抑えられる．なお，ブロック暗号において想定されるパラメータの上界は $m_{all} \leq 32$ 及び $r_{all} \leq 100$ 程度であり，IPS-BP は汎用計算機でも実行が可能である．

2.4 TWINE 及び LBlock への適用

IPS-PM の先行研究 [39][43] との比較のため，TWINE[43] 及び LBlock[52] に IPS-BP を適用する．例として，TWINE に対して $\mathcal{J} = \{1, 2, ..., 15\}$ なる選択平文集合を入力した際の中間値の状態 (図 2.10) を用いて例示する．なお，8 段目から 12 段目出力の全中間値が Ă 状態である．

BijFuncSearch: $\mathbf{v}_{\mathcal{J}} \mapsto \mathbf{w}_{\mathcal{J}_{bij}}$ なる全単射が得られる中間値の添字集合 $\mathcal{J}_{bij}$ を得る．$\mathcal{J}_{bij}$ に含まれる中間値は Ă 状態であり，図中では四角で囲われた Ă で示す．また，BijFuncSearch を実行する過程で $\mathcal{J}_{out}$ の要素となった中間値についても Ă 状態であり，図中では下線が引かれた Ă で示す．

InitSearch: BijFuncSearch で得られた $\mathcal{J}_{bij}$ を用いて Ă 及び Ĕ 状態となる中間値を探索する (図中では Ĕ 状態の中間値は発見されなかった)．この過程で Ă 状態として判定された中間値は四角や下線等がない Ă で示す．表 2.2 を用いて最後に Ă や Č 状態の中間値が判定される．

AddSearch: Ă や Ŭ 状態の中間値について，Ă 状態として判定できるかを確認する．新たに Ă 状態と判定された中間値は図中で破線の四角で囲われている Ă で示

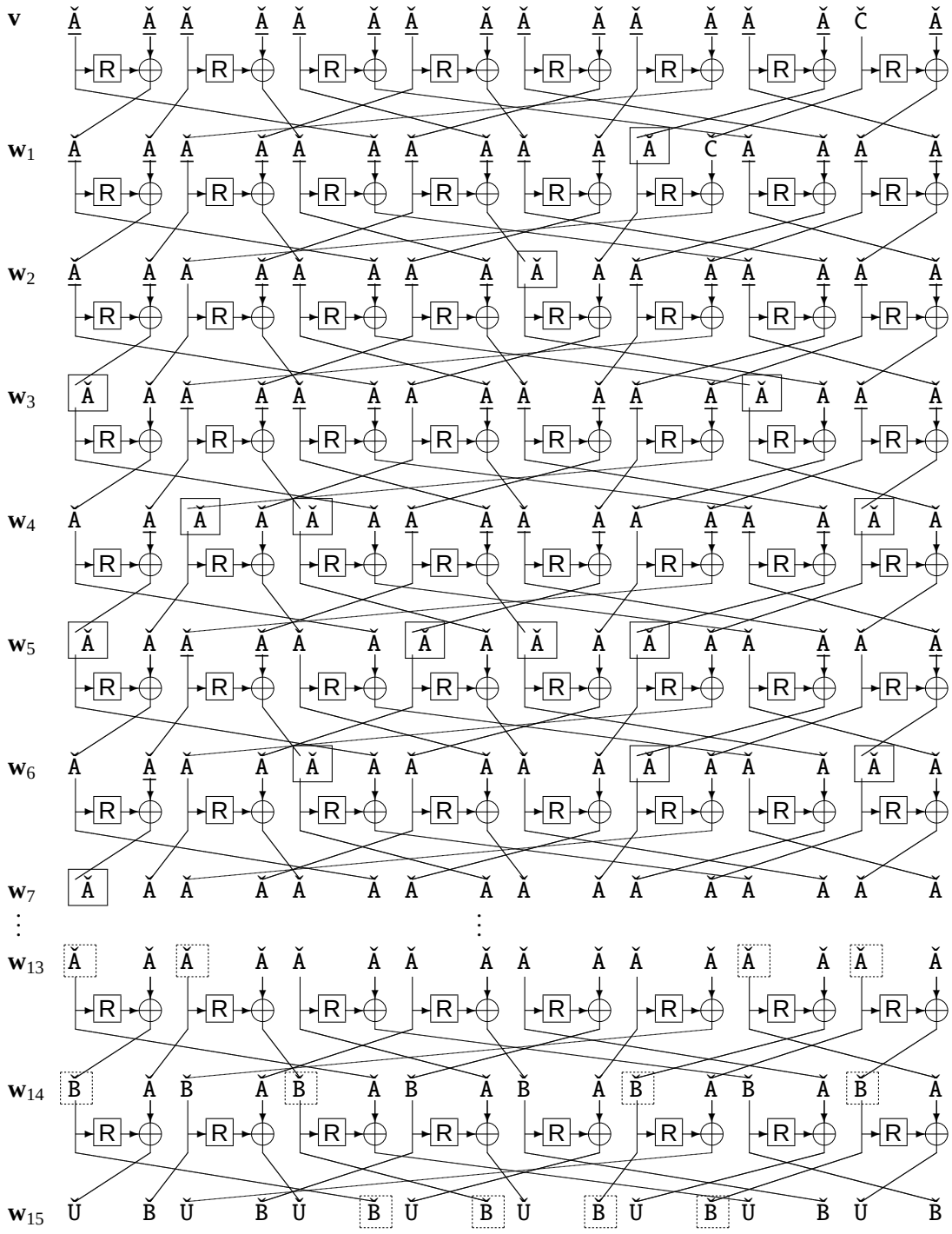


図 2.10　TWINE への全単射の特性を利用した Integral 特性探索手法 (IPS-BP) の適用例

す．表 2.2 を用いて最後に $\check{\mathtt{B}}$ や $\check{\mathtt{U}}$ 状態の中間値が見つかり，これらについても破線の四角で囲われている．

例として，本来 $\check{\mathtt{B}}$ 状態として判定された $\mathbf{w}_{13,1}$ が $\check{\mathtt{A}}$ 状態と判定される過程について示す．$\mathbf{w}_{13,1}$ を $\mathbf{w}_{13,1} = \mathsf{F}(\mathbf{w}_7)$ と表現したとき，$\mathbf{w}_{7,10}$ によって $\mathbf{w}_{13,1}$ は変化しない．$\mathcal{J}_{mid} = \{1, 2, ..., 16\} \backslash \{10\}$ とする．$\mathtt{W}_{7,12} = \mathtt{A}$，$\mathtt{W}_{7,i} = \mathtt{C}$ $((7, i) \in \mathcal{J}_{mid} \backslash \{(7, 12)\})$ を代入し，表 2.1 を用いると $\mathtt{W}_{13,1} = \mathtt{A}$ であることが分かる．InjCheck を用いると，$\mathbf{w}_{\mathcal{J}_{mid}} = \mathsf{G}_{inj}(\mathbf{w}_{\mathcal{J}_{bij}})$ なる G_{inj} が単射である．さらに，$\mathbf{w}_{\mathcal{J}_{mid} \backslash \{(7,12)\}} = \mathsf{G}'_{inj}(\mathbf{w}_{\mathcal{J}_{bij} \backslash \{(6,11)\}})$ なる G'_{inj} が単射である．したがって，$\mathcal{W}_{\mathcal{J}_{mid}}$ において $\mathbf{w}_{7,12}$ は $\check{\mathtt{A}}$ 状態 (定義 2.3 参照) であり，$\check{\mathtt{W}}_{13,1} = \check{\mathtt{A}}$ と判定できる．

結果として，$\mathcal{W}_{15}^{\{1,3,5,7,9,11,13,15\}} = \mathsf{IP}_{15}(\mathcal{V}_{\{1,2,...,16\} \backslash \{15\}})$ が得られる．また，平文の奇数ワード $\mathbf{v}_i$ $(i \in \{1, 3, 5, ..., 15\})$ を唯一の定数ワードとした選択平文集合から以下が得られる．

$$\mathcal{W}_{15}^{\{1,3,5,7,9,11,13,15\}} = \mathsf{IP}_{15}(\mathcal{V}_{\{1,2,...,16\} \backslash \{i\}}) \tag{2.15}$$

同様に，LBlock においては平文の左側のワード $\mathbf{v}_i$ $(i \in \{1, 2, ..., 8\})$ を唯一の定数ワードとした選択平文集合から以下が得られる．

$$\mathcal{W}_{15}^{\{8,9,...,16\}} = \mathsf{IP}_{15}(\mathcal{V}_{\{1,2,...,16\} \backslash \{i\}}) \tag{2.16}$$

これらの結果は IPS-PM [25] による探索結果よりも平衡ワードの数が多い．IPS-PM では 15 段目出力における平衡ワード数は 4 であるのに対し [43][39]，IPS-BP では 8 個の平衡ワードが探索できている．

次に，IPS-BP による探索結果と IPS-DP [44][46] による探索結果を比較する．Xiang らは IPS-DP を TWINE 及び LBlock に適用して Integral 特性を探索した [53]．その際，選択平文集合はビット単位で変数を設定した $\mathcal{V}_{\mathcal{I}}$ $(\mathcal{I} = \{2, 3, ..., 64\})$ で探

索を行っている．これにより，TWINE で

$$\mathcal{W}_{16}^{\{1,3,5,7,9,11,13,15\}} = \mathsf{IP}_{16}(\mathcal{V}_{\mathcal{I}}) \tag{2.17}$$

という結果が，LBlock で

$$\mathcal{W}_{16}^{\{8,9,\ldots,16\}} = \mathsf{IP}_{16}(\mathcal{V}_{\mathcal{I}}) \tag{2.18}$$

という結果が得られた．IPS-BP の探索結果と比べると，平衡ワードは同一であるが成立する段数が増加していることが分かる．このように，IPS-DP ではブール関数の特性を考慮することで，IPS-BP よりも多く平衡ワードを探索できる．しかし，IPS-DP は平文長またはワード数に対して指数関数時間の計算量を要する．Xiang らは混合整数線形計画法 [51] を用いて TWINE で 2.6 分，LBlcok で 4.9 分 (CPU : Intel Core i7-2600 3.40GHz，メモリ : 8GByte) で探索を実行した [53]．一方，IPS-BP では TWINE 及び LBlock の Integral 特性探索が数秒で実行できる (CPU : Intel Core i7-3770 3.40GHz，メモリ：16GByte).

IPS-DP よりも IPS-BP を使用することが適当と考えられる場面を 2 つ示す．ひとつは，暗号開発の初期段階である．このような段階では，暗号化関数の構成法について多くの候補が存在する．そこで，高速な IPS-BP で Integral 特性を探索することで Integral 攻撃耐性の低い構成法を除外することができる．もうひとつは，平文長が大きいブロック暗号の評価である．将来においてより安全性の高い平文長が 256 ビットや 512 ビットのブロック暗号が提案される可能性がある．平文長またはワード数に対して指数関数時間を要する IPS-DP では探索が困難となる可能性が高い．これに対し，ワード数に対して多項式時間で実行が可能な IPS-BP は有利に探索が実行可能であると考える．

2.5 第 2 章のまとめ

アルゴリズムの安全性評価として，Integral 攻撃における IPS-BP を示した．IPS-BP は IPS-PM と同様に全単射の特性のみを用いて探索を行うが，IPS-PM よりも多くの平衡ビットを探索することができる．しかし，ブール関数の特性を考慮して探索を行う IPS-DP に比べると，探索できる平衡ビットは少ない．実行に要する計算量を IPS-PM 及び IPS-DP と比較すると IPS-BP は中間に位置する．IPS-DP が適用できない場面として，暗号開発の初期段階や平文長が大きいブロック暗号の評価が挙げられる．このような場面において，多項式時間で実行が可能な IPS-BP は有効であり，将来的に安全なアルゴリズムの開発に貢献する．

第3章

差分バイアス攻撃に対する耐性評価手法

本論文では実装の安全性評価として，サイドチャネル攻撃に対する形式的評価に注目する．なお，サイドチャネル攻撃の仮定について付録 I に示す．本章では，以下に示す漏洩モデルを仮定する (概要を図 3.1 に示す)[9].

定義 3.1 (ランダム漏洩モデル)**.** ランダム漏洩モデル (RL (Random Leakage) モデル) において，攻撃者は任意に選択した平文 $\mathbf{v}$ に対応する漏洩値 $l = \mathsf{f}(\mathbf{v}, \mathbf{k})$ を測定することができる．各測定時に $\mathsf{f} : \mathbb{F}_2^{n_{all}+n_{key}} \to \mathbb{F}_2$ は中間値を出力する関数全体から乱択され，かつ，攻撃者にとって漏洩値がどの中間値に相当するかは未知とする．なお，漏洩値 l が 1 ビットである場合をビット単位 RL モデル，1 ワードの場合 (f のかわりに $\mathsf{F} : \mathbb{F}_2^{n_{all}+n_{key}} \to \mathbb{F}_2^{n_{ward}}$ が乱択される) をワード単位 RL モデルとよぶ.

RL モデルにおいて安全性評価を行い，対策が行える既知の攻撃としてはマルウェア等によるメモリ内容の盗聴が考えられる．これはソフトウェア実装された暗号化処理を対象とした攻撃で，中間値や鍵の情報が保存されたメモリに攻撃者がアクセス可能とする．ただし，得られた情報がどの処理において使用されたかが未知である場合，RL モデルによって解析が可能である.

RL モデルを仮定した攻撃法として，Bogdanov らは差分バイアス攻撃 (DBA (Differential Bias Attack)) を提案した [9]．Bogdanov らの先行手法を DBA-PM (Differential Bias Attack - Previous Method) と表記する．DBA は入力する選択平文組として以下の標本平文組を用いる (概要を図 3.2 に示す).

定義 3.2 (標本平文組)**.** 平文 $\mathbf{v}$ 及び副鍵 $\mathbf{k}_i$ で一意に定まる (他の副鍵に影響を受けない) 中間値全体を $\mathsf{G}(\mathbf{v}, \mathbf{k}_i)$ とする．$\mathbf{k}_i$ の候補を $\mathbf{k}_i^o$ とする．候補 $\mathbf{k}_i^o$ の標本平文組 $(\mathbf{v}^1, \mathbf{v}^2)$ とは，差分値 $\mathsf{G}(\mathbf{v}^1, \mathbf{k}_i^o) \oplus \mathsf{G}(\mathbf{v}^2, \mathbf{k}_i^o)$ をバイナリ値で表現したときに 0 となるビットの数が最大となる選択平文組をいう.

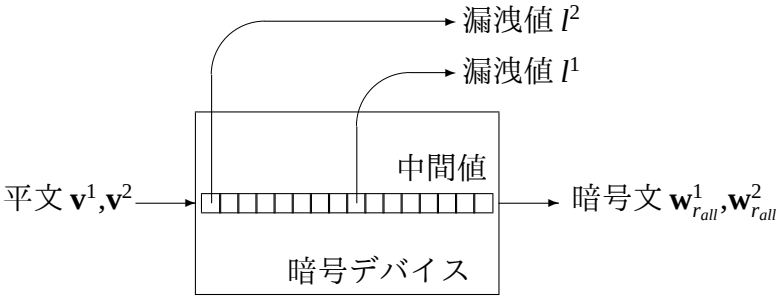


図 3.1　ランダム漏洩モデルの概要図

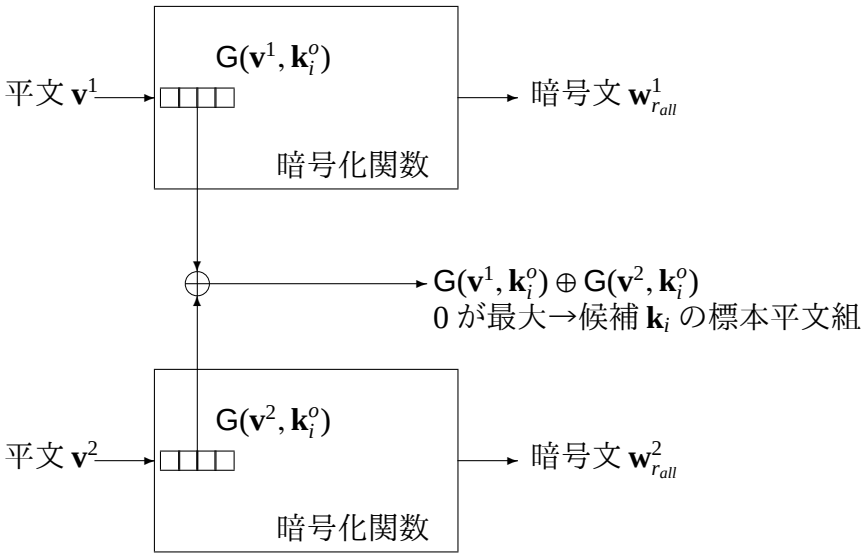


図 3.2　標本平文組の選択法

G の逆関数が既知ならば，攻撃者はこれを用いて定義を満たす $(\mathsf{G}(\mathbf{v}^1, \mathbf{k}_i^o), \mathsf{G}(\mathbf{v}^2, \mathbf{k}_i^o))$ から候補 $\mathbf{k}_i^o$ の標本平文組 $(\mathbf{v}^1, \mathbf{v}^2)$ を得る．本論文では攻撃者にとって G の逆関数が既知であると仮定する．攻撃者は，候補 $\mathbf{k}_i^o$ が正しいかを標本平文組 $(\mathbf{v}^1, \mathbf{v}^2)$ から得られた漏洩値の差分 $\Delta l = l^1 \oplus l^2$ が 0 となる確率を用いて確認する．$\mathbf{k}_i^o$ に対する Δl の確率分布を $Pr[\Delta l|\mathbf{k}_i^o]$ とする．攻撃者は確率分布を推定するため，複数の標本平文組を入力して $\Delta l = 0$ となる回数を数える．これを繰り返し，全ての $\mathbf{k}_i$ の候補に対して分布をそれぞれ推定する．定義 3.2 から，候補 $\mathbf{k}_i^o$ が正しい鍵である場合に $\mathsf{G}(\mathbf{v}^1, \mathbf{k}_i^o) \oplus \mathsf{G}(\mathbf{v}^2, \mathbf{k}_i^o)$ において 0 となるビットの数が最大となる．G で定

義された中間値の一部が漏洩値となる可能性があるので，$\mathbf{k}_i^o$ が正しい鍵である場合の $Pr[\Delta l = 0|\mathbf{k}_i^o]$ の推定値が他の候補よりも高くなる．

攻撃に必要な漏洩値の量をデータ量とよび q で表現する (付録 I 参照)．DBA の有効性を計算量 t 及びデータ量 q で評価する．Bogdanov らは DBA-PM を AES-128 (付録 D，図 D.3，[12]) に適用している．その結果，計算量は現実的に計算機で処理可能なレベル ($t \leq 2^{50}$) まで削減しているが，現実的に入手が困難な $q \geq 2^{34}$ というデータ量を必要とする．サイドチャネル攻撃を行う攻撃者に対する制約として，計算量よりもデータ量が重視される．平文長が 128 ビットの AES-128 において 2^{34} というデータ量を得るためには，少なくとも 256[GByte] の平文データの暗号化処理を行い毎回漏洩値を測定する必要がある．これはサイドチャネル攻撃においては現実的でないため，DBA は実行困難と結論付けられる．しかし，必要なデータ量を削減することで現実的な脅威となる可能性がある．そこで，本論文では与えられたデータ量が少ない攻撃者を仮定した攻撃手法である鍵列挙 [48] を用いた DBA-KE (Differential Bias Attack with Key Enumeration) 及び DBA-KE 対する耐性評価手法を提案する．

第 3.1 節で DBA-PM を，第 3.2 節で DBA-KE を示す．DBA-KE に対する耐性評価手法を第 3.3 節に示し，第 3.4 節で AES-128 に適用する．

3.1 差分バイアス攻撃の先行研究 [9]

Bogdanov らの先行研究では AES-128 ($r_{all} = 10$)[12] を対象としている [9]．定義 3.1 の RL モデルにおける $\mathsf{f}(\mathbf{v}, \mathbf{k})$ ($\mathsf{F}(\mathbf{v}, \mathbf{k})$) は平文 $\mathbf{v}$，鍵 $\mathbf{k}$，r 段目出力 $\mathbf{w}_r$ または r 段目で使用される段鍵 $\mathbf{rk}_r$ の 1 ビット (1 ワード) を出力する関数全体から乱択されると仮定する ($r \in \{1, 2, ..., 10\}$)．また，漏洩する中間値に制限を加えることで攻撃者にとって有利な仮定を行うことができる．図 3.3 及び 3.4 に例を示す．各セルは 1 ワード (8 ビット) を示し，漏洩する可能性がある値全体を示している．図 3.3 では灰色のワードで示されている 2 段目出力 $\mathbf{w}_2$ 及び段鍵 $\mathbf{rk}_2$ が漏洩する

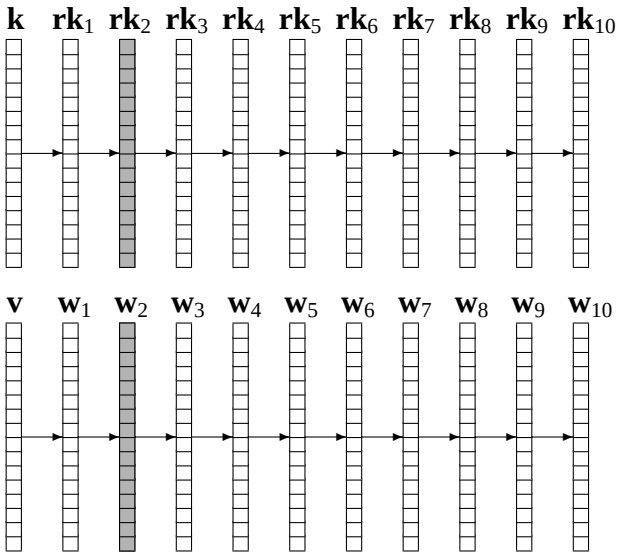


図 3.3　特定の段関数の出力のみが漏洩する例

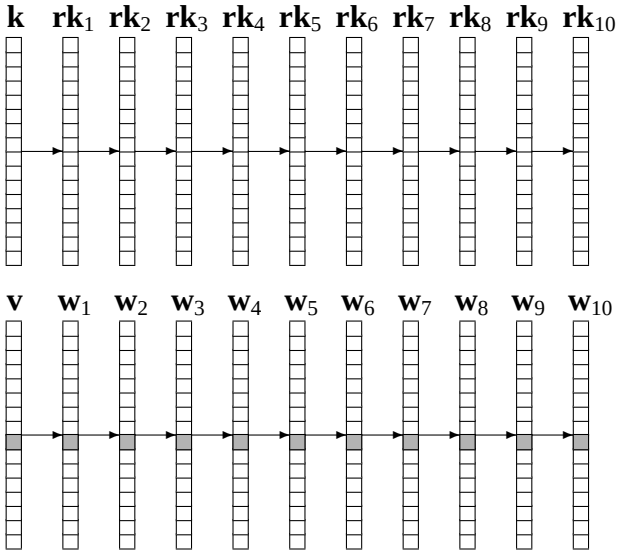


図 3.4　各段関数出力の特定のビットまたはワードの値が漏洩する例

と仮定している．また，図 3.4 では平文及び中間値の 9 番目のワード $\mathbf{v}_9$ 及び $\mathbf{w}_{r,9}$ $(r \in \{1, 2, ..., 10\})$ が漏洩すると仮定している．漏洩する中間値の段数と位置を漏洩領域とよぶ．一般的に，暗号化の初期段階の中間値や段鍵を得ることで鍵推定を効率的に行うことができる．このような中間値等が含まれる割合が大きい場合，攻撃者にとって有利な漏洩領域となる．

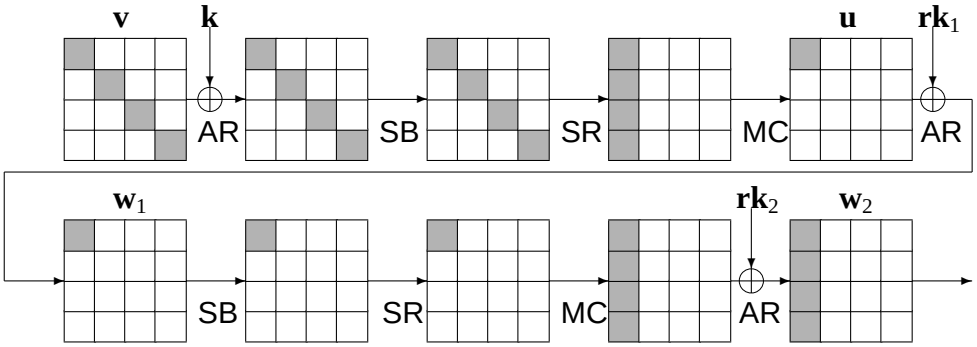


図 3.5　副鍵の推定が正しい場合の丸め差分特性.

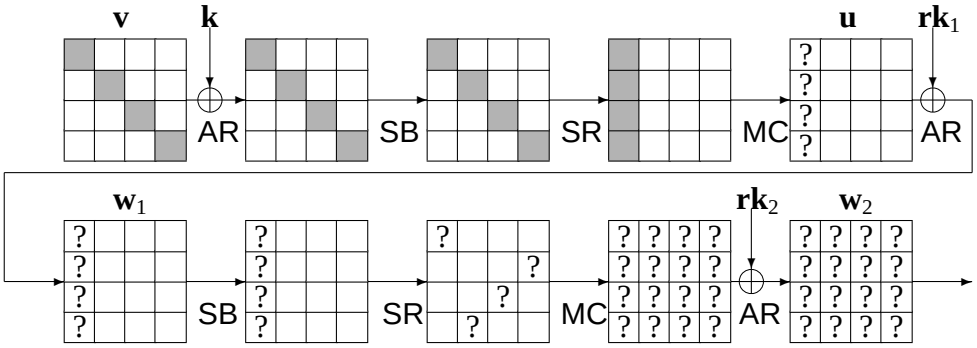


図 3.6　副鍵の推定が正くない場合の丸め差分特性

DBA-PM では，図 3.5 及び 3.6 に示す丸め差分特性を利用する．丸め差分特性は，選択平文組を入力した際の中間値の差分の状態に対し確定的に成立する．例えば，図 3.5 及び 3.6 は AES-128 の 2 段分の処理における中間値の差分の状態を示している．各セルは 1 ワード (8 ビット) を示し，灰色のセルは確定的に差分が 0 以外となるワード，白色のセルは確定的に差分が 0 となるワード及び「?」で示されたセルは差分が 0 となるか未知なワードを示す．1 段目の段関数内において，MixColumns の出力を $\mathbf{u}$ とする．攻撃者は副鍵 $(\mathbf{k}_1, \mathbf{k}_6, \mathbf{k}_{11}, \mathbf{k}_{16})$ の候補を選択し，この副鍵と平文で一意に定まる $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4)$ のうち 3 ワードの差分が 0 となる選択平文組を定義 3.2 に示した標本平文組とする (AES のワードの添字は付録 D の図 D.4 参照)．選択した候補が正しければ図 3.5 のような丸め差分特性が得られる．正しくない場

合，$(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4)$ の差分が未知となり図 3.6 のような特性となる．$Pr[\Delta l|\text{correct}]$ と $Pr[\Delta l|\text{wrong}]$ を選択した副鍵の候補が正しい場合 (correct) と正しくない場合 (wrong) の分布とする．同じ位置にある中間値が漏洩する場合とそうでない場合とがある．同じ位置から漏洩した場合，$Pr[\Delta l = 0|\text{correct}] > Pr[\Delta l = 0|\text{wrong}]$ が成立する．したがって，RL モデルにおいては一般的にこのような確率の偏りが存在する．もし，$Pr[\Delta l|\text{wrong}]$ よりも $Pr[\Delta l|\text{correct}]$ に近い分布を持つ鍵候補が得られたならば受理し，そうでない場合は非受理とする．

1 度に副鍵 32 ビットを推定するため，これを 4 回繰り返すことで 128 ビットの鍵 $\mathbf{k}$ 全体を推定することができる (分割統治法)．なお，以降は 32 ビットのワード及び副鍵を対象とするので，$\acute{\mathbf{v}}_i$，$\acute{\mathbf{k}}_i$ 及び $\acute{\mathbf{u}}_i$ を以下に定義する．

$$
\begin{aligned}
&\acute{\mathbf{v}}_1 = (\mathbf{v}_1, \mathbf{v}_6, \mathbf{v}_{11}, \mathbf{v}_{16}), \quad \acute{\mathbf{k}}_1 = (\mathbf{k}_1, \mathbf{k}_6, \mathbf{k}_{11}, \mathbf{k}_{16}), \quad \acute{\mathbf{u}}_1 = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4)\\
&\acute{\mathbf{v}}_2 = (\mathbf{v}_5, \mathbf{v}_{10}, \mathbf{v}_{15}, \mathbf{v}_4), \quad \acute{\mathbf{k}}_2 = (\mathbf{k}_5, \mathbf{k}_{10}, \mathbf{k}_{15}, \mathbf{k}_4), \quad \acute{\mathbf{u}}_2 = (\mathbf{u}_5, \mathbf{u}_6, \mathbf{u}_7, \mathbf{u}_8)\\
&\acute{\mathbf{v}}_3 = (\mathbf{v}_9, \mathbf{v}_{14}, \mathbf{v}_3, \mathbf{v}_8), \quad \acute{\mathbf{k}}_3 = (\mathbf{k}_9, \mathbf{k}_{14}, \mathbf{k}_3, \mathbf{k}_8), \quad \acute{\mathbf{u}}_3 = (\mathbf{u}_9, \mathbf{u}_{10}, \mathbf{u}_{11}, \mathbf{u}_{12})\\
&\acute{\mathbf{v}}_4 = (\mathbf{v}_{13}, \mathbf{v}_2, \mathbf{v}_7, \mathbf{v}_{12}), \quad \acute{\mathbf{k}}_4 = (\mathbf{k}_{13}, \mathbf{k}_2, \mathbf{k}_7, \mathbf{k}_{12}), \quad \acute{\mathbf{u}}_4 = (\mathbf{u}_{13}, \mathbf{u}_{14}, \mathbf{u}_{15}, \mathbf{u}_{16})
\end{aligned}
\tag{3.1}
$$

前述のように確率分布の偏りから鍵候補の受理／非受理を判断するが，DBA-PM では仮説検定を用いて判断している．そこで，仮説検定の実行に必要な標本平文組数 d を見積もる．d が小さいほど攻撃者は効率的に攻撃を行える．標本平文組数はその期待値を Kullback-Leibler 情報量 [29] を用いて見積もることができる [18]．Δl について確率分布を考えたときの Kullback-Leibler 情報量は以下のとおり．

$$
\mathsf{D}(Pr[\Delta l|\text{correct}]||Pr[\Delta l|\text{wrong}]) = \sum_{\Delta l} Pr[\Delta l|\text{correct}] \log_2 \frac{Pr[\Delta l|\text{correct}]}{Pr[\Delta l|\text{wrong}]} \tag{3.2}
$$

標本平文組数 d の期待値は第 1 種誤り率 α 及び第 2 種誤り率 β との間に

$$
\mathbb{E}[d] \geq \frac{\beta \cdot \log_2 \left(\frac{\beta}{1-\alpha}\right) + (1-\beta) \cdot \log_2 \left(\frac{1-\beta}{\alpha}\right)}{\mathsf{D}(Pr[\Delta l|\text{correct}]||Pr[\Delta l|\text{wrong}])} \tag{3.3}
$$

が成立する [18][32]．なお，α は正しい鍵であるのに非受理とする確率，β は正しい鍵でないのに受理する確率である．情報理論的に最適な検定を行える場合，式 (3.3) において等式が成立する．先行研究 [9] においては具体的な仮説検定の手法は明記されていないため，本論文では最適な検定のひとつである逐次確率比検定 [50] を用いる．

$Pr[\Delta l|\text{correct}]$ 及び $Pr[\Delta l|\text{wrong}]$ をビット単位及びワード単位 RL モデルにおいて形式的に求める方法を以下に示す．

ビット単位 RL モデルにおける Δl の形式的確率分布: n_{total}，n_{zero}，$n_{nonzero}$ をそれぞれ漏洩範囲に存在する全ビット数，差分が確定的に 0 であるワードに含まれるビット数及び確定的に 0 でないワードに含まれるビット数とする ($n_{total} = n_{zero} + n_{nonzero}$)．例えば，図 3.5 及び 3.6 の白色及灰色のワードに含まれるビット数はそれぞれ n_{zero} 及び $n_{nonzero}$ に含まれる．例えば，図 3.5 の 2 段目出力 $\mathbf{w}_2$ と段鍵 $\mathbf{rk}_2$ のみを漏洩範囲とする場合，$n_{total} = 256$，$n_{zero} = 224$ 及び $n_{nonzero} = 32$ となる．$\Delta l = 0$ となる形式的確率は以下のように求まる．

$$Pr[\Delta l = 0|\acute{\mathbf{k}}_i] = 1 \cdot \frac{n_{zero}}{(n_{total})^2} + \frac{127}{255} \cdot \frac{n_{nonzero}}{(n_{total})^2} + \frac{1}{2} \cdot \frac{(n_{total})^2 - n_{zero} - n_{nonzero}}{(n_{total})^2} \qquad (3.4)$$

この式が成立する理由を以下に示す．漏洩値の差分は全体で $(n_{total})^2$ 組存在する．漏洩値がいずれも同じ位置にある中間値であり，確定的に 0 であるワードからの漏洩値であるならば確率 1 で差分は 0 となる．また，差分が確定的に 0 でないワードからの漏洩値ならば，差分値は $(0,0,0,0,0,0,0,1)$ から $(1,1,1,1,1,1,1,1)$ までの 255 通りの値となる．その中で，特定のビットの差分が 0 となるものは，差分値全体の半数から $(0,0,0,0,0,0,0,0)$ を引いた 127 通り存在する．上記の 2 つの場合に当てはまらない差分値が 0 となる確率には偏りがないとみなし 1/2 とする．

n_{total}，n_{zero} 及び $n_{nonzero}$ を副鍵の推定が正しい場合とそうでない場合とで求め，$Pr[\Delta l = 0|\text{correct}]$ 及び $Pr[\Delta l = 0|\text{wrong}]$ を得る．なお，$Pr[\Delta l = 1|\text{correct}] =$

$1 - Pr[\Delta l = 0|\text{correct}]$ 及び $Pr[\Delta l = 1|\text{wrong}] = 1 - Pr[\Delta l = 0|\text{wrong}]$ となる．

ワード単位 RL モデルにおける Δl の形式的確率分布: m_{total}，m_{zero}，$m_{nonzero}$ をそれぞれ漏洩範囲に存在する全ワード数，確定的に差分が 0 であるワード数及び確定的に差分が 0 でないワード数とする ($m_{total} = m_{zero} + m_{nonzero}$)．例えば，図 3.5 の 2 段目出力 $\mathbf{w}_2$ と段鍵 $\mathbf{rk}_2$ のみを漏洩範囲とする場合，$m_{total} = 32$，$m_{zero} = 28$ 及び $m_{nonzero} = 4$ となる．$\Delta l = 0$ となる形式的確率は以下のように求まる．

$$Pr[\Delta l = 0|\acute{\mathbf{k}}_i] = 1 \cdot \frac{m_{zero}}{(m_{total})^2} + \frac{1}{256} \cdot \frac{(m_{total})^2 - m_{zero} - m_{nonzero}}{(m_{total})^2} \tag{3.5}$$

この式が成立する理由を以下に示す．ランダムな漏洩値の差分は全体で $(m_{total})^2$ 組存在する．漏洩値がいずれも同じ位置にある中間値であり，かつ，確定的に 0 であるワードの漏洩値であるならば確率 1 で差分は 0 となる．上記以外では一様分布を仮定し，1/256 の確率で 0 となる．

m_{total}，m_{zero} 及び $m_{nonzero}$ を副鍵の推定が正しい場合とそうでない場合とで求め，$Pr[\Delta l = 0|\text{correct}]$ 及び $Pr[\Delta l = 0|\text{wrong}]$ を得る．なお，Δl が 0 以外の値になる場合は $Pr[\Delta l \neq 0|\text{correct}] = 1 - Pr[\Delta l = 0|\text{correct}]$ 及び $Pr[\Delta l \neq 0|\text{wrong}] = 1 - Pr[\Delta l = 0|\text{wrong}]$ のように $\Delta l \neq 0$ である全値の周辺確率のみ考慮する．

$Pr[\Delta l|\text{correct}]$ 及び $Pr[\Delta l|\text{wrong}]$ を式 (3.2) に代入して Kullback-Leibler 情報量を求める．第 1 種誤り率 α 及び第 2 種誤り率 β を任意に設定し，式 (3.3) から d の期待値を得る．なお，先行研究では $\alpha = \beta = 2^{-32}$ としている．

逐次確率比検定を用いた DBA の AES-128 への適用について Algorithm 3.1 に示す．副鍵の候補 $\acute{\mathbf{k}}_i^o$ の尤度比 $\lambda_{\acute{\mathbf{k}}_i^o}$ は初期値として 1 を代入し，逐次的に更新する．尤度比の更新に標本平文組を用いる．差分が 1 ワード以外 0 である中間値 $(\mathbf{u}, \mathbf{u}')$ から段関数を逆算して得られる $(\mathbf{v}, \mathbf{v}')$ が $\acute{\mathbf{k}}_i^o$ の標本平文組となる．任意の選択平文 $\mathbf{v} \in \mathcal{V}$ に対し，それぞれ q_{same} 回の漏洩値を測定する．なお，RL モデルでは同様の

Algorithm 3.1 差分バイアス攻撃の先行手法 (DBA-PM) の AES-128 への適用

input $\mathcal{V}$ 及び $\mathcal{L}$.
for $i = 1 \rightarrow 4$ **do**
  for $\acute{\mathbf{k}}_i \in \mathbb{F}_2^{32}$ **do**
    尤度比を初期化 $\lambda_{\acute{\mathbf{k}}_i^o} \leftarrow 1$.
    while $(1-\alpha)/\beta < \lambda_{\acute{\mathbf{k}}_i^o} < \beta/(1-\alpha)$ **do**
      差分が 1 ワード以外 0 である $(\acute{\mathbf{u}}_i, \acute{\mathbf{u}}'_i)$ を乱択する.
      $(\acute{\mathbf{u}}_i, \acute{\mathbf{u}}'_i)$ 及び $\acute{\mathbf{k}}_i$ から $(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}'_i)$ を計算する.
      $(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}'_i)$ を含む選択平文組 $(\mathbf{v}, \mathbf{v}') \in \mathcal{V} \times \mathcal{V}$ を選ぶ.
      $(\mathbf{v}, \mathbf{v}')$ に対応する漏洩値 $\{l_j\}_{j=1}^{q_{same}}$ 及び $\{l'_{j'}\}_{j'=1}^{q_{same}}$ を取り出す.
      $\lambda_{\acute{\mathbf{k}}_i^o} \leftarrow \lambda_{\acute{\mathbf{k}}_i^o} \times \prod_{j,j'} \frac{Pr[l_j \oplus l'_{j'} | \text{correct}]}{Pr[l_j \oplus l'_{j'} | \text{wrong}]}$
    end while
    if $\beta/(1-\alpha) \leq \lambda_{\acute{\mathbf{k}}_i^o}$ **then**
      $\acute{\mathbf{k}}_i$ を副鍵の候補の集合 $\mathcal{T}_{\acute{\mathbf{k}}_i}$ に加える.
    end if
  end for
end for
残った候補 $\mathcal{T}_{\mathbf{k}} = \mathcal{T}_{\acute{\mathbf{k}}_1} \times \mathcal{T}_{\acute{\mathbf{k}}_2} \times \mathcal{T}_{\acute{\mathbf{k}}_3} \times \mathcal{T}_{\acute{\mathbf{k}}_4}$ から総当りで正しい鍵 $\mathbf{k}^\dagger$ を特定する.
return $\mathbf{k}^\dagger$

平文を暗号化したとしても，測定する値は毎回乱択されることに注意．標本平文組からそれぞれの平文に対応する q_{same} 個の漏洩値が得られるので，合計 $(q_{same})^2$ 組の標本平文組が得られる．これらの標本平文組に対し，尤度比 $\lambda_{\acute{\mathbf{k}}_i^o}$ を更新する．$Pr[l_j \oplus l'_{j'} = 0|\text{correct}] > Pr[l_j \oplus l'_{j'} = 0|\text{wrong}]$ が成立するので，候補 $\acute{\mathbf{k}}_i^o$ が正しい場合は尤度比が増加し，正しくない場合は減少する．尤度比 $\lambda_{\acute{\mathbf{k}}_i^o}$ が閾値 $\beta/(1-\alpha)$ よりも大きいならばこの候補は受理し，副鍵の候補の列 $\mathcal{T}_{\acute{\mathbf{k}}_i}$ に加える．また，閾値 $(1-\alpha)/\beta$ よりも小さければ非受理とする．残った副鍵の候補から鍵の候補の列 $\mathcal{T}_{\mathbf{k}} = \mathcal{T}_{\acute{\mathbf{k}}_1} \times \mathcal{T}_{\acute{\mathbf{k}}_2} \times \mathcal{T}_{\acute{\mathbf{k}}_3} \times \mathcal{T}_{\acute{\mathbf{k}}_4}$ を生成し，総当りで正しい鍵を探索する．なお，鍵の検証は平文と暗号文の組を用いて行う．$\alpha = \beta = 2^{-32}$ のとき，$1 + (2^{32} - 1) \cdot 2^{-32} \approx 2$ 個の候補のみが採択され，その中に正しい候補が存在する確率は $1 - 2^{-32} \approx 1$ である．したがって，仮説検定を行った後，鍵の候補の列 $\mathcal{T}_{\mathbf{k}}$ の要素数は 16 個となる．

DBA-PM を AES-128 に適用した際の計算量 t 及びデータ量 q は以下のとおり．

$$t = 4 \cdot 2^{32} \cdot \frac{\mathbb{E}[d]}{(q_{same})^2} \cdot \frac{q_{same}}{10} + (1 + 2^{-32} \cdot (2^{32} - 1))^4$$
$$\approx 2^{30.68} \cdot \frac{\mathbb{E}[d]}{q_{same}} \tag{3.6}$$

$$q = 4 \cdot 2^{32} \cdot q_{same} \tag{3.7}$$

なお，式 (3.6) の 1 行目の項 $q_{same}/10$ は AES が 10 段構成のため q_{same} 回の段関数処理を意味し，先行研究ではこれと $(q_{same})^2$ 回の XOR 処理 (Algorithm 3.1 では尤度の更新時に行う) の計算量が同等であると述べている．全副鍵の候補に対して尤度の更新を $\mathbb{E}[d]/(q_{same})^2$ 回行い，終了後に残った候補に対して総当りを行う．また，データ量については $\acute{\mathbf{v}}_i$ ($i \in \{1, 2, 3, 4\}$) を変数として $\mathbb{F}_2^{32}$ の任意の要素を入力する選択平文集合 (その他のワードは定数) を q_{same} 回ずつ入力して測定を行う．そのため，式 (3.7) のデータ量が必要となる．先行研究で使用された漏洩領域について，DBA-PM に必要な計算量及びデータ量を表 3.1 (ビット単位 RL モデル) 及び表 3.2 (ワード単位 RL モデル) に示す．$w_{r,i,j}$ (または $rk_{r,i,j}$) は r 段目出力の i 番目の 8 ビット中間値ワード (または副鍵) に含まれる j 番目のビットを，$\mathbf{w}_{r,i}$ (または $\mathbf{rk}_{r,i}$) は r 段目出力の i 番目の 8 ビット中間値ワード (または副鍵) を示す．

3.2 鍵列挙を用いた差分バイアス攻撃の提案

RL モデルにおいて，少ないデータ量で実行可能な DBA-KE を提案する．なお，鍵列挙はサイドチャネル攻撃全般における汎用的な手法であるため，付録 L 及び M で詳細を述べる．鍵列挙を用いることで鍵の候補を対数尤度順に列挙して鍵の候補の列を得る．この列は第 3.2.1 項で定義する最適な鍵候補の列となり，効率よく鍵の推定が可能となる．第 3.2.2 項で DBA-KE の攻撃手法を示す．その後，DBA-KE のデータ量と計算量の評価方法について第 3.2.3 項及び第 3.2.4 項に示す．

表 3.1　ビット単位ランダム漏洩モデルにおける差分バイアス攻撃の先行手法 (DBA-PM) の計算量及びデータ量

漏洩する中間値 $i \in [1:16], j \in [1:16]$		$\mathbb{E}[d]$	q_{same}	DBA-PM	
				t	q
段数	$w_{1,i,j}, rk_{1,i,j}$	$2^{28.84}$	256	$2^{51.84}$	$2^{42.00}$
	$w_{2,i,j}, rk_{2,i,j}$	$2^{24.84}$	256	$2^{47.84}$	$2^{42.00}$
	$w_{3,i,j}, rk_{3,i,j}$	$2^{39.98}$	256	$2^{62.98}$	$2^{42.00}$
中間値の位置 (i, j 固定)	$v_{i,j}, w_{r,i,j}$	$2^{19.90}$	11	$2^{47.44}$	$2^{37.46}$
	$v_{i,j}, w_{r,i,j}$ $(r \neq 9)$	$2^{19.36}$	10	$2^{47.04}$	$2^{37.32}$
	$v_{i,j}, w_{r,i,j}$ $(r \neq 1,2,8,9)$	$2^{33.32}$	7	$2^{60.41}$	$2^{36.81}$
段数及び中間値の位置	$v_{i,j}, w_{r,i,j}, rk_{r,i,j}$	$2^{38.04}$	256·11	$2^{57.58}$	$2^{45.46}$
	$v_{i,j}, w_{r,i,j}, rk_{r,i,j}$ $(r \neq 9)$	$2^{37.49}$	256·10	$2^{57.17}$	$2^{45.32}$
	$v_{i,j}, w_{r,i,j}, rk_{r,i,j}$ $(r \neq 1,2,8,9)$	$2^{51.22}$	256·7	$2^{71.41}$	$2^{44.81}$

表 3.2　ワード単位ランダム漏洩モデルにおける差分バイアス攻撃の先行手法 (DBA-PM) の計算量及びデータ量

漏洩する中間値 $i \in [1:16]$		$\mathbb{E}[d]$	q_{same}	DBA-PM	
				t	q
段数	$\mathbf{w}_{1,i}, \mathbf{rk}_{1,i}$	$2^{19.84}$	32	$2^{45.84}$	$2^{39.00}$
	$\mathbf{w}_{2,i}, , \mathbf{rk}_{2,i}$	$2^{14.48}$	32	$2^{40.48}$	$2^{39.00}$
	$\mathbf{w}_{3,i}, , \mathbf{rk}_{3,i}$	$2^{29.64}$	32	$2^{55.64}$	$2^{42.00}$
中間値の位置 (i 固定)	$\mathbf{v}_i, \mathbf{w}_{r,i}$	$2^{15.62}$	11	$2^{43.16}$	$2^{37.46}$
	$\mathbf{v}_i, \mathbf{w}_{r,i}$ $(r \neq 9)$	$2^{15.52}$	10	$2^{43.19}$	$2^{37.32}$
	$\mathbf{v}_i, \mathbf{w}_{r,i}$ $(r \neq 1,2,8,9)$	$2^{24.22}$	7	$2^{52.41}$	$2^{36.81}$
段数及び中間値の位置	$\mathbf{v}_i, \mathbf{w}_{r,i}, \mathbf{rk}_{r,i}$	$2^{24.04}$	16×11	$2^{46.58}$	$2^{42.45}$
	$\mathbf{v}_i, \mathbf{w}_{r,i}, \mathbf{rk}_{r,i}$ $(r \neq 9)$	$2^{23.58}$	16×10	$2^{46.26}$	$2^{42.32}$
	$\mathbf{v}_i, \mathbf{w}_{r,i}, \mathbf{rk}_{r,i}$ $(r \neq 1,2,8,9)$	$2^{38.04}$	16×7	$2^{61.23}$	$2^{41.80}$

3.2.1 最適な鍵候補の列

$\mathcal{T}_{\mathbf{k}} = \{\mathbf{k}^1, \mathbf{k}^2, ..., \mathbf{k}^{t_k}\}$ を漏洩値の測定によって生成した鍵候補の列とする ($t_k \leq 2^{n_{key}}$). 攻撃者は $\mathbf{k}^1$ から $\mathbf{k}^{t_k}$ の順番に検証を行い，$\mathcal{T}_{\mathbf{k}}$ に含まれない候補については検証を行わない．なお，t_k は列挙して検証を行う鍵の候補数であり，攻撃者の計算能力に基づいて決定される．実装の容易さを考え，$\mathcal{T}_{\mathbf{k}}$ を生成する基準として対数尤度を用いる．これは，尤度が実数の乗算で計算されるのに対し，対数尤度は加算で済むため計算量的に優れているからである．最適な列を以下に定義する．

定義 3.3 (最適な列)**.** 攻撃者が漏洩値 $\mathcal{L}$ から，鍵 $\mathbf{k}$ の候補の列 $\mathcal{T}_{\mathbf{k}} = \{\mathbf{k}^1, \mathbf{k}^2, ..., \mathbf{k}^{t_k}\}$ を得たとする．鍵候補の列 $\mathcal{T}_{\mathbf{k}}$ が以下を満たすとき，$\mathcal{T}_{\mathbf{k}}$ は最適な列である．

(i) 任意の $o < o'$ なる $\mathbf{k}^o, \mathbf{k}^{o'} \in \mathcal{T}_{\mathbf{k}}$ に対し以下が成立する．

$$\log_2(Pr[\mathcal{L}|\mathbf{k}^o]) \geq \log_2(Pr[\mathcal{L}|\mathbf{k}^{o'}]) \tag{3.8}$$

(ii) 任意の $\mathbf{k}_l^o \in \mathcal{T}_{\mathbf{k}}$ 及び $\mathbf{k}^{o''} \notin \mathcal{T}_{\mathbf{k}}$ に対し以下が成立する．

$$\log_2(Pr[\mathcal{L}|\mathbf{k}^o]) \geq \log_2(Pr[\mathcal{L}|\mathbf{k}^{o''}]) \tag{3.9}$$

DBA-KE においては，分割統治法により 32 ビットの副鍵について最適な列を生成する．なお，副鍵の候補の列を副列とよぶ．副列から鍵列挙を用いて 128 ビットの鍵について最適な列を生成する．

3.2.2 攻撃の手順

準備として関数 $\mathsf{I}_{\mathsf{DSM}_i}$ を定義する．まず，AES の段関数における AddRoundkey, SubBytes 及び MixColumns の一部から構成される $\mathbb{F}_2^{64} \rightarrow \mathbb{F}_2^{32}$ なる関数 SM_i とする (定義 3.2 の関数 G に相当)．関数 SM_i の添字 i は式 (3.1) に対応しており，例えば SM_1 は以下のとおり．

$$\begin{aligned}\acute{\mathbf{u}}_1 &= \mathsf{SM}_1(\acute{\mathbf{v}}_1, \acute{\mathbf{k}}_1) \\ &= \mathsf{MDS}(\mathsf{S}(\mathbf{v}_1 \oplus \mathbf{k}_1), \mathsf{S}(\mathbf{v}_6 \oplus \mathbf{k}_6), \mathsf{S}(\mathbf{v}_{11} \oplus \mathbf{k}_{11}), \mathsf{S}(\mathbf{v}_{16} \oplus \mathbf{k}_{16}))\end{aligned} \tag{3.10}$$

次に，関数 SM_i を並列に処理する以下の関数 $(\mathbb{F}_2^{96} \rightarrow \mathbb{F}_2^{64})$ を定義する．

$$\begin{aligned}(\acute{\mathbf{u}}_i, \acute{\mathbf{u}}'_i) &= \mathsf{DSM}_i(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}'_i, \acute{\mathbf{k}}_i) \\ &= (\mathsf{SM}_i(\acute{\mathbf{v}}_i, \acute{\mathbf{k}}_i), \mathsf{SM}_i(\acute{\mathbf{v}}'_i, \acute{\mathbf{k}}_i)) \end{aligned} \tag{3.11}$$

関数 $\mathsf{I}_{\mathsf{DSM}_i}$ は差分値 $\acute{\mathbf{u}}_i \oplus \acute{\mathbf{u}}'_i$ の 1 ワードを除く全てのワードが確定的に差分が 0 となる場合に 1 を，そうでない場合は 0 を出力する．副鍵の候補 $\acute{\mathbf{k}}_i^o$ と選択平文組 $(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}'_i)$ に対し関数 $\mathsf{I}_{\mathsf{DSM}_i}$ を計算すると，この平文組が $\acute{\mathbf{k}}_i^o$ の標本平文組か確認できる．

Algorithm 3.2 に DBA-KE の AES-128 への適用を示す．KeyEnum は最適な鍵列挙 [48] を使用する (付録 M 参照)．入力 $\mathcal{U}_{\acute{\mathbf{k}}_i}$ は鍵候補とその対数尤度の組 $\left(\acute{\mathbf{k}}_i^o, \lambda_{\acute{\mathbf{k}}_i^o}\right)$ を要素とする列である．まず，鍵列挙の最適な副列 $\mathcal{U}_{\acute{\mathbf{k}}_i}$ を生成する．ある副鍵の候補 $\acute{\mathbf{k}}_i^o$ の漏洩値に対する尤度は以下に求まる．

$$\begin{aligned} Pr[\mathcal{L} \times \mathcal{L} | \acute{\mathbf{k}}_i^o] &= \prod_{(l,l')} Pr[l \oplus l' | \acute{\mathbf{k}}_i^o] \\ &= \left(\prod_{(l,l') | \mathsf{I}_{\mathsf{DSM}_i}(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}'_i, \acute{\mathbf{k}}_i^o)=0} Pr[l \oplus l' | \text{wrong}] \right) \cdot \left(\prod_{(l,l') | \mathsf{I}_{\mathsf{DSM}_i}(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}'_i, \acute{\mathbf{k}}_i^o)=1} Pr[l \oplus l' | \text{correct}] \right) \\ &= \left(\prod_{(l,l')} Pr[l \oplus l' | \text{wrong}] \right) \cdot \left(\prod_{(l,l') | \mathsf{I}_{\mathsf{DSM}_i}(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}'_i, \acute{\mathbf{k}}_i^o)=1} \frac{Pr[l \oplus l' | \text{correct}]}{Pr[l \oplus l' | \text{wrong}]} \right) \end{aligned} \tag{3.12}$$

選択平文組数が定数であるならば，$\prod_{(l,l')} Pr[l \oplus l' | \text{wrong}]$ は定数となる．したがって，対数尤度は

$$\begin{aligned} \lambda_{\acute{\mathbf{k}}_i^o} &= \log_2 \left(Pr[\mathcal{L} \times \mathcal{L} | \acute{\mathbf{k}}_i^o] \right) \\ &= \sum_{(l,l') | \mathsf{I}_{\mathsf{DSM}_i}(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}'_i, \acute{\mathbf{k}}_i^o)=1} \log_2 \left(\frac{Pr[l \oplus l' | \text{correct}]}{Pr[l \oplus l' | \text{wrong}]} \right) + const, \end{aligned} \tag{3.13}$$

となる．なお，*const* は鍵候補の順番に影響を与えないので鍵列挙の際は考慮する必要がない．

Algorithm 3.2 鍵列挙を用いた差分バイアス攻撃 (DBA-KE) の AES-128 への適用

input $\mathcal{V}$ and $\mathcal{L}$.
for $i = 1 \rightarrow 4$ **do**
  for $\acute{\mathbf{k}}_i^o \in \mathbb{F}_2^{32}$ **do**
    対数尤度比の初期化 $\lambda_{\acute{\mathbf{k}}_i^o} \leftarrow 0$.
    for $(\mathbf{v}, \mathbf{v}') \in \mathcal{V} \times \mathcal{V}$ **do**
      if $\mathsf{I}_{\mathsf{DSM}_i}(\acute{\mathbf{v}}_i, \mathbf{v}'_i, \acute{\mathbf{k}}_i^o) = 1$ **then**
        $(\mathbf{v}, \mathbf{v}')$ に対応する $\{l_j\}_{j=1}^{q_{same}}$ 及び $\{l'_{j'}\}_{j'=1}^{q_{same}}$ を取り出す.
        対数尤度の更新 $\lambda_{\acute{\mathbf{k}}_i^o} \leftarrow \lambda_{\acute{\mathbf{k}}_i^o} + \sum_{i,j} \log_2 \left(\frac{Pr[l_i \oplus l'_j | \text{correct}]}{Pr[l_i \oplus l'_j | \text{wrong}]} \right)$.
      end if
    end for
  end for
  任意の $\left(\acute{\mathbf{k}}_i^o, \lambda_{\acute{\mathbf{k}}_i^o}\right)$ を $\lambda_{\acute{\mathbf{k}}_i^o}$ について降順に並べて $\mathcal{U}_{\acute{\mathbf{k}}_i}$ に保存する.
end for
$\mathbf{k}^\dagger \leftarrow \mathsf{keyEnum}(\mathcal{U}_{\acute{\mathbf{k}}_1}, \mathcal{U}_{\acute{\mathbf{k}}_2}, \mathcal{U}_{\acute{\mathbf{k}}_3}, \mathcal{U}_{\acute{\mathbf{k}}_4}, t_k)$
return $\mathbf{k}^\dagger$ or "失敗"

Algorithm 3.2 においては，ある選択平文組が標本平文組となるかを $\mathsf{I}_{\mathsf{DSM}_i}$ で確認する．$\mathsf{I}_{\mathsf{DSM}_i}$ が 1 を出力するならば，対応する漏洩値を取り出して対数尤度 $\lambda_{\acute{\mathbf{k}}_i^o}$ を更新する．副鍵 $\acute{\mathbf{k}}_i$ の全候補の対数尤度を求めた後，$(\acute{\mathbf{k}}_i^o, \lambda_{\acute{\mathbf{k}}_i^o})$ を $\lambda_{\acute{\mathbf{k}}_i}$ について降順に並べて最適な副列 $\mathcal{U}_{\acute{\mathbf{k}}_i}$ を生成する．

上記で得られた副列から鍵列挙を行い鍵候補の列を最適な列として生成し，$\mathbf{k}^1$ から $\mathbf{k}^{t_k}$ の順に正しい鍵 $\mathbf{k}^\dagger$ を探索する．t_k 番目以降の鍵候補については検証を行わないので，t_k 回の検証で正しい鍵が見つからない場合攻撃は失敗する．

3.2.3 全単射の特性を用いたデータ量の削減

DBA-PM では $\acute{\mathbf{v}}_i$ $(i \in \{1, 2, 3, 4\})$ を変数として $\mathbb{F}_2^{32}$ の任意の要素を入力する選択平文集合を考え，それぞれの平文に対して q_{same} 個ずつ漏洩値を測定している．これに対し，DBA-KE では保存されたデータから乱択した選択平文組が特定の候補の標本平文組かを確認する．これにより DBA-PM よりも少ないデータ量で攻撃を行うことができる．なお，選択平文組は副鍵に対応した 32 ビット (4 ワード) の

み乱択し，それ以外は一定にする．

副鍵の候補 $\acute{\mathbf{k}}_i^o$ の標本平文組数 d を得るのに必要なデータ量 q を見積もる．その際，DSM_i (式 (3.10) 及び (3.11) 参照) において全単射の特性を利用する．関数 SM_i は $\acute{\mathbf{k}}_i$ を定数とすると $\mathbb{F}_2^{32} \to \mathbb{F}_2^{32}$ なる全単射となるので，DSM_i についても $\mathbb{F}_2^{64} \to \mathbb{F}_2^{64}$ なる全単射となる．

まず，選択平文組全体の中で標本平文組となるものの割合を見積もる (概要を図 3.7 に示す)．まず，1 ワードを除く全ワードの差分が 0 となる中間値 $\acute{\mathbf{u}}_i$ の組数を計算する．差分が 0 でない 1 ワードの組は $\binom{2^8}{2}$ 組存在し，差分が 0 である 3 ワードの組は 2^{24} 組存在する．また，差分が 0 でない 1 ワードは 4 ワードの中から選ばれる．したがって，目的とする $\acute{\mathbf{u}}_i$ の組数は全部で $4 \cdot 2^{24} \cdot \binom{2^8}{2}$ となる．

次に，$\mathsf{I}_{\mathsf{DSM}_i}(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}_i', \acute{\mathbf{k}}_i) = 1$ なる平文 4 ワード $\acute{\mathbf{v}}_i$ の組数を考える．関数 DSM_i における全単射の特性から，任意の $(\acute{\mathbf{u}}_i, \acute{\mathbf{u}}_i')$ に対して，対応する $(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}_i')$ が存在する．つまり，$\mathsf{I}_{\mathsf{DSM}_i}(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}_i', \acute{\mathbf{k}}_i) = 1$ となる $\acute{\mathbf{v}}_i$ の組数は 1 ワードを除く全ワードの差分が 0 となる $\acute{\mathbf{u}}_i$ の組数と同じである．したがって，求めたい割合は

$$\frac{4 \cdot 2^{24} \cdot \binom{2^8}{2}}{\binom{2^{32}}{2}} \approx 2^{-22}. \tag{3.14}$$

となる．用意した平文データ $\mathcal{V}$ に含まれる値が異なる選択平文の個数を q_{dif} とする．式 (3.14) より，q_{dif} 個のデータからは $\binom{q_{dif}}{2}$ 組の選択平文組が得られる．このうち，標本平文組として使えるのは $\binom{q_{dif}}{2} \cdot 2^{-22}$ 組と見積もられる．したがって，d 組の標本平文組を得るには，以下の異なる選択平文が必要となる．

$$q_{dif} \approx 2^{11} \cdot \frac{(d)^{\frac{1}{2}}}{q_{same}} \tag{3.15}$$

なお，異なる平文からそれぞれ q_{same} 個の漏洩値を得るので，1 個の副鍵の推定に必要なデータ量は $q_{dif} \cdot q_{same}$ となる．また，副鍵は 4 個存在するのでデータ量は

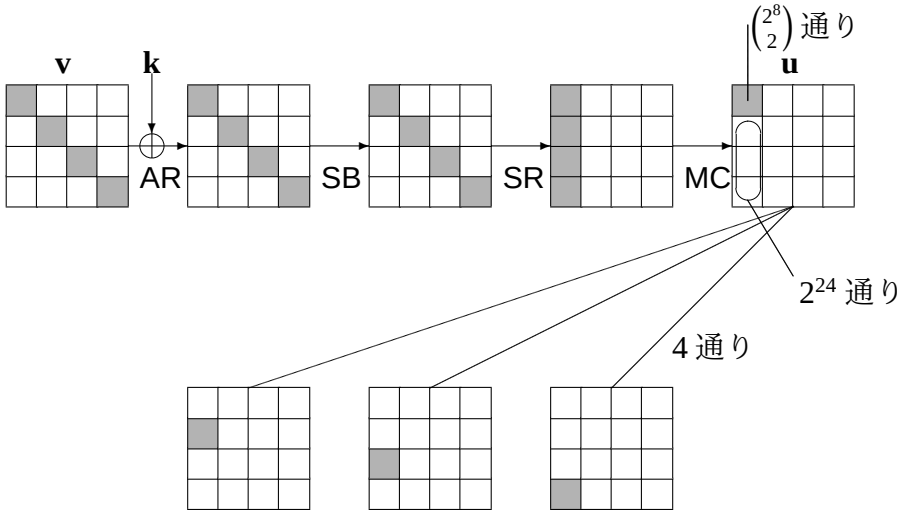


図 3.7　標本平文組の割合を導出する方法の概要

以下のようになる．

$$\begin{aligned} q &= 2^{11} \cdot \frac{(d)^{\frac{1}{2}}}{q_{same}} \cdot q_{same} \cdot 4. \\ &= 2^{13} \cdot (d)^{\frac{1}{2}} \end{aligned} \tag{3.16}$$

これは DBA-KE のデータ量 (式 (3.7) 参照) よりも大幅に少ないことが分かる．

3.2.4　計算量の評価

DBA-KE の計算量は以下のように見積もられる．

$$t = t_{prep} + t_{ke} + t_k \tag{3.17}$$

ここで，t_{prep} 及び t_{ke} はそれぞれ副列の生成と鍵列挙に必要な計算量を示している．式 (3.15) から，副列の生成に必要な計算量は

$$\begin{aligned} t_{prep} &= d \cdot 2^{22} \cdot \frac{2}{10} \cdot 2^{32} \cdot 4 \\ &\approx 2^{30.68} \cdot (q)_{dif}^2 \end{aligned} \tag{3.18}$$

と求まる．なお，2/10 は DSM_i の計算量である (2 段分の暗号化処理に相当).

次に鍵列挙の計算量 t_{ke} について考察する．もし，最適な鍵列挙 [48] を使用した場合，検証する鍵候補数 t_k に対して t_{ke} が増加することが分かっている [10]．これにより，全体計算量 t の制約から t_k を減らす必要が生じ，攻撃成功率が減少する．そのため，t_{ke} が抑えられる準最適な鍵列挙 [10][13][33][34][37] を実用上は使用する必要がある．

第 3.3 節において述べる耐性評価手法においては，計算量の下界である t_k を用いる．任意の鍵列挙の計算量は t_k 以上であるので，攻撃者にとって t_k 回の暗号化処理が実行困難であれば，DBA-KE も困難となる．

3.3 鍵列挙を用いた差分バイアス攻撃に対する耐性評価手法の提案

RL モデルを仮定し，DBA-KE に対する耐性評価手法を提案する．耐性評価の結果として，DBA-KE が実行困難になるデータ量を見積もる．その際，攻撃者は $\mathsf{A}_{t\leq t_{adv}}$ を仮定する．提案する評価手法では以下の 2 つの評価が存在する．

(i) 情報理論的評価 (第 3.3.1 項): 計算量の期待値を情報理論的に導出する．その際，推測エントロピーを用いる．DBA-KE において推測エントロピーは導出が困難であるが，その下界 [35] はわずかな計算量で求められる．そのため，推測エントロピーの下界を求め，これが攻撃者の計算能力 t_{adv} を超えるデータ量を求める．

(ii) ランク評価を用いた実験的評価 (第 3.3.2 項): 攻撃成功率をランク評価を用いて実験的に求める．なお，ヒストグラムを用いたランク評価 [20] を用いる (付録 N 参照)．ランク評価を複数回実行することで攻撃成功率が得られ，これが 0 となるデータ量を求める．

ランク評価を用いた実験的評価は計算量が大きいため，実験できるデータ量等に制約がある．これに対し，情報理論的評価はわずかな計算量で実行できるが基準

が厳密ではない．ランク評価を用いた実験的評価が実行できない場合は，情報理論的評価で攻撃が実行困難になるデータ量を見積もる必要がある．

3.3.1 情報理論的評価

DBA-KE の計算量を理論的に見積もる手法として，以下の推測エントロピーの下界を導出する [2][35].

定義 3.4 (推測エントロピー [28][35]). 攻撃者が漏洩値 $\mathcal{L}$ から，鍵 $\mathbf{k}$ の候補の列 $\mathcal{T}_{\mathbf{k}} = \{\mathbf{k}^1, \mathbf{k}^2, ..., \mathbf{k}^{2^{n_{key}}}\}$ を得たとする．推測エントロピーは

$$G = \mathop{\mathbb{E}}_{\mathcal{L}} \mathop{\mathbb{E}}_{\mathbf{k}^\dagger}[\mathsf{I}(\mathcal{T}_{\mathbf{k}})] \tag{3.19}$$

で定義される．ここで，$\mathsf{I}(\mathcal{T}_{\mathbf{k}})$ は正しい鍵の候補 $\mathbf{k}^o(\mathbf{k}^o = \mathbf{k}^\dagger)$ の添字 o を出力する．

推測エントロピーは鍵列挙を行った攻撃者が正しい鍵を特定するまでに検証を行う鍵の候補数の漏洩値に対する期待値を示している．したがって，推測エントロピー G は

$$G = \sum_{\mathcal{L}} Pr[\mathcal{L}] \sum_{o=1}^{2^{n_{key}}} o \cdot Pr[\mathbf{k}^o|\mathcal{L}]. \tag{3.20}$$

から得られる．$Pr[\mathbf{k}|\mathcal{L}]$ は漏洩値に対する事後確率分布であり，攻撃者にとって有利な漏洩値に対しては $\mathbf{k}^1$ においてピークがある分布に，そうでない場合は一様分布に近づく．これに鍵候補の添字 o との積の総和を計算することで，$\mathcal{L}$ を与えられた攻撃者が鍵検証を行う回数の期待値となる．式 (3.20) を計算するには，全鍵候補の事後確率を計算する必要があり，一般的な鍵長のブロック暗号では式 (3.20) の計算は困難である．

これに対し，推測エントロピーの下界 ($G \geq LG$) は以下で求まることが知られている [2].

$$LG = \frac{1}{1 + n_{key}} \sum_{\mathcal{L}} \left(\sum_{\mathbf{k}^\dagger} Pr[\mathcal{L}, \mathbf{k}^\dagger]^{\frac{1}{2}} \right)^2 \tag{3.21}$$

なお，$Pr[\mathcal{L}, \mathbf{k}^{\dagger}]$ は正しい鍵が $\mathbf{k}^{\dagger}$ であり，そのときに漏洩値 $\mathcal{L}$ が得られる同時確率である．LG の計算を式 (3.21) で行うことは困難であるが，以下の命題から DBA-KE においては LG をわずかな計算量で求めることができる．

命題 3.1. AES-128 に対して DBA-KE を適用したとき，各漏洩値が互いに独立であり，かつ，攻撃者は全ての副鍵の候補に対し d 組の標本平文組に対応した漏洩値を得られると仮定する．Δl の形式的確率分布 $Pr[\Delta l|\text{correct}]$ 及び $Pr[\Delta l|\text{wrong}]$ に対する推測エントロピーの下界は以下の式で定数時間で求めることができる．

$$LG = \frac{1}{129}\left(1 + \left(2^{32} - 1\right) \cdot \left(Pr[\Delta l = 0|\text{correct}]^{\frac{1}{2}} \cdot Pr[\Delta l = 0|\text{wrong}]^{\frac{1}{2}} + Pr[\Delta l \neq 0|\text{correct}]^{\frac{1}{2}} \cdot Pr[\Delta l \neq 0|\text{wrong}]^{\frac{1}{2}}\right)^{d}\right)^{4} \tag{3.22}$$

証明． $\mathcal{V}_i$ 及び $\mathcal{L}_i$ を $\mathcal{V}$ 及び $\mathcal{L}$ の部分集合とし，副鍵 $\mathbf{k}_i$ の候補の対数尤度比の計算に使用される ($i \in \{1, 2, 3, 4\}$)．例えば，$\mathcal{V}_i$ は $(\mathbf{v}_1, \mathbf{v}_6, \mathbf{v}_{11}, \mathbf{v}_{16})$ の 32 ビットのみ変数ビットで，その他は定数ビットとなる選択平文集合である．式 (3.21) を簡略化するため，分割統治法を行う．

$$\begin{aligned} LG &= \frac{1}{129} \sum_{\mathcal{L}} \left(\prod_{i=1}^{4} \sum_{\acute{\mathbf{k}}_i^{\dagger}} Pr[\mathcal{L} \times \mathcal{L}|\acute{\mathbf{k}}_i^{\dagger}]^{\frac{1}{2}} \cdot Pr[\acute{\mathbf{k}}_i^{\dagger}]^{\frac{1}{2}} \right)^2 \\ &= \frac{1}{129} \cdot \prod_{i=1}^{4} \frac{1}{2^{32}} \sum_{\mathcal{L}_i} \left(\sum_{\acute{\mathbf{k}}_i^{\dagger}} Pr[\mathcal{L}_i \times \mathcal{L}_i|\acute{\mathbf{k}}_i^{\dagger}]^{\frac{1}{2}} \right)^2 \\ &= \frac{1}{129} \cdot \left(\frac{1}{2^{32}} \sum_{\mathcal{L}_i} \left(\sum_{\acute{\mathbf{k}}_i^{\dagger}} Pr[\mathcal{L}_i \times \mathcal{L}_i|\acute{\mathbf{k}}_i^{\dagger}]^{\frac{1}{2}} \right)^2 \right)^4 \end{aligned} \tag{3.23}$$

$\mathcal{L}_i$ に含まれる漏洩値の差分が 0 となる組数で尤度が求まることを利用するため，$\mathcal{V}_i$ に対応する $\mathcal{L}_i$ を $\mathbf{d} = (d^1, d^2, ..., d^{2^{32}})$ で表現する．ここで，$d^o \in \{0, 1, ..., d\}$ は副鍵の候補 $\acute{\mathbf{k}}_i^o$ ($o \in \{1, 2, ..., 2^{32}]$) の標本平文組に対応する漏洩値の組 $(l, l') \in \mathcal{L} \times \mathcal{L}$ の中で差分が 0 である組数とする．また，尤度比の総和の 2 乗部分について展開

すると，添字 o 及び o' に対する 2 重の総和に置き換えられる．

$$
\begin{aligned}
LG &= \frac{1}{129}\cdot\left(\frac{1}{2^{32}}\sum_{\mathbf{d}}\left(\left(\prod_{o=1}^{2^{32}}\binom{d}{d^o}\right)\cdot\left(\sum_{\acute{\mathbf{k}}_i^{\dagger}} Pr[\mathbf{d}|\acute{\mathbf{k}}_i^{\dagger}]^{\frac{1}{2}}\right)^2\right)\right)^4 \\
&= \frac{1}{129}\cdot\left(\frac{1}{2^{32}}\sum_{\mathbf{d}}\left(\left(\prod_{o=1}^{2^{32}}\binom{d}{d^o}\right)\sum_{o'=1}^{2^{32}}\sum_{o''=1}^{2^{32}} Pr[\mathbf{d}|\acute{\mathbf{k}}_i^{o'}]^{\frac{1}{2}}\cdot Pr[\mathbf{d}|\acute{\mathbf{k}}_i^{o''}]^{\frac{1}{2}}\right)\right)^4 \\
&= \frac{1}{129}\cdot\left(\frac{1}{2^{32}}\sum_{o'=1}^{2^{32}}\sum_{o''=1}^{2^{32}}\sum_{\mathbf{d}}\left(\left(\prod_{o=1}^{2^{32}}\binom{d}{d^o}\right)\cdot Pr[\mathbf{d}|\acute{\mathbf{k}}_i^{o'}]^{\frac{1}{2}}\cdot Pr[\mathbf{d}|\acute{\mathbf{k}}_i^{o''}]^{\frac{1}{2}}\right)\right)^4 \qquad (3.24)
\end{aligned}
$$

ここで，o' 及び o'' が一致する場合としない場合で式を分割する．$o' \neq o''$ の場合，$\mathbf{k}_i^{o'}$ または $\mathbf{k}_i^{o''}$ のいずれかが正しい副鍵である．そのため，$\mathbf{k}_i^{o'}$ が正しい場合と $\mathbf{k}_i^{o''}$ が正しい場合で 2 通り存在する．2^{32} 通りの o に対して $o = o'$ のときは副鍵の推定が正しいため $Pr[d^{o'}|\text{correct}]$ を用い，その他は $Pr[d^{o'}|\text{wrong}]$ を用いる．

$$
\begin{aligned}
LG &= \frac{1}{129}\cdot\left(\frac{2^{32}}{2^{32}}\sum_{\mathbf{d}}\left(\left(\prod_{o=1}^{2^{32}}\binom{d}{d^o}\right)\cdot Pr[\mathbf{d}|\acute{\mathbf{k}}_i^{o'}]\right)+\frac{\binom{2^{32}}{2}\cdot 2}{2^{32}}\sum_{\mathbf{d}}\left(\left(\prod_{o=1}^{2^{32}}\binom{d}{d^o}\right)\cdot Pr[\mathbf{d}|\acute{\mathbf{k}}_i^{o'}]^{\frac{1}{2}}\cdot Pr[\mathbf{d}|\acute{\mathbf{k}}_i^{o''}]^{\frac{1}{2}}\right)\right)^4 \\
&= \frac{1}{129}\cdot\left(\sum_{\mathbf{d}}\left(\left(\prod_{o=1}^{2^{32}}\binom{d}{d^o}\right)\cdot\left(\prod_{\substack{o=1\\ o\neq o'}}^{2^{32}} Pr[d^o|\text{wrong}]\cdot Pr[d^{o'}|\text{correct}]\right)\right)\right. \\
&\left.+(2^{32}-1)\sum_{\mathbf{d}}\left(\left(\prod_{o=1}^{2^{32}}\binom{d}{d^o}\right)\cdot\left(\prod_{\substack{o=1\\ o\neq o'}}^{2^{32}} Pr[d^o|\text{wrong}]^{\frac{1}{2}}\cdot Pr[d^{o'}|\text{correct}]^{\frac{1}{2}}\prod_{o=1}^{2^{32}} Pr[d^o|\text{wrong}]^{\frac{1}{2}}\right)\right)\right)^4 \\
&= \frac{1}{129}\cdot\left(\prod_{\substack{o=1\\ o\neq o'}}^{2^{32}}\sum_{d^o=0}^{d}\binom{d}{d^o}\cdot Pr[d^o|\text{wrong}]\sum_{d^{o'}=0}^{d}\binom{d}{d^{o'}}\cdot Pr[d^{o'}|\text{correct}]\right. \\
&\left.+(2^{32}-1)\cdot\left(\prod_{\substack{o=1\\ o\neq o'}}^{2^{32}}\sum_{d^o=0}^{d}\binom{d}{d^o}\cdot Pr[d^o|\text{wrong}]\right)\cdot\left(\sum_{d^{o'}=0}^{d}\binom{d}{d^{o'}}\cdot Pr[d^{o'}|\text{correct}]^{\frac{1}{2}}\cdot Pr[d^{o'}|\text{wrong}]^{\frac{1}{2}}\right)\right)^4
\end{aligned}
\qquad (3.25)
$$

ここで，以下の 2 通りの二項定理の関係を利用する (correct と wrong を交換して

も成立する).

$$\sum_{d^o=0}^{d}\binom{d}{d^o}Pr[d^o|\text{wrong}]$$

$$=\sum_{d^o=0}^{d}\binom{d}{d^o}Pr[\Delta l=0|\text{wrong}]^{d-d^o}\cdot Pr[\Delta l\neq 0|\text{wrong}]^{d^o}$$

$$=(Pr[\Delta l=0|\text{wrong}]+Pr[\Delta l\neq 0|\text{wrong}])^{d}=1 \tag{3.26}$$

$$\sum_{d^o=0}^{d}\binom{d}{d^o}Pr[d^o|\text{correct}]^{\frac{1}{2}}\cdot Pr[d^o|\text{wrong}]^{\frac{1}{2}}$$

$$=\sum_{d^o=0}^{d}\binom{d}{d^o}Pr[\Delta l=0|\text{correct}]^{\frac{d-d^o}{2}}\cdot Pr[\Delta l\neq 0|\text{correct}]^{\frac{d^o}{2}}$$

$$\cdot Pr[\Delta l=0|\text{wrong}]^{\frac{d-d^o}{2}}\cdot Pr[\Delta l\neq 0|\text{wrong}]^{\frac{d^o}{2}}$$

$$=\sum_{d^o=0}^{d}\binom{d}{d^o}(Pr[\Delta l=0|\text{correct}]^{\frac{1}{2}}\cdot Pr[\Delta l=0|\text{wrong}]^{\frac{1}{2}})^{d-d^o}$$

$$\cdot(Pr[\Delta l\neq 0|\text{correct}]^{\frac{1}{2}}Pr[\Delta l\neq 0|\text{wrong}]^{\frac{1}{2}})^{d-d^o}$$

$$=Pr[\Delta l=0|\text{correct}]^{\frac{1}{2}}\cdot Pr[\Delta l=0|\text{wrong}]^{\frac{1}{2}}+Pr[\Delta l\neq 0|\text{correct}]^{\frac{1}{2}}\cdot Pr[\Delta l\neq 0|\text{wrong}]^{\frac{1}{2}} \tag{3.27}$$

これらを代入すると，式 (3.22) が得られる． □

データ量 d を変化させて LG を導出することで，d と LG の関係が判明する．LG が t_{adv} よりも大きい場合，DBA-KE の計算量の期待値が攻撃者の計算能力を超えていることを示せる．LG は定数時間で求められるため，データ量に区分を設けて t_d 回 LG を導出した場合，情報理論的評価は t_d に対し線形時間で実行が可能である．しかし，推測エントロピーは計算量の期待値なので，攻撃が実行困難であるかを明確に決定できない．そのため，計算量が大きいランク評価を用いた実験的評価を行い，攻撃が実行困難となる条件を求める．

3.3.2 ランク評価を用いた実験的評価

DBA-KE の攻撃成功率を見積もる手法として，ランク評価を用いた実験的評価を導入する．ランク評価は漏洩値と正しい鍵を入力とし，鍵列挙を行った際の正しい鍵のランク (列における順番) を出力する．ランクは鍵列挙による攻撃の計算量の下界と一致し，本論文では t_k で定義した (第 3.2.4 項参照)．高速に実行でき，かつ，ランクの上界と下界の差が狭いランク評価として，本論文では付録 N に示すヒストグラムを用いたランク評価 HistRankEst[20] を用いる．なお，実験においては物理情報等を用いず，DBA-KE の計算機シミュレーションを行う．

AES-128 に対する DBA-KE の攻撃成功率 $p_{success}$ を見積もる手法を Algorithm 3.3 に示す．本来，ランク評価では漏洩値と正しい鍵を乱択して計算量を見積もるが，Algorithm 3.3 では漏洩値は 1 度だけ乱択し，正しい鍵のみを t_{end} 回乱択する．これにより，現実的な時間内で実験が可能となる．最も計算量が高くなるのが副鍵 $\mathbf{k}_i$ のヒストグラム H_i の生成であり，2^{32} 回の対数尤度計算が必要になる．さらに，d 個の標本平文組から対応する漏洩値組の差分を計算するため，計算量は $2^{32} \cdot d$ となる．そのため，Algorithm 3.3 では 1 度だけ漏洩値を乱択してヒストグラム H_i を生成する．この処理には並列処理が適用できる (Algorithm 3.3 中の “**in parallel do**” 内処理)．

4 つの副鍵に対応したヒストグラム H_1，H_2，H_3 及び H_4 を生成した後，正しい鍵 $\mathbf{k}^{\dagger}$ を乱択しつつ t_{end} 回の試行を行う．正しい鍵の対数尤度を計算した後，ランク評価 HistRankEst に入力してランク t_k を得る．$t_k \leq t_{adv}$ ならば攻撃は成功とみなす．上記を繰り返すことで攻撃成功率 $p_{success}$ が得られる．この結果から，$p_{success}$ が 0 となるデータ量を見積もる．

Algorithm 3.3 の計算量は $2^{32} \cdot d \cdot 4 + t_{end} \cdot t_{re}$ と見積もられる．$2^{32} \cdot d \cdot 4$ はヒストグラム H_i ($i \in \{1, 2, 3, 4\}$) 生成時の漏洩値の乱択に必要な暗号化回数であり，$t_{end} \cdot t_{re}$ はその後のランク評価の計算量である．なお，t_{re} はヒストグラムを用いたランク評価に必要な計算量 (付録 N 参照) である．

Algorithm 3.3 並列化した差分バイアス攻撃に対するランク評価

正しい鍵 $\mathbf{k}^{\dagger}$ を乱択する．
for $i = 1 \rightarrow 4$ **do**
  for $\acute{\mathbf{k}}_i^o \in \mathbb{F}_2^{32} \backslash \{\acute{\mathbf{k}}_i^{\dagger}\}$ **in parallel do**
    対数尤度比の初期化 $\lambda_{\acute{\mathbf{k}}_i^o} \leftarrow 0$.
    for $j = 1 \rightarrow d$ **do**
      差分が 1 ワード以外 0 である $(\acute{\mathbf{u}}_i, \acute{\mathbf{u}}_i')$ を乱択する．
      $(\acute{\mathbf{u}}_i, \acute{\mathbf{u}}_i')$ 及び $\acute{\mathbf{k}}_i^o$ から $(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}_i')$ を計算する．
      $(\acute{\mathbf{v}}_i, \acute{\mathbf{v}}_i')$ を含む選択平文組 $(\mathbf{v}, \mathbf{v}') \in \mathcal{V} \times \mathcal{V}$ を選ぶ．
      $(\mathbf{v}, \mathbf{v}')$ を $\mathbf{k}^{\dagger}$ で暗号化し，対応する漏洩値 (l, l') を乱択する．
      $\lambda_{\acute{\mathbf{k}}_i^o} \leftarrow \lambda_{\acute{\mathbf{k}}_i^o} + \log_2 \left(\frac{Pr[l \oplus l' | \text{correct}]}{Pr[l \oplus l' | \text{wrong}]} \right)$
    end for
    ビンの添字を計算 $b \leftarrow \lfloor \lambda_{\acute{\mathbf{k}}_i^o} / l_{bin} \rfloor$.　　▷ l_{bin}:ビンの幅
    ビンに属する候補数を更新 $\mathsf{H}_i(b) \leftarrow \mathsf{H}_i(b) + 1$.
  end for
end for
成功回数を初期化 $t_{success} \leftarrow 0$
for $t_{trial} = 1 \rightarrow t_{end}$ **do**
  正しい鍵 $\mathbf{k}^{\dagger}$ を乱択し，ビンの添字を初期化 $b^{\dagger} \leftarrow 0$.
  for $i = 1 \rightarrow 4$ **do**
    対数尤度比の初期化 $\lambda_{\acute{\mathbf{k}}_i^{\dagger}} \leftarrow 0$.
    for $j = 1 \rightarrow d$ **do**
      正しい副鍵 $\acute{\mathbf{k}}_i^{\dagger}$ に対応する選択平文組 $(\mathbf{v}, \mathbf{v}')$ を乱択する．
      選択平文組 $(\mathbf{v}, \mathbf{v}')$ を $\mathbf{k}^{\dagger}$ で暗号化し，(l, l') を乱択する．
      $\lambda_{\acute{\mathbf{k}}_i^{\dagger}} \leftarrow \lambda_{\acute{\mathbf{k}}_i^{\dagger}} + \log_2 \left(\frac{Pr[l \oplus l' | \text{correct}]}{Pr[l \oplus l' | \text{wrong}]} \right)$
    end for
    ビンの添字を計算 $b_i^{\dagger} \leftarrow \lfloor \lambda_{\acute{\mathbf{k}}_i^{\dagger}} / l_{bin} \rfloor$.
    ビンに属する候補数を更新 $\mathsf{H}_i(b_i^{\dagger}) \leftarrow \mathsf{H}_i(b_i^{\dagger}) + 1$.
    $b^{\dagger} \leftarrow b^{\dagger} + b_i^{\dagger}$
  end for
  $t_k \leftarrow \mathsf{HistRankEst}(\mathsf{H}_1, \mathsf{H}_2, \mathsf{H}_3, \mathsf{H}_4, b^{\dagger})$
  if $t_k \leq t_{adv}$ **then**
    $t_{success} \leftarrow t_{success} + 1$
  end if
end for
return $p_{success} = t_{success} / t_{end}$.

3.4 AES-128 への適用

先行研究 [9] との比較のため，AES-128 (付録 D，図 D.3，[12]) の DBA-KE への耐性を評価する．攻撃者 $\mathsf{A}_{t\leq 2^{60}}$ 及び $\mathsf{A}_{t\leq 2^{80}}$ を仮定する (付録 I 参照)．付録 O に示す理由から，これらの攻撃者はそれぞれ短期的安全性と長期的安全性に対応している．短期的に使用する暗号デバイスでは攻撃者 $\mathsf{A}_{t\leq 2^{60}}$ に対する安全性を確認すれば十分である．しかし，長期的に使用する暗号デバイスについては，攻撃者 $\mathsf{A}_{t\leq 2^{80}}$ に対する安全性を保証する必要がある．

本論文では，標本平文組数 d を増加させ，推測エントロピーの下界 LG 及び攻撃成功率 p_{succes} の変化を示す．しかし，ランク評価を用いた実験的評価には $2^{32}\cdot d\cdot 4$ 回の暗号化処理が必要であるので，表 3.3 に示す実験環境を用いて高速化する．また，データ量を $d = 500\cdot i$ ($i \in \{1, 2, ..., 5\}$) に制限する．この理由は，$d = 2,500$ で $2^{45.28}$ 回の暗号化処理で実験が可能であり，これは表 3.3 の環境で約 21 時間で実行可能となるからである．さらに，漏洩範囲を最も攻撃者に有利なワード単位 RL モデル (表 3.2 参照) における 2 段目出力 $\mathbf{w}_{2,i}$ 及びその段鍵 $\mathbf{rk}_{2,i}$ ($i \in \{1, 2, ..., 16]$) とする．この漏洩範囲で DBA-KE を防ぐことができれば，他の漏洩範囲についても防ぐことができる．

まず，情報理論的評価の結果を図 3.8 に示す．$d \leq 5,107$ 及び $d \leq 7,599$ のとき，LG は 2^{80} 及び 2^{60} よりも小さくなる．式 (3.16) を用いてデータ量に変換すると，$q \leq 2^{19.16}$ 及び $q \leq 2^{19.44}$ となる．

次に，$t_{end} = 2^{32}$ として Algorithm 3.3 を実行した結果を図 3.9 に示す．$d \leq 1,000$ 及び $d \leq 2,000$ の場合，攻撃者 $\mathsf{A}_{t\leq 2^{60}}$ 及び $\mathsf{A}_{t\leq 2^{80}}$ に対して攻撃成功率がほぼ 0 となる．式 (3.16) を用いてデータ量に変換すると，$q \leq 18.48$ 及び $q \leq 17.98$ である．したがって，実験を行った攻撃者にとってかなり有利な漏洩範囲においても，これらのデータ量を超えない場合は短期的または長期的な安全性が保証できる．

DBA-KE に対する対策として，漏洩するデータ量を削減するために鍵の使用期限を適切に設定する必要がある．その際，提案手法によって得られるデータ量が

表 **3.3**　計算機実験環境

CPU	Intel Xenon E5-2620 2.00GHz
メモリ	6GB
GPU	NVIDIA Tesla K20Xm (GK110) × 4
CUDA バージョン	6.00

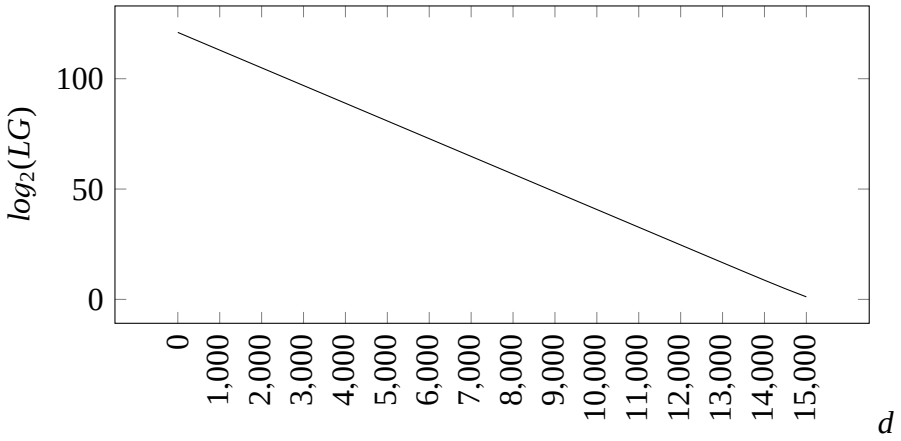
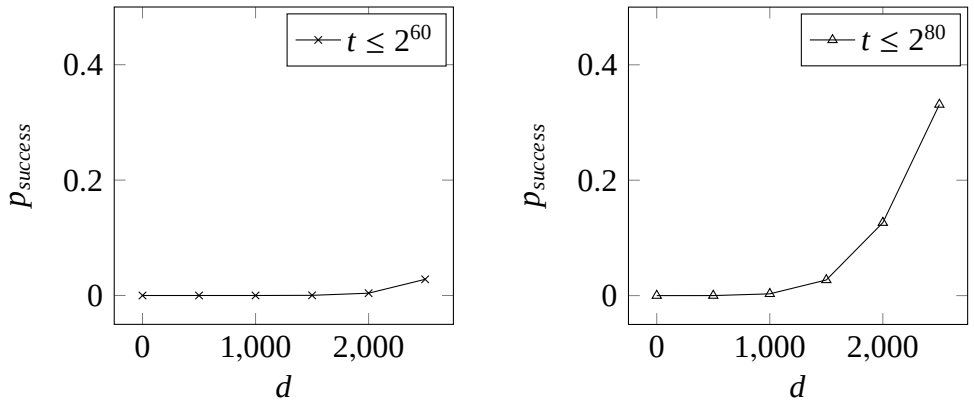


図 3.8　漏洩範囲 $\mathbf{W}_{2,i}$ 及び $\mathbf{RK}_{2,i}$ の場合の推測エントロピーの下界.



(a) $\mathsf{A}_{t\leq 2^{60}}$　　(b) $\mathsf{A}_{t\leq 2^{80}}$

図 3.9　漏洩範囲 $\mathbf{W}_{2,i}$ 及び $\mathbf{RK}_{2,i}$ の場合の Algorithm 3.3 の適用結果.

参考となる．また，攻撃者にとって有利な漏洩となる AES の 2 段目出力のような中間値について優先的に漏洩防止策を取ることが効果的である．

3.5　第 3 章のまとめ

実装の安全性評価として，RL モデルにおける形式的評価を提案した．まず，DBA-PM を改良した DBA-KE を提案した．DBA-KE は与えられた漏洩値に対して最適な列を生成して鍵を列挙するため，DBA-PM よりも少ないデータ量で攻撃が可能である．次に，DBA-KE に対する耐性評価手法を提案した．攻撃者にとって最も有利な 2 段目出力とその副鍵が漏洩範囲にあると仮定して AES-128 の評価を行った．その結果，データ量が $q \leq 2^{18.48}$ 及び $q \leq 2^{17.98}$ であるならば，攻撃者 $\mathsf{A}_{t\leq 2^{60}}$ 及び $\mathsf{A}_{t\leq 2^{80}}$ に対して DBA-KE を防げるため，短期的安全性及び長期的安全性を保証できる．

この方法で RL モデルにおける形式的評価を行った場合，DBA-KE を防ぐために必要なデータ量を得る．得られたデータ量を基準として，鍵更新の頻度を考えることで DBA-KE への対策となる．また，攻撃者にとって有利となる漏洩範囲が判明するので，漏洩防止を行う参考となる．

第 4 章

サイドチャネル Cube 攻撃に対する耐性評価手法

もうひとつの実装の安全性評価として，1 ビットアクセス漏洩モデルを仮定する (概要を図 4.1 に示す)[15].

定義 4.1 (1 ビットアクセス漏洩モデル)**.** 1 ビットアクセス漏洩モデル (1AL (1-bit Accessed Leakage) モデル) において，攻撃者は任意に選択した平文 $\mathbf{v}$ に対応する漏洩値 $l = \mathsf{f}(\mathbf{v}, \mathbf{k})$ を測定する．中間値 1 ビットを出力する $\mathsf{f} : \mathbb{F}_2^{n_{all}+n_{key}} \to \mathbb{F}_2$ は与えられており，攻撃の過程で変化しない．なお，攻撃者は f で求まる中間値にアクセスできると表現する．

1AL モデルにおいて安全性評価を行い，対策が行える既知の攻撃としては漏洩防止が行われている暗号デバイスへの攻撃が考えられる．これはハードウェア実装された暗号化処理を対象とした攻撃で，漏洩防止が十分に行われていない中間値の情報を消費電力等の物理情報から入手する．もし，情報が得られる中間値のビット数が非常に少ない場合，1AL モデルによって安全性の解析が可能となる．

1AL モデルにおいて有効な攻撃法として，Dinur らが提案したサイドチャネル Cube 攻撃 (SCCA (Side-Channel Cube Attack))[15] がある．SCCA では図 4.1 で示したとおり，漏洩値 l をブール関数 f で表現する．ブール関数の特性から，選択平文集合に対する積分で特定の項の係数が得られる (付録 K 参照)．このとき，鍵 $\mathbf{k}$ を変数とする線形方程式が得られる場合がある．なお，定数ビットがすべて 0 である選択平文集合を Cube とよび，これを用いる．線形方程式が得られる Cube が複数存在すれば線形方程式系が得られる．誤りのない測定環境においては，線形方程式系の独立な式数が変数の数を上回れば鍵は一意に推定できる．そのため，

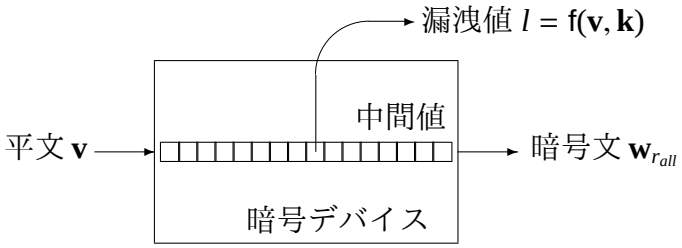


図 4.1　1 ビットアクセス漏洩モデルの概要

攻撃者はできるだけ多くの Cube を探索する (Cube 探索)．しかし，誤りのある測定環境においては誤り耐性を考慮した鍵推定が必要となる．

誤りをモデル化する方法として 2 元消失通信路を仮定したモデル (BEC (Binary Erasure Channel) モデル)[15] と 2 元対称通信路を仮定したモデル (BSC (Binary Symmetric Channel) モデル)[30] がある．BEC モデルにおける SCCA の誤り耐性は解析が容易であり，理論が確立されている [15]．これに対し，BSC モデルについては理論が確立されていないため，本論文では BSC モデルを採用する．なお，これに定義 4.1 の 1AL モデルを適用した 1AL-BSC モデルを仮定する．BSC モデルでは得られた漏洩値が反転確率 ρ で本来の値と異なる．このモデルにおける攻撃法として最尤復号を利用した攻撃法を Li らが提案している [30]．Li らの手法を SCCA-PM (Side-Channel Cube Attack - Previous Method) とよぶ．

本論文では SCCA-PM を改良して誤り耐性を向上させるため，第 3.2 節と同様に鍵列挙を利用した SCCA-KE (Side-Channel Cube Attack with Key Enumeration) を提案する．SCCA-KE は分割統治法を採用し，鍵を副鍵に分割してそれぞれ列を生成する．ここで，SCCA-PM ではハミング重みを基準にして整列するのに対し，SCCA-KE では対数尤度を基準にして整列する (第 3.2.1 項参照)．

SCCA-PM よりも誤り耐性の高い SCCA-KE を考慮することで，より安全な実装を行うことができる．そのため，SCCA-KE に対する耐性評価手法を提案する．その際，DBA-KE と同様に情報理論的評価とランク評価を用いた実験的評価を行

う (第 3.3.1 項及び第 3.3.2 項参照)．また，SCCA-KE を RESENT[8] に適用して評価を行う．

第 4.1 節で SCCA の先行研究を述べ，第 4.2 節で SCCA-KE を示す．SCCA-KE に対する耐性評価手法は第 4.3 節に示し，第 4.4 節で PRESENT に適用する．

4.1 サイドチャネル Cube 攻撃の先行研究

SCCA の研究には測定誤りを考慮したものと考慮しないものがある．本論文ではより現実的な仮定として前者を選択する．SCCA の原理を示すため第 4.1.1 項で誤りのない測定環境における SCCA の先行研究を示す．その後，第 4.1.2 項で誤りをモデル化して誤り耐性を研究した先行研究を紹介する．

4.1.1 誤りのない測定環境におけるサイドチャネル Cube 攻撃 [15]

$\mathsf{f} : \mathbb{F}_2^{n_{var}+n_{key}} \to \mathbb{F}_2$ を平文 $\mathbf{v}$ 及び鍵 $\mathbf{k}$ を変数とするブール関数とする．変数ビットの添字集合 $\mathcal{I} \subset \{1, 2, ..., n_{var}\}$ で定義する選択平文集合を $\mathcal{V}_\mathcal{I}$ とする．ブール関数の特性から，$\mathsf{a}_\mathcal{I}(\mathbf{v}_{const}, \mathbf{k}) = \bigoplus_{\mathbf{v}\in\mathcal{V}_\mathcal{I}} \mathsf{f}(\mathbf{v}, \mathbf{k})$ が得られる (付録 K 参照)．ここで，定数ビットが全て 0 である $\mathcal{V}_\mathcal{I}$ を Cube $\mathcal{C}_\mathcal{I}$ と定義する．平文 $\mathbf{v}$ を変数 $\mathbf{v}_{var}$ 及び定数 $\mathbf{v}_{const}$ に分割すると，$\mathcal{C}_\mathcal{I} = \{\mathbf{v} | \mathbf{v}_{var} \in \mathbb{F}_2^{n_{var}}, \mathbf{v}_{const} = 0\}$ と表現できる．また，$\mathcal{C}_\mathcal{I}$ による積分値を $\mathsf{a}'_\mathcal{I}(\mathbf{k}) = \bigoplus_{\mathbf{v}\in\mathcal{C}_\mathcal{I}} \mathsf{f}(\mathbf{v}, \mathbf{k})$ とする．例えば，$\mathsf{a}_\mathcal{I}(\mathbf{v}_{const}, \mathbf{k}) = k_1k_2 \oplus v_1k_3$ 及び $1 \notin \mathcal{I}$ ならば $\mathsf{a}'_\mathcal{I}(\mathbf{k}) = k_1k_2$ となる．

漏洩値を $x_\mathbf{v} = \mathsf{f}(\mathbf{v}, \mathbf{k})$ と表現する．SCCA では一般的に $\mathsf{a}'_\mathcal{I}(\mathbf{k}) = \bigoplus_{\mathbf{v}\in\mathcal{C}_\mathcal{I}} x_\mathbf{v}$ が線形方程式となるものを鍵の推定に用いる．以下に示す条件を満たす Cube ならば，線形方程式が得られる．

(i) $\mathsf{a}'_\mathcal{I}$ が定数関数でない．例えば，十分な数の乱択した $\mathbf{k}$ に対して $\mathsf{a}'_\mathcal{I}(\mathbf{k})$ が同じ値となるならば，$\mathsf{a}'_\mathcal{I}$ は定数関数とみなす．

(ii) $\mathsf{a}'_\mathcal{I}$ が線形関数である．線形関数を判定する BLR テスト [4] を利用する．以下の式が十分な数の乱択した $\mathbf{k}^o$ 及び $\mathbf{k}^{o'}$ に対して成立するならば，$\mathsf{a}'_\mathcal{I}$ を線

形関数とみなす．なお，**0** は全ての成分が 0 である鍵を示す．

$$\mathsf{a}'_{\mathcal{I}}(\mathbf{k}^o) \oplus \mathsf{a}'_{\mathcal{I}}(\mathbf{k}^{o'}) \oplus \mathsf{a}'_{\mathcal{I}}(\mathbf{0}) = \mathsf{a}'_{\mathcal{I}}(\mathbf{k}^o \oplus \mathbf{k}^{o'}) \tag{4.1}$$

このような判定を Cube 判定とよぶ．一般的にそれぞれ 100 個の乱択した鍵の候補に対して $\mathsf{a}'_{\mathcal{I}}(\mathbf{k})$ を計算する [15].

攻撃者が L 個の異なる Cube から以下の線形方程式系を得られると仮定する．

$$\begin{aligned} b_{1,1} \cdot k_1 \oplus b_{1,2} \cdot k_2 \oplus \ldots \oplus b_{1,n} \cdot k_n &= x_1 \\ b_{2,1} \cdot k_1 \oplus b_{2,2} \cdot k_2 \oplus \ldots \oplus b_{2,n} \cdot k_n &= x_2 \\ &\vdots \\ b_{L,1} \cdot k_1 \oplus b_{L,2} \cdot k_2 \oplus \ldots \oplus b_{L,n} \cdot k_n &= x_L \end{aligned} \tag{4.2}$$

ここで $x_i = \bigoplus_{\mathbf{v} \in \mathcal{C}_{\mathcal{I}_i}} x_{\mathbf{v}} \oplus \mathsf{a}'_{\mathcal{I}_i}(\mathbf{0})$ であり，$\mathsf{a}'_{\mathcal{I}_i}(\mathbf{0})$ は多項式の定数項である．なお，線形方程式系に含まれる k_i を鍵変数，x_i を RHS (Right-Hand Side) 値とよび，それらのベクトルを **k** 及び **x** とする．独立な線形方程式の数が n よりも大きいならば，鍵変数は一意に推定される．

ここで，SCCA では一般的に鍵の一部を上記の方法で推定する．つまり，一部の鍵は一切の情報が得られず，総当たりによる推定が必要となる．本章では $\mathbf{k}_l$ を式 (4.2) の鍵変数のように線形方程式系から情報が得られる n_l ビットの鍵，$\mathbf{k}_{nl}$ を線形方程式の鍵変数に含まれない n_{nl} ビットの鍵とする．なお，完全な鍵は $\mathbf{k} = (\mathbf{k}_l, \mathbf{k}_{nl})$ と表現する (鍵長 $n_{key} = n_l + n_{nl}$).

4.1.2 誤りのある測定環境におけるサイドチャネル Cube 攻撃 [30]

誤りをモデル化する方法としては BEC モデル [15] と BSC モデル [30] が提案されている．前述のように BEC モデルにおける SCCA の手法は確立されているため [15]，本論文では BSC モデルにおいて誤り耐性のある SCCA を対象とする．図 4.2 に BSC モデルにおける反転確率の概要を示す．BSC モデルでは測定した

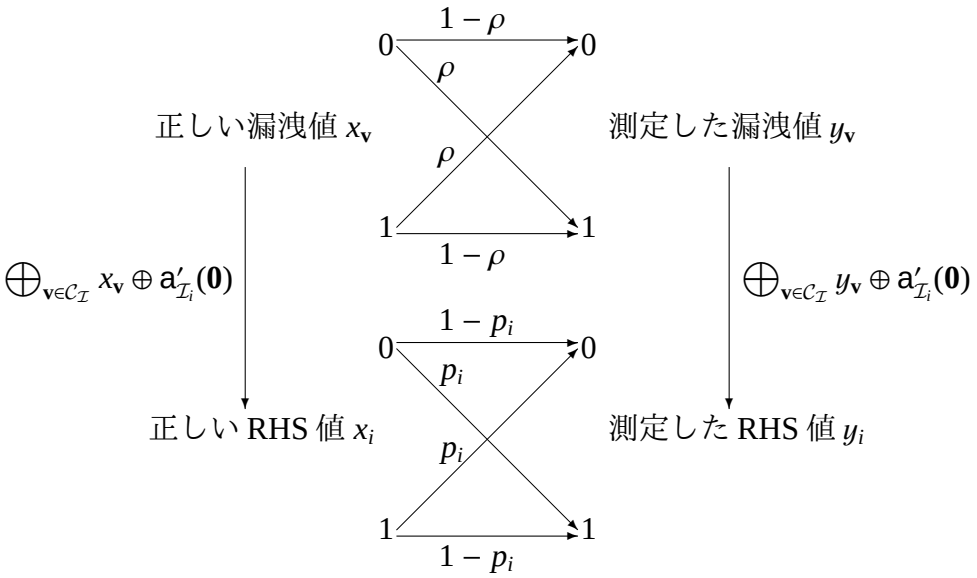


図 4.2　BSC モデルにおける反転確率の概要図

漏洩値 $y_{\mathbf{v}}$ が反転確率 $\rho = Pr[x_{\mathbf{v}} \oplus y_{\mathbf{v}} = 1] = 1/2 - \mu$ にしたがって得られる ($x_{\mathbf{v}}$ は正しい漏洩値)．Piling-up 補題 [36] を用いると，RHS 値 $x_i = \bigoplus_{\mathbf{v}\in\mathcal{C}_{\mathcal{I}_i}} x_{\mathbf{v}} \oplus \mathsf{a}'_{\mathcal{I}_i}(\mathbf{0})$ の反転確率は

$$p_i = Pr[x_i \oplus y_i = 1] = \frac{1}{2} - 2^{|\mathcal{C}_{\mathcal{I}_i}|-1}\mu^{|\mathcal{C}_{\mathcal{I}_i}|} \tag{4.3}$$

で得られる [30]．ここで，$y_i = \bigoplus_{\mathbf{v}\in\mathcal{C}_{\mathcal{I}_i}} y_{\mathbf{v}} \oplus \mathsf{a}'_{\mathcal{I}_i}(\mathbf{0})$ は測定した漏洩値の総和で得られる．x_i 及び y_i を区別するため，それぞれ正しい RHS 値及び測定した RHS 値とよび，それぞれのベクトルを $\mathbf{x}$ 及び $\mathbf{y}$ とする．

BSC モデルにおいて式 (4.2) の線形方程式系から鍵を推定する方法として，Li らは最尤復号を利用した SCCA-PM を提案している．SCCA-PM では鍵推定を $[L,n_l]$ 線形符号の復号として実行している．

$B = (b_{i,j})$ ($i \in \{1, 2, ..., L\}$, $j \in \{1, 2, ..., n_l\}$) を $(L \times n_l)$ 行列 ($[L,n_l]$ 線形符号の生成行列) とし，$\mathbf{b}_i$ を B の i 番目の列ベクトルとする．$x_i = \mathbf{k}_l \cdot \mathbf{b}_i$ を計算することで正しい RHS 値が得られる．一方，攻撃者はこれに対応する測定した RHS 値 y_i を得る．線形方程式系を $(B, \mathbf{k}_l, \mathbf{x})$ または $(B, \mathbf{k}_l, \mathbf{y})$ と表現する．SCCA-PM では，鍵 $\mathbf{k}_l$

の候補 $\mathbf{k}_l^o$ に対するハミング重み HD から

$$\arg\min_{\mathbf{k}_l^o} \mathrm{HD}(\mathbf{k}_l^o),\ \mathrm{HD}(\mathbf{k}_l^o) = \sum_{i=1}^{L}(x_i^o \oplus y_i),\ (x_i^o = \mathbf{k}_l^o \cdot \mathbf{a}_i) \tag{4.4}$$

なる $\mathbf{k}_l^o$ を出力する．

一般的な線形符号の復号問題は n_l が大きい場合に実行が困難になる．計算量を削減するため，SCCA-PM は分割統治法を採用し，$\mathbf{k}_l$ を m_{key} 個の副鍵 $\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_{m_{key}}$ に分割する．ここで，$\mathbf{k}_j$ の成分を少なくとも 1 個は鍵変数として有する線形方程式を考え，これに対応する列ベクトルを集めたものを行列 B_j とする．以後，$(B_j, \mathbf{k}_j, \mathbf{x}_j)$ または $(B_j, \mathbf{k}_j, \mathbf{y}_j)$ を副線形方程式系とよぶ.

SCCA-PM では各副線形方程式系で全副鍵の候補 $\mathbf{k}_j$ のハミング重みを計算する．なお，SCCA-PM の分割統治法においては，提案者の Li らは副鍵の鍵変数を他の副線形方程式系と共有させることで計算量が削減できると主張している．このような方法を重複分割統治法とよぶ.

攻撃成功率を高めるため，複数の候補を列として保存する．その際の列はハミング重みを基準として昇順で整列されている．また，SCCA-PM では副線形方程式系においてそれぞれ t_{const} 個の副鍵の候補を保存する．$\mathcal{T}_{\mathbf{k}_j}$ を副鍵 $\mathbf{k}_j$ の候補の列とする $(|\mathcal{T}_{\mathbf{k}_j}| = t_{const})$．$m_{key}$ 個副線形方程式系が存在するため，$\mathbf{k}_l$ の候補数は $(t_{const})^{m_{key}}$ となる．この候補の列を $\mathcal{T}_{\mathbf{k}_l}(= \mathcal{T}_{\mathbf{k}_1} \times \mathcal{T}_{\mathbf{k}_2} \times ... \times \mathcal{T}_{\mathbf{k}_{m_{key}}})$ とする．また，完全な鍵 $\mathbf{k}$ を入力して正しい鍵かどうかの検証を行う必要があるので，情報がない鍵 $\mathbf{k}_{nl}$ の列 $\mathcal{T}_{\mathbf{k}_{nl}}$ $(|\mathcal{T}_{\mathbf{k}_{nl}}| = 2^{n_{nl}})$ を考慮する必要がある．攻撃者は $\mathcal{T}_{\mathbf{k}} = \mathcal{T}_{\mathbf{k}_l} \times \mathcal{T}_{\mathbf{k}_{nl}}$ の中から正しい鍵を探索する．したがって，計算量は $(t_{const})^{m_{key}} \cdot 2^{n_{nl}}$ となる．しかし，重複分割統治法で共有されている副鍵のビットが一致する候補のみ検証を行うため，計算量が削減できると Li らは述べている.

4.2 鍵列挙を用いたサイドチャネル Cube 攻撃の提案

SCCA-PM を改良して誤り耐性を向上させた SCCA-KE を提案する．SCCA-KE は鍵の推定を線形方程式系から行い，分割統治法を用いるという点については SCCA-PM を踏襲している．SCCA-PM との相違点は列の生成法が異なることである．SCCA-KE では m_{key} 個の副列 $\{\mathcal{T}_{\mathbf{k}_1}, \mathcal{T}_{\mathbf{k}_2}, ..., \mathcal{T}_{\mathbf{k}_{m_{key}}}\}$ と情報がない鍵 $\mathbf{k}_{nl}$ の列 $\mathcal{T}_{\mathbf{k}_{nl}}$ から鍵列挙を用いて $\mathcal{T}_{\mathbf{k}}$ を生成する．なお，$\mathcal{T}_{\mathbf{k}_{nl}}$ については情報が得られていないので，ランダムに整列する．

第 3.2.1 項の定義 3.3 の条件を SCCA に適用する．なお，$\mathbf{y}$ を線形方程式系の測定した RHS 値ベクトルとし，各測定は独立とする．鍵候補の列 $\mathcal{T}_{\mathbf{k}_l}$ が以下を満たすとき，$\mathcal{T}_{\mathbf{k}_l}$ は最適な列である．

(i) 任意の $o < o'$ なる $\mathbf{k}_l^o, \mathbf{k}_l^{o'} \in \mathcal{T}_{\mathbf{k}_l}$ に対し以下が成立する．

$$\sum_{i=1}^{L} \log_2(Pr[y_i|\mathbf{k}_l^o]) \geq \sum_{i=1}^{L} \log_2(Pr[y_i|\mathbf{k}_l^{o'}]) \tag{4.5}$$

(ii) 任意の $\mathbf{k}_l^o \in \mathcal{T}_{\mathbf{k}_l}$ 及び $\mathbf{k}_l^{o''} \notin \mathcal{T}_{\mathbf{k}_l}$ に対し以下が成立する．

$$\sum_{i=1}^{L} \log_2(Pr[y_i|\mathbf{k}_l^o]) \geq \sum_{i=1}^{L} \log_2(Pr[y_i|\mathbf{k}_l^{o''}]) \tag{4.6}$$

ここで $Pr[y_i|\mathbf{k}_l^o] = Pr[x_i^o \oplus y_i]$ であり，$\mathbf{x}^o = (x_1^o, x_2^o, ..., x_L^o) = \mathbf{k}_l^o \cdot B$ と求まる．

SCCA-KE ではまず，副列を最適な列として生成する．ここで，SCCA-PM のように重複分割統治法を採用せず，各副線形方程式系で他の系と鍵変数が共有されないように分割する．このように鍵を副鍵に分割した際の対数尤度は

$$\sum_{j=1}^{m_{key}} \sum_{i=1}^{L_j} \log_2(Pr[y_{j,i}|\mathbf{k}_j^o]) \tag{4.7}$$

と求まる ($\mathbf{k}_l^o = (\mathbf{k}_1^o, \mathbf{k}_2^o, ..., \mathbf{k}_{m_{key}}^o)$)．ここで，$L_j$ は j 番目の副線形方程式系の線形方程式数とする．

一方で，SCCA-PM では以下の理由から現在のところ最適な列を生成することができない．

(i) 反転確率 p_i が Cube サイズ $|\mathcal{C}_{\mathcal{I}_i}|$ によって異なるときハミング重みで整列された列は最適な列にならない．$\mathsf{HD}(\mathbf{k}_l^o) \leq \mathsf{HD}(\mathbf{k}_l^{o'})$ が式 (4.5) と同値になるのは全ての線形方程式において p_i が同じ値となる場合のみである．p_i は Cube サイズに依存するので，ハミング重みを使用するのは適当でない．

(ii) 重複分割統治法では鍵列挙を適用することができない．もし対数尤度が式 (4.7) で求められるのであれば，鍵列挙を適用できる．しかし，鍵変数が重複している場合，重複した鍵変数の対数尤度を減算する．また，このような重複を含む副列から最適な列を生成できる鍵列挙はまだ提案されていない．

4.2.1 攻撃の手順

SCCA-KE を Algorithm 4.1 に示す．まず，線形方程式系 $(B, \mathbf{k}, \mathbf{y})$ を m_{key} 個の副線形方程式系に分割する．その際，鍵変数において重複が存在しないので，対数尤度は各副線形方程式系で独立に計算できる．以下に例を示す．

$$\left(\left(\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}, (k_1, k_2, k_3, k_4, k_5, k_6), (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)\right)\right)$$
$$\Rightarrow \left(\left(\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, (k_1, k_4), (y_1, y_7)\right), \left(\left(\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}, (k_2, k_5), (y_2, y_3, y_5)\right), \left(\left(\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, (k_3, k_6), (y_4, y_6)\right)\right.\right. \tag{4.8}$$

各副線形方程式系において，全候補の対数尤度 $\lambda_{\mathbf{k}_j^o}$ を計算する．副鍵の候補及びその対数尤度 $(\mathbf{k}_j^o, \lambda_{\mathbf{k}_j^o})$ を列 $\mathcal{U}_{\mathbf{k}_j}$ に対数尤度で降順に保存する．第 4.1 節で述べたとおり，n_{nl} ビットの情報がない鍵 $\mathbf{k}_{nl}$ が存在するので $\mathcal{U}_{\mathbf{k}_{nl}}$ を用意する．測定し

Algorithm 4.1 鍵列挙を用いたサイドチャネル Cube 攻撃 (SCCA-KE)

input $(B, \mathbf{k}_l, \mathbf{y})$
$(B, \mathbf{k}_l, \mathbf{y})$ を m_{key} 個の副系 $(B_j, \mathbf{k}_j, \mathbf{y}_j)$ $(j \in \{1, 2, ..., m_{key}\})$ に分割する．
for $j = 1 \rightarrow m_{key}$ **do**
  for $o = 1 \rightarrow 2^{n_j}$ **do**
    $\mathbf{x}_j^o \Leftarrow \mathbf{k}_j^o \cdot B_j$
    $\lambda_{\mathbf{k}_j^o} \Leftarrow \sum_{i=0}^{L_j} \log_2(Pr[x_{j,i}^o \oplus y_{j,i}])$ ▷ $Pr[x_{j,i}^o \oplus y_{j,i}]$ は式 (4.3) で求める．
  end for
  任意の $(\mathbf{k}_j^o, \lambda_{\mathbf{k}_j^o})$ を $\mathcal{U}_{\mathbf{k}_j}$ において $\lambda_{\mathbf{k}_j^o}$ に対して降順で保存する．
end for
任意の $(\mathbf{k}_{nl}^o, -n_{nl})$ を $\mathcal{U}_{\mathbf{k}_{nl}}$ に保存する．
$\mathbf{k}^\dagger \Leftarrow \mathsf{KeyEnum}(\mathcal{U}_{\mathbf{k}_1}, \mathcal{U}_{\mathbf{k}_2}, ..., \mathcal{U}_{\mathbf{k}_{m_{key}}}, \mathcal{U}_{\mathbf{k}_{nl}}, t_k)$
return $\mathbf{k}^\dagger$ or “失敗”

た情報がないので，対数尤度は一様に $\log_2(2^{-n_{nl}}) = -n_{nl}$ であり，$\mathcal{U}_{\mathbf{k}_{nl}}$ にランダムに整列して保存する．次に，鍵列挙を用いて正しい鍵 $\mathbf{k}^\dagger$ を探索する．その際，t_k 個の鍵を対数尤度順に検証する．これら t_k 個の鍵の中に正しい鍵が含まれないならば，Algorithm 4.1 の探索は失敗する．ここで，関数 KeyEnum には最適な鍵列挙 [48] を使用する (付録 M 参照).

この結果，SCCA-KE では最適な列を生成することができる．その理由は以下のとおりである．

(i) Cube サイズを考慮することで正確な対数尤度を計算して副列 $\mathcal{U}_{\mathbf{k}_j}$ を生成する．鍵列挙を用いることで $\mathcal{U}_{\mathbf{k}}$ は最適な列となり，最尤復号に基づいた鍵推定が可能となる．

(ii) SCCA-KE では重複分割統治法を採らない．つまり，副線形方程式系は互いに鍵変数を共有せず，それぞれ独立して副列を生成することができる．これにより，既存の鍵列挙 [48] の適用が可能となる．

4.2.2 計算量の評価

DBA-KE と同様，式 (3.18) で SCCA-KE の計算量は求まる．その中で，副列の生成に必要な計算量 t_{prep} は分割統治法によって各副線形方程式系の鍵変数の数 n_j に対し

$$t_{prep} = \sum_{j=1}^{m_{key}} L_j \cdot 2^{n_j} \tag{4.9}$$

と求まる．なお，副列の生成については第 4.3.3 項に示す実験的評価において実行される．しかし，t_{prep} は n_j に対して指数関数的に計算量が増加することから，n_j が大きい場合には実行が困難になる．この点に関し，以下の観測 (Observation) が PRESENT に対する実験結果 (第 4.4.1 項に示す) と他のブロック暗号に対する先行研究 [1][23] から得られている．

観測 (Observation). ブロック暗号の解析において，n_l 個の鍵変数の線形方程式系を互いに鍵変数を共有しない m_{key} 個の副線形方程式系に分割する．その際，各副線形方程式系の鍵変数の数 n_j は一般的に小さい．

したがって，一般的に式 (4.9) の計算量である副列の生成は実行可能である．鍵列挙の計算量 t_{ke} については第 3.2.4 項と同様の考察が行える．したがって，第 4.3 節の評価手法においては DBA-KE と同様，全体の計算量は $t = t_k$ と仮定する．

4.3 鍵列挙を用いたサイドチャネル Cube 攻撃に対する耐性評価手法の提案

1AL-BSC モデルを仮定し，SCCA-KE に対する耐性評価手法を提案する．その際，攻撃者は $\mathsf{A}_{t \le t_{adv}, q \le q_{adv}}$ を仮定する (付録 I 参照)．この仮定で SCCA-KE を防ぐのに十分な漏洩する中間値や反転確率の条件を導出する．提案する評価手法では以下の 3 つの評価が存在する．

(i) Cube 探索 (第 4.3.1 項): 計算機実験で総当りによる Cube 探索を行う．探索結果から，$\rho = 0$ であっても攻撃が計算量的に困難となる条件が判明する．

(ii) 情報理論的評価 (第 4.3.2 項): 推測エントロピーの下界を理論的に求め，攻撃者の計算能力 t_{adv} を超える反転確率 ρ を求める．

(iii) ランク評価を用いた実験的評価 (第 4.3.3 項): 攻撃成功率をランク評価を用いて実験的に求め，攻撃成功率が 0 となる反転確率 ρ を求める．

DBA-KE と同様，計算量が大きいランク評価を用いた実験的評価が困難な場合，情報理論的評価のみで評価する必要がある (第 3.3 節参照).

4.3.1 Cube 探索

Cube 探索では，暗号化処理の全中間値に対して線形方程式が得られる Cube を探索する．これにより，アクセスできる中間値，Cube 及び線形方程式の対応が得られる．なお，Cube $\mathcal{C}_{\mathcal{I}}$ を定義する変数ビットの添字集合 $\mathcal{I}$ を変化させつつ，第 4.1.1 項で示した Cube 判定を適用する．現在提案されている Cube 探索には，ランダムウォークによるもの [15][30] と総当りによるもの [23] がある．ランダムウォークについては十分な試行回数が示されていないそのため，包括的に安全性を評価するならば，総当りによる Cube 探索を行う必要がある．本論文では添字集合 $\mathcal{I}$ について総当りによる Cube 探索を行う．

総当りによる Cube 探索では $\binom{n_{all}}{|\mathcal{I}|}$ 回の Cube 判定が必要となる．Cube 判定においては一般的に，$100 \times 2^{|\mathcal{I}|}$ 回の試行を行うことで信頼できる結果を得る [15]. $n_{all} = 64$ であるときの Cube サイズに対する試行回数を表 4.1 にまとめる．

ここで，$\rho \geq 0.1$ のように誤りが無視できない測定環境では Cube サイズが大きいものは探索を行う必要がないと考える．この理由について，図 4.3 を用いて示す．図 4.3 には反転確率 $\rho \in (0 : 0.5]$ に対する RHS 値の反転確率 p_i の関係を示す (式 (4.3) 参照)．任意の Cube サイズにおいて，$\lim_{\rho \to 0.5} p_i = 0.5$ となり攻撃が困難になる．Cube サイズが大きい場合，この収束が早くなる．図 4.3 の $|\mathcal{I}| > 6$ に注目すると，反転確率が $\rho \geq 0.1$ のとき $p_i \approx 0.5$ となることが分かる．したがって，第 4.4.1 項では $|\mathcal{I}| \leq 6$ という制約を設け，攻撃が実行困難となる反転確率 ρ を求める．

表 4.1　総当りの Cube 探索に必要な試行回数

Cube サイズ $\lvert\mathcal{I}\rvert$	試行回数
3	$2^{24.99}$
4	$2^{29.92}$
5	$2^{34.51}$
6	$2^{38.80}$
7	$2^{42.85}$
8	$2^{46.69}$

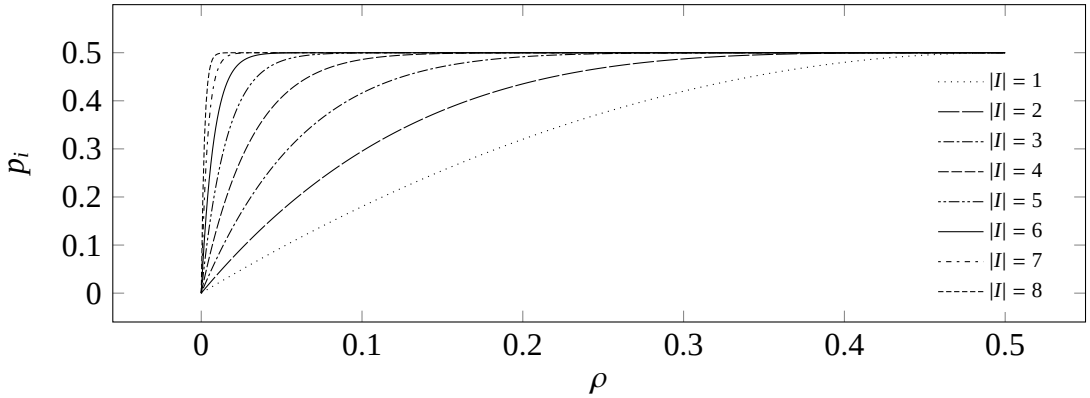


図 4.3　Cube サイズと反転確率の関係

Cube 探索結果から，測定に誤りがない場合 (反転確率 $\rho = 0$) であっても攻撃が実行困難となる条件を示すことができる．情報がない鍵変数の数 n_{nl} が $2^{n_{nl}} \geq t_{adv}$ となる場合，確定的に計算量が t_{adv} を超える．したがって，アクセスできる中間値が与えられたとき，この条件を満たす場合には $\rho = 0$ であっても攻撃が実行困難であると示せる．

4.3.2　情報理論的評価

DBA-KE と同様に推測エントロピー (定義 3.4 参照) の下界を求めることで計算量の期待値の下界を求める．SCCA-KE では LG ($G \geq LG$) は以下のように求まる．

命題 4.1. SCCA-KE では推測エントロピーの下界 LG は

$$
\begin{aligned}
LG &= \frac{1}{1+n_{key}} \sum_{\mathbf{y}} \left(\sum_{\mathbf{k}^{\dagger}} Pr[\mathbf{k}^{\dagger}, \mathbf{y}]^{\frac{1}{2}} \right)^2 \\
&= \frac{2^{n_{nl}}}{1+n_{key}} \prod_{j=1}^{m_{key}} \left(2^{-n_j} \sum_{o=1}^{2^{n_j}} \sum_{\substack{o'=1 \\ o' \neq o}}^{2^{n_j}} \left(\prod_{\substack{i=1 \\ x^o_{j,i} \neq x^{o'}_{j,i}}}^{L_j} 2 \cdot p_{j,i}^{\frac{1}{2}} \cdot (1-p_{j,i})^{\frac{1}{2}} \right) + 1 \right)
\end{aligned} \tag{4.10}
$$

と求まる．なお，$\mathbf{x}^o_j = \mathbf{k}^o_j \cdot B_j = (x^o_{j,1}, x^o_{j,2}, ..., x^o_{j,L_j})$ は $\mathbf{k}^o_j$ が正しい副鍵である場合の RHS 値ベクトルであり，$p_{j,i} = Pr[x^o_{j,i} \oplus y_{j,i} = 1]$ とする．

証明 . LG は副線形方程式系ごとに計算する．その際，鍵候補の事前確率は一様分布とみなす ($Pr[\mathbf{k}_j] = 2^{-n_j}$, $Pr[\mathbf{k}_{nl}] = 2^{-n_{nl}}$).

$$
\begin{aligned}
LG &= \frac{1}{1+n_{key}} \sum_{\mathbf{y}} \left(\sum_{\mathbf{k}^{\dagger}_l} \prod_{j=1}^{m_{key}} Pr[\mathbf{y}_j | \mathbf{k}^{\dagger}_j]^{\frac{1}{2}} \cdot Pr[\mathbf{k}^{\dagger}_j]^{\frac{1}{2}} \right)^2 \cdot \left(\sum_{\mathbf{k}^{\dagger}_{nl}} Pr[\mathbf{k}^{\dagger}_{nl}]^{\frac{1}{2}} \right)^2 \\
&= \frac{2^{n_{nl}}}{1+n_{key}} \prod_{j=1}^{m_{key}} 2^{-n_j} \sum_{\mathbf{y}_j} \left(\sum_{\mathbf{k}^{\dagger}_j} Pr[\mathbf{y}_j | \mathbf{k}^{\dagger}_j]^{\frac{1}{2}} \right)^2 \\
&= \frac{2^{n_{nl}}}{1+n_{key}} \prod_{j=1}^{m_{key}} LG_j
\end{aligned} \tag{4.11}
$$

LG_j の計算について簡略化を行う．まず，2 乗部分について展開を行う．

$$
\begin{aligned}
LG_j &= 2^{-n_j} \sum_{\mathbf{y}_j} \left(\sum_{\mathbf{k}^{\dagger}_j} Pr[\mathbf{y}_j | \mathbf{k}^{\dagger}_j]^{\frac{1}{2}} \right)^2 \\
&= 2^{-n_j} \sum_{\mathbf{y}_j} \sum_{o=1}^{2^{n_j}} \sum_{o'=1}^{2^{n_j}} Pr[\mathbf{y}_j | \mathbf{k}^{o}_j]^{\frac{1}{2}} \cdot Pr[\mathbf{y}_j | \mathbf{k}^{o'}_j]^{\frac{1}{2}} \\
&= 2^{-n_j} \sum_{\mathbf{y}_j} \sum_{o=1}^{2^{n_j}} \sum_{o'=1}^{2^{n_j}} \prod_{i=1}^{L_j} Pr[y_{j,i} \oplus x^{o}_{j,i}]^{\frac{1}{2}} \cdot Pr[y_{j,i} \oplus x^{o'}_{j,i}]^{\frac{1}{2}}
\end{aligned}
$$

総和と積の順序を変更することで以下のように簡略化が達成される．

$$
\begin{aligned}
LG_j &= 2^{-n_j} \sum_{o=1}^{2^{n_j}} \sum_{o'=1}^{2^{n_j}} \sum_{\mathbf{y}_j} \prod_{i=1}^{L_j} Pr[y_{j,i} \oplus x_{j,i}^{o}]^{\frac{1}{2}} \cdot Pr[y_{j,i} \oplus x_{j,i}^{o'}]^{\frac{1}{2}} \\
&= 2^{-n_j} \sum_{o=1}^{2^{n_j}} \sum_{o'=1}^{2^{n_j}} \prod_{i=1}^{L_j} \sum_{y_{j,i}} Pr[y_{j,i} \oplus x_{j,i}^{o}]^{\frac{1}{2}} \cdot Pr[y_{j,i} \oplus x_{j,i}^{o'}]^{\frac{1}{2}} \\
&= 2^{-n_j} \sum_{o=1}^{2^{n_j}} \sum_{o'=1}^{2^{n_j}} \prod_{i=1}^{L_j} \left(Pr[0 \oplus x_{j,i}^{o}]^{\frac{1}{2}} \cdot Pr[0 \oplus x_{j,i}^{o'}]^{\frac{1}{2}} + Pr[1 \oplus x_{j,i}^{o}]^{\frac{1}{2}} \cdot Pr[1 \oplus x_{j,i}^{o'}]^{\frac{1}{2}} \right) \\
&= 2^{-n_j} \sum_{o=1}^{2^{n_j}} \sum_{\substack{o'=1 \\ o' \neq o}}^{2^{n_j}} \left(\prod_{\substack{i=1 \\ x_{j,i}^{o} \neq x_{j,i}^{o'}}}^{L_j} 2 \cdot p_{j,i}^{\frac{1}{2}} \cdot (1 - p_{j,i})^{\frac{1}{2}} \right) + 1
\end{aligned}
\tag{4.12}
$$

なお，最後の変形では $x_{j,i}^{o} = x_{j,i}^{o'}$ の場合は確率の積は常に 1 となるので，$x_{j,i}^{o} \neq x_{j,i}^{o'}$ なる確率の積のみを考慮している．また，もともと $o = o'$ の場合は全ての確率の積が 1 となる．このような $(\mathbf{k}_j^{o}, \mathbf{k}_j^{o'})$ の対は 2^{n_j} 個存在し，$2^{-n_j} \cdot 2^{n_j} = 1$ が最後に加算されている． □

式 (4.10) の計算量は

$$
t_{LG} = \sum_{j=1}^{m_{key}} L_j \cdot 2^{n_j} \cdot (2^{n_j} - 1) \tag{4.13}
$$

となる．第 4.2.2 項で示した観測 (Observation) から n_j は一般的に小さいため，t_{LG} は現実的な実行時間となる．この結果，LG が t_{adv} よりも大きい場合，SCCA-KE の計算量の期待値が攻撃者の計算能力を超えていることを示せる．しかし，攻撃が実行困難であるかは判明しないので，計算量が大きいランク評価を用いた実験的評価が必要となる．

4.3.3 ランク評価を用いた実験的評価

SCCA-KE の計算機シミュレーションを複数回試行し，攻撃成功率 $p_{success}$ を見積もる手法を Algorithm 4.2 に示す．なお，副鍵のビット長 n_j が一般的に短いことから各試行の計算量が Algorithm 3.3 よりも小さくなる．そのため，試行ごと

Algorithm 4.2 鍵列挙を用いたサイドチャネル Cube 攻撃 (SCCA-KE) に対するランク評価を用いた実験的評価手法

input 計算量の閾値 t_{adv}，変数が $\mathbf{k}$ である行列 B 及び確率ベクトル $\mathbf{p}$.
$(B, \mathbf{k}, \mathbf{y})$ を m_{key} 個の副系 $(B_j, \mathbf{k}_j, \mathbf{y}_j)$ $(j \in \{1, 2, ..., m_{key}\})$ に分割する.
$\mathbf{p}$ を $(B_j, \mathbf{k}_j, \mathbf{y}_j)$ に対応する m_{key} 個のベクトル $\mathbf{p}_j$ $(j \in \{1, 2, ..., m_{key}\})$ に分割する.
$t_{success} \leftarrow 0$ 及び $b^{\dagger} \leftarrow 0$ ▷ $t_{success}$ は成功数に対するカウンター変数.
for $t_{trial} = 1 \rightarrow t_{end}$ **do** ▷ $b^{\dagger}$ は正しい鍵が属するビンの添字.
  for $j = 1 \rightarrow m_{key}$ **do**
    正しい副鍵 $\mathbf{k}_j^{\dagger}$ の値を乱択する.
    $\mathbf{x}_j^{\dagger} \leftarrow \mathbf{k}_j^{\dagger} \cdot B_j$
    $\lambda_{\mathbf{k}_j^{\dagger}} \leftarrow \sum_{i=1}^{L_j} \log_2(Pr[x_{j,i}^{\dagger} \oplus y_{j,i}])$ ▷ $Pr[x_{j,i}^{\dagger} \oplus y_{j,i} = 1] = p_{j,i}$.
    $b^{\dagger} \leftarrow b^{\dagger} + \lfloor \lambda_{\mathbf{k}_i^{\dagger}} / l_{bin} \rfloor$. ▷ l_{bin}:ビンの幅
    $\mathbf{x}_j^{\dagger}$ から確率 $\mathbf{p}_j$ にしたがうように $\mathbf{y}_j$ を乱択する.
    for $o = 1 \rightarrow 2^{|n_j|}$ **do**
      $\mathbf{x}_j^{o} \leftarrow \mathbf{k}_j^{o} \cdot B_j$
      $\lambda_{\mathbf{k}_j^{o}} \leftarrow \sum_{i=1}^{L_j} \log_2(Pr[x_{j,i}^{o} \oplus y_{j,i}])$ ▷ $Pr[x_{j,i}^{o} \oplus y_{j,i} = 1] = p_{j,i}$.
      ビンの添字を計算 $b \leftarrow \lfloor \lambda_{\mathbf{k}_j^{o}} / l_{bin} \rfloor$.
      ビンに属する候補数を更新 $\mathsf{H}_j(b) \leftarrow \mathsf{H}_j(b) + 1$.
    end for
  end for
  $\mathsf{H}_{nl}(\lfloor -n_{nl} / l_{bin} \rfloor) \leftarrow 2^{n_{nl}}$
  $t_k \leftarrow \mathsf{HistRankEst}(\mathsf{H}_1, \mathsf{H}_2, ..., \mathsf{H}_{m_{key}}, \mathsf{H}_{nl}, b^{\dagger})$
  if $t_{adv} \geq t_k$ **then**
    $t_{success} \leftarrow t_{success} + 1$
  end if
end for
return $p_{success} = t_{success} / t_{end}$

に漏洩値と正しい鍵の両方を乱択する．なお，Algorithm 3.3 では試行ごとに正しい鍵のみ乱択している．

まず，Algorithm 4.1 と同様に線形方程式系を分割する．また，確率ベクトル $\mathbf{p} = (p_1, p_2, ..., p_L)$ (式 (4.3) 参照) を各副線形方程式系に対応するベクトル $\mathbf{p}_j$ に分割する．以降，以下の手順を t_{end} 回繰り返す．その際，毎回の試行において正しい鍵 $\mathbf{k}^\dagger = (\mathbf{k}_l^\dagger, \mathbf{k}_{nl}^\dagger)$ 及び測定した RHS 値ベクトルを乱択し変更する．

任意の $j \in \{1, 2, ..., m_{key}\}$ において正しい副鍵 $\mathbf{k}_j^\dagger$ を乱択し，正しい RHS 値ベクトル $\mathbf{x}_j^\dagger$ を $\mathbf{k}_j^\dagger$ から計算する．測定した RHS 値ベクトル $\mathbf{y}_j$ を確率 $\mathbf{p}_j$ に従うように乱択する．正しい鍵の対数尤度 $\lambda_{\mathbf{k}^\dagger}$ を計算し，$\lfloor \lambda_{\mathbf{k}_j^o} / l_{bin} \rfloor$ (l_{bin}:ビンの幅) を正しい鍵のビンの添字 $b^\dagger$ に加える．他の副鍵の候補に対しても同様にビンの添字を計算し，ヒストグラム H_j を逐次的に更新する．また，情報がない副鍵 $\mathbf{k}_{nl}$ に関しては，一様な対数尤度 $\log_2(2^{-n_{nl}}) = -n_{nl}$ を持つため，すべて同じビン $\lfloor -n_{nl}/l_{bin} \rfloor$ に入れる．

HistRankEst に上記で得られたヒストグラムと正しい鍵の対数尤度を入力してランク t_k を計算する．もし $t_{adv} \geq t_k$ であるならば，攻撃成功とみなし成功数のカウンター変数 $t_{success}$ を 1 増加させる．t_{end} 回の試行の後，成功率 $p_{success} = t_{sucess}/t_{end}$ を出力する．

Algorithm 4.2 の計算量は $t_{end} \cdot \left(t_{prep} + t_{re}\right)$ と見積もられる．ここで，t_{prep} は式 (4.9) で求まり，t_{re} はヒストグラムを用いたランク評価に必要な計算量 (付録 N 参照) である．

4.4 PRESENT への適用

先行研究 [30] との比較のため，PRESENT (付録 E，図 E.1，[8]) を想定し，第 4.3 節で示した SCCA-KE に対する耐性評価手法を実行する．PRESENT の鍵長は 80 ビットであり，長期的安全性 ($\mathsf{A}_{t \leq 2^{80}}$ に対する安全性) は示せないため，短期的安全性 ($\mathsf{A}_{t \leq 2^{60}}$ に対する安全性) のみ考慮して安全性を評価する．そのため，$t_{adv} = 2^{60}$

とする．

さらに，攻撃者が得られるデータ量の最大値を $q_{adv} = 2^{15}$ とし，計算量を踏まえて攻撃者は $\mathsf{A}_{t\leq 2^{60},q\leq 2^{15}}$ を仮定する．これは少量のデータの暗号化に使用されるスマートカード等のデバイスを想定している．$q = 2^{15}$ のデータ量は，平文データ 256[KByte] の暗号化処理において毎回漏洩値を測定する必要があり，このコストは仮定した暗号デバイスでは過大である．このように，現実よりも強力な攻撃者 $\mathsf{A}_{t\leq 2^{60},q\leq 2^{15}}$ を仮定することでスマートカード等の短期的安全性を十分に示せる．

第 4.4.1 項では PRESENT における Cube 探索の結果を示す．第 4.4.2 項では SCCA-KE に対する評価を行う．

4.4.1 Cube 探索結果

第 4.3.1 項で述べたとおり，Cube サイズを $|\mathcal{I}| \leq 6$ に限定して総当りによる Cube 探索を実行した．その際，以下の 3 つの漏洩モデルを考慮した．なお，PRESENT の r 段目出力 64 ビットを $(w_{r,1}, w_{r,2}, ..., w_{r,64})$ とする $(r \in \{1, 2, ..., r_{all}\})$.

(i) 単一ビットが漏洩 (SB モデル)[54]: 攻撃者は r 段目出力の 1 ビット $w_{r,i}$ $(i \in \{1, 2, ..., 64\})$ にアクセスできる．

(ii) 8 ビット中間値の HW の LSB が漏洩 (HW8 モデル)[30]: 攻撃者は r 段目の 2 個の S-box の出力 8 ビットの HW の LSB にアクセスできる．これは $\bigoplus_{j=1}^{8} w_{r,8\cdot(i-1)+j}$ $(i \in \{1, 2, ..., 8\})$ と表現される．なお，8 ビットレジスタを想定している．

(iii) 4 ビット中間値の HW の LSB が漏洩 (HW4 モデル): 攻撃者は r 段目の単一の S-box の出力 4 ビットの HW の LSB にアクセスできる．これは $\bigoplus_{j=1}^{4} w_{r,4\cdot(i-1)+j}$ $(i \in \{1, 2, ..., 16\})$ と表現される．なお，4 ビットレジスタを想定している．

LSB は中間値の XOR で表現される．つまり，HW のバイナリ値の中で最も低次数の多項式で表現が可能であり，これは攻撃者にとって有利な漏洩となる (サイズの小さい Cube が多数存在する)．任意の (r, i) (r 段目出力の i 番目のビット，各

漏洩モデルにおける添字 i は各漏洩モデルの定義に従う) について，$|\mathcal{I}| \leq 6$ なる添字集合 $\mathcal{I}$ について総当たりで Cube $\mathcal{C}_{\mathcal{I}}$ から線形方程式を得られるかを確認した (第 4.3.1 項参照).

ここで，攻撃者 $\mathsf{A}_{t\leq 2^{60},q\leq 2^{15}}$ を仮定しているので，$n_{nl} \geq 60$ の場合は攻撃が実行困難である．したがって，$n_{nl} < 60$ ($n_l \geq 20$) となる中間値アクセスについて，表 4.2，4.3 及び 4.4 に示す．なお，“$\max n_j$” 及び “$\max L_j$” は全ての副線形方程式系における鍵変数の数 n_j と式数 L_j の最大値を示している ($j \in \{1, 2, ..., m_{key}\}$)．これらが情報理論的評価及び実験的評価の計算が実行可能かを決定する (式 (4.9) 及び式 (4.13) 参照)．実際，$\max n_j \leq 16$ 及び $\max L_j \leq 2,220$ であるので，これらの計算は実行可能であり，これは第 4.2.2 項で示した観測 (Observation) に一致する．

4.4.2 評価結果

表 4.5 に評価に使用した漏洩値から得られる Cube の詳細を示す．なお，各漏洩モデルについて段数が異なる漏洩値を 2 種類ずつ使用する．

ここで，表 4.2，4.3 及び 4.4 で示した Cube をそのまま使用する場合，データ量が 2^{15} を超える場合がある．その場合，得られる RHS 値の反転確率 p_i が大きくなることから，サイズの大きい Cube から優先的に削減する．このような理由で評価に使用しない Cube について，表 4.5 では下線を引いて削減後の数字を示す．

さらに，Cube 探索結果ではデータ量が 2^{15} よりも小さい場合がある．その場合，同様の Cube を複数回使用し，線形方程式系の式数を増やす．式数が増えることで正しい RHS 値が推定しやすくなり，計算量が削減できる．簡単のため，探索された Cube 全体を t_{cube} 回ずつ使用する (表 4.5 右端の列参照)．その際，情報理論的評価の計算量 t_{LG} (式 (4.13) 参照) は増加しない．線形方程式系を t_{cube} 個ずつ用意することになるので，元の線形方程式数 L_j から $L_j \cdot t_{cube}$ に増加する．そのた

表 **4.2**　単一ビットが漏洩する場合の Cube 探索結果

r	i	各 Cube サイズの Cube 数						n_l	L	$\max n_j$	$\max L_j$
		1	2	3	4	5	6				
3	1	8	48	120	160	0	48	32	384	1	30
3	5	0	0	144	144	0	3456	32	3744	1	288
3	26	0	0	0	2880	0	0	32	2880	1	144
3	33	0	0	0	1920	0	0	32	1920	1	96
3	49	0	0	0	2880	0	0	32	2880	1	144
4	17	0	0	0	231	731	4813	51	5775	16	2209
4	49	0	0	0	240	731	4815	51	5786	16	2220

表 **4.3**　8 ビット中間値の HW の LSB が漏洩する場合の Cube 探索結果

r	i	各 Cube サイズの Cube 数						n_l	L	$\max n_j$	$\max L_j$
		1	2	3	4	5	6				
2	1	0	48	0	0	144	0	64	192	3	9
2	2	8	24	0	48	96	0	56	176	8	36
3	1	0	0	0	0	0	108	24	108	3	18
3	2	0	0	0	68	80	36	30	184	3	26

表 **4.4**　4 ビット中間値の HW の LSB が漏洩する場合の Cube 探索結果

r	i	各 Cube サイズの Cube 数						n_l	L	$\max n_j$	$\max L_j$
		1	2	3	4	5	6				
2	1	16	32	16	0	0	0	32	64	1	3
2	2	0	48	0	0	144	0	64	192	3	9
2	3	0	40	0	0	80	0	64	120	2	4
2	4	0	48	0	0	144	0	64	192	3	9
2	5	0	0	288	288	72	0	32	648	1	27
3	1	0	20	0	102	120	72	60	314	4	52
3	2	0	0	0	0	0	108	24	108	3	18
3	3	0	0	0	0	0	72	24	72	3	12
3	4	0	0	0	0	0	108	24	108	3	18
3	5	0	0	0	0	0	432	24	432	3	72
3	9	0	0	0	0	0	288	24	288	3	48
3	13	0	0	0	0	0	432	24	432	3	72

表 4.5　評価に使用した漏洩値から得られる Cube の詳細

漏洩モデル	r	i	各 Cube サイズの Cube 数						q	t_{cube}
			1	2	3	4	5	6		
SB モデル	3	1	8	48	120	160	720	48	$2^{14.86}$	1
	4	49	0	0	0	240	731	0	$2^{14.73}$	1
HW8 モデル	2	2	64	192	0	384	768	0	$2^{14.94}$	8
	3	2	0	0	0	340	400	180	$2^{14.86}$	5
HW4 モデル	2	1	1808	3616	1808	0	0	0	$2^{14.81}$	113
	3	1	0	60	0	306	360	216	$2^{14.89}$	3

め，式数増加後の LG は

$$LG = \frac{2^{n_{nl}}}{1 + n_{key}} \prod_{j=1}^{m_{key}} \left(2^{-n_j} \sum_{o=1}^{2^{n_j}} \sum_{\substack{o'=1 \\ o' \neq o}}^{2^{n_j}} \left(\prod_{\substack{i=1 \\ x_{j,i}^{o} \neq x_{j,i}^{o'}}}^{L_j} 2 \cdot p_{j,i}^{\frac{1}{2}} \cdot (1 - p_{j,i})^{\frac{1}{2}} \right)^{t_{cube}} + 1 \right) \tag{4.14}$$

と求まり，t_{LG} は増加しないことが分かる．このように，情報理論的評価はデータ量が増加しても同様の計算量で実行が可能である．しかし，ランク評価を用いた実験的評価 Algorithm 4.2 では，乱択される漏洩値で攻撃成功率が変化するため t_{prep} (式 (4.9) 参照) は t_{cube} 倍となる．そのため，仮定する q_{adv} が大きい場合にはランク評価を用いた実験的評価は実行困難になるため，情報理論的評価のみで耐性評価を行う必要がある．なお，表 4.5 で示した条件においては実験的評価についても実行可能である．

情報理論的評価及びランク評価を用いた実験的評価の結果を図 4.4 及び 4.5 にそれぞれ示す．後者においては，計算時間の制約から正しい鍵 $\mathbf{k}^{\dagger}$ 及び測定した RHS 値ベクトル $\mathbf{y}$ を変更しつつ 1,000 回試行を行った (Algorithm 4.2 において $t_{end} = 1,000$)．表 3.3 で示した環境において並列処理を行なわず，C++言語 (g++

バージョン 4.8) で実装した．その結果，表 4.5 で示した条件を全て網羅した評価を約 23 時間で終了した．

PRESENT における SCCA-KE に対する耐性評価結果から，以下の考察が得られる．

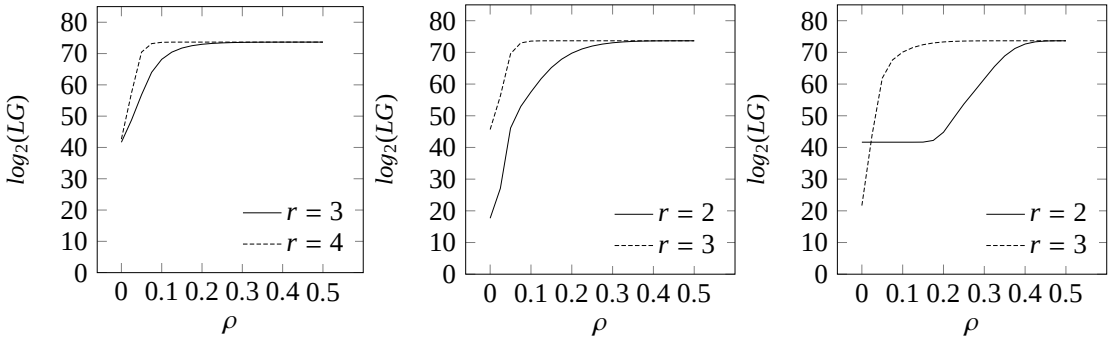
(i) PRESENT の 2 から 4 段目出力以外を使用する場合，情報がない鍵のビット数が 60 を超えるので攻撃者 $\mathsf{A}_{t\leq 2^{60},q\leq 2^{15}}$ にとって攻撃は実行困難である．したがって，2〜4 段目出力の漏洩を防止できれば SCCA-KE は防げる．

(ii) 図 4.4 及び 4.5 から，全ての条件で反転確率が $\rho \geq 0.4$ ならば攻撃成功率は 0 となる．したがって，このような反転確率を達成するような雑音の付加によって SCCA-KE は防ぐことができる．

また，SCCA-KE 及び SCCA-PM を適用した場合の攻撃効果の比較を図 4.6 に示す．その際の漏洩モデルは先行研究 [30] において用いられたものと同様である．これは，漏洩モデルは HW8 モデルであり，$(r, i) = (2, 1)$ なる 1 ビット $\bigoplus_{j=1}^{8} w_{2,j}$ に攻撃者がアクセスできると仮定している．なお，$\mathsf{A}_{t\leq 2^{60},q\leq 2^{15}}$ を仮定しており，$t_{cube} = 6$ 回ずつ探索された全 Cube を使用する．

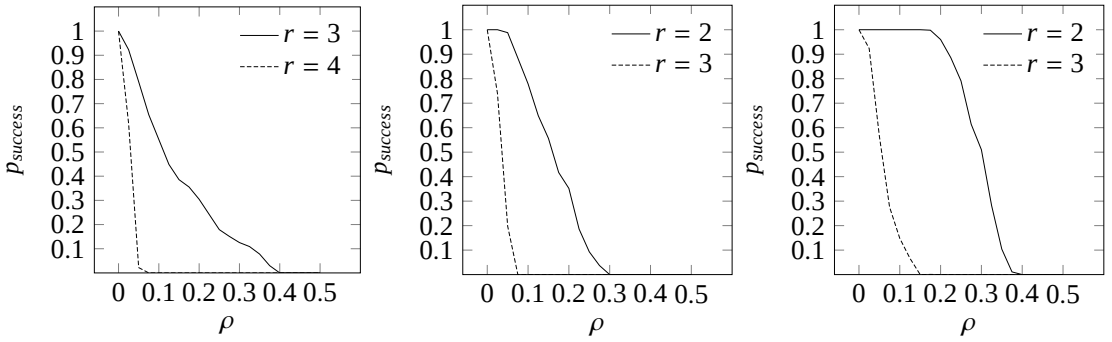
図 4.6 (a) より，SCCA-KE は SCCA-PM に比べて誤り耐性が高いことが分かる．なお，SCCA-PM では鍵列挙を行わないので推測エントロピーは定義できないため，参考として SCCA-KE の *LG* のみを図 4.6 (b) に示す．

参考のため，SCCA-PM の実験方法について示す．まず，最終的には正しい鍵と列 $\mathcal{T}_{\mathbf{k}}$ ($|\mathcal{T}_{\mathbf{k}}| = t_{adv} = 2^{60}$) の最後の鍵候補のハミング距離を計算する．もし前者が後者よりも小さければ，この攻撃は成功とみなすことができる．先行研究 [30] と同様の方法で，64 個の鍵変数を有する系を 4 個の副線形方程式系に分割する．SCCA-PM は各副線形方程式系から得られる副列はすべて同じ要素数となるように生成する．そのため，これらの副列の最後の要素となる副鍵の候補のハミング距離を求め，全副列に対して総和を計算することで列 $\mathcal{T}_{\mathbf{k}}$ の最後の鍵候補のハミ



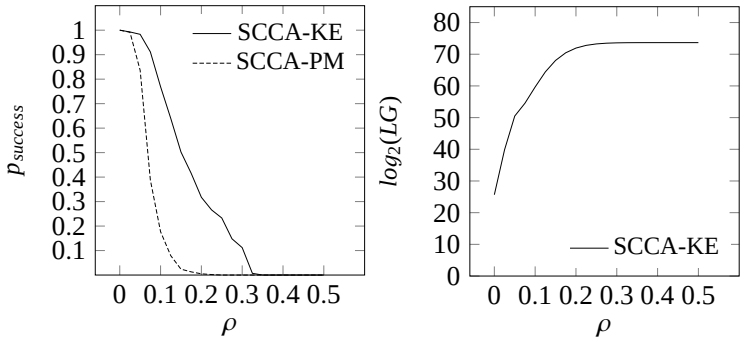
(a) SB モデル　　(b) HW8 モデル　　(c) HW4 モデル

図 4.4　PRESENT に対する SCCA-KE の推測エントロピーの下界



(a) SB モデル　　(b) HW8 モデル　　(c) HW4 モデル

図 4.5　PRESENT に対する SCCA-KE の攻撃成功率



(a) 攻撃成功率の比較　　(b) 推測エントロピー

図 4.6　HW8 モデルにおける $(r, i) = (2, 1)$ を想定した場合の SCCA-KE 及び SCCA-PM の攻撃成功率及び SCCA-KE の推測エントロピーの下界.

ング距離が求まる．

4.5　第 4 章のまとめ

実装の安全性評価として，1AL-BSC モデルにおける形式的評価を提案した．まず，SCCA-PM を改良して SCCA-KE を提案した．SCCA-KE は与えられた漏洩値に対して最適な列を生成して鍵を列挙する．次に，SCCA-KE に対する耐性評価手法を提案した．PRESENT に適用し，$\mathsf{A}_{t\leq 2^{60},q\leq 2^{15}}$ を仮定して耐性評価行った．その結果，2 から 4 段目出力の漏洩防止，または，反転確率が 0.4 以上で SCCA-KE は防げることを示した．

この方法で 1AL-BSC モデルにおける形式的評価を行った場合，漏洩防止によって攻撃を防ぐことができる中間値の段数等が判明する．また，もしそのような中間値を与えたとしても SCCA-KE を防ぐのに十分な反転確率 ρ が得られ，漏洩防止と雑音の付加の基準となる．

第5章

結論

5.1 研究の成果

情報インフラの拡大にともない，情報セキュリティの基礎となる暗号の重要性が高まっている．また，暗号はセンサーネットワークのような新たな分野においても導入が進んでいる．用途の多様化に伴い，暗号の評価はより多角的な視点で実施される必要性が生じている．本論文で注目したブロック暗号についても，軽量性を重視する傾向が強くなり，安全性とのバランスを今まで以上に考慮する必要がある．本論文では，ブロック暗号のアルゴリズム及びその実装の安全性について，未知の攻撃を視野にいれた評価手法を研究した．その中で，暗号化関数が持つ特性に注目した選択平文攻撃及びサイドチャネル攻撃に対する耐性評価手法を提案した．

アルゴリズムの安全性評価として，Integral 攻撃における IPS-BP を提案した．IPS-BP は高速に Integral 特性を探索する．Integral 特性が成立する段数は Integral 攻撃の攻撃可能段数を決定するため，ブロック暗号の段数設定において重要なパラメータとなる．IPS-BP は多項式時間で高速に実行が可能であり，実行時間が大きい IPS-DP のかわりに使用すべき場面が存在する．

実装の安全性評価として，RL モデル及び 1AL モデルを用いた DBA-KE 及び SCCA-KE に対する耐性評価手法を提案した．それぞれの漏洩モデルにおいて，データ量等のパラメータと推測エントロピーの下界及び攻撃成功率の関係が判明する．これにより，DBA-KE または SCCA-KE を防ぐのに十分なパラメータが得られる．サイドチャネル攻撃に対する対策を行う際，得られたパラメータを基準

とすることで未知攻撃への対策となる．

5.2 今後の課題

暗号研究の大きな課題として，暗号開発の包括的なフレームワークの確立がある．安全性に関する要件を設定したとき，これを満たすアルゴリズムとその実装を開発可能とするフレームワークが望まれる．ブラックボックス仮定における安全性の評価については，長年の研究の中で安全性の基準となる差分攻撃や Integral 攻撃といった強力な攻撃法が示された．これら攻撃法における攻撃可能段数を網羅的に導出することがフレームワークとして確立されつつある．しかし，攻撃可能段数を導出する手順が全ての攻撃法において確立されているわけではない．本論文では IPS-BP を提案したが，このように効率的なアルゴリズムが増えることでフレームワークの可用性が高まる．

サイドチャネル攻撃耐性評価においても，上記に類似したフレームワークが確立されることが望ましい．しかし，ブラックボックス仮定のような明確な攻撃条件を設定することができない．形式的評価では漏洩情報のモデル化により攻撃条件を明確化するが，サイドチャネル攻撃全般に通用する攻撃条件は得られない．そのため，多くの実装法を網羅するには複数の漏洩モデルを必要とする．本論文では RL モデル及び 1AL モデルにおける耐性評価手法を提案したが，フレームワークを確立するにはさらに多くの漏洩モデルが必要である．

謝　　辞

本論文は，防衛大学校理工学研究科後期課程に在籍中の研究成果をまとめたものです．防衛大学校情報工学科田中秀磨准教授にはこのような機会を与えていただくとともに，指導教官として研究の進め方や論文の執筆方法など多くの御指導をいただきました．厚くお礼申し上げます．また，以前の所属であるコンピュータ工学研究室及び現所属のサイバーセキュリティ研究室において公私ともに多くの助言をいただきました黒川恭一教授，三村守准教授及び岩井啓輔講師に心より感謝申し上げます．さらには，本論文の審査にあたり，御多忙中にも関わらず御指導いただきました横浜国立大学大学院環境情報研究院四方順司教授並びに防衛大学校情報工学科中村康弘教授に深く感謝いたします．

本論文はまた，防衛省海上自衛隊が実施する部外委託教育の研究成果でもあります．理工学研究科前期課程と後期課程を合わせて 4 年間という長期にわたる研修を行うため，各種調整をしていただいた関係各位に厚く御礼申し上げます．

同じ研究室で共に学んだ朴駿漢海軍少佐，渡部匡 1 等海尉，チメドツェレンエンフボルド陸軍中尉，田畠佑紀技官，小手川誠 1 等海尉，籠谷健吾 1 等陸尉，柘植裕介 1 等海尉，石丸歩 2 等陸尉，田山俊介 2 等陸尉， 今井淳太 2 等陸尉，三浦紘弥 2 等陸尉，ボルド・ムンフバートル陸軍中尉及びグェン・ドク・ソン陸軍中尉に感謝致します．

最後に，研究に集中できるよう，心身ともに支えてくれた妻に心から感謝します．

平成２９年１２月

参 考 文 献

[1] Abdul-Latip, S.F., Reyhanitabar, M.R., Susilo, W., Seberry, J.: On the security of NOEKEON against side channel cube attacks. In: International Conference on Information Security Practice and Experience. pp. 45–55. Springer (2010)

[2] Arikan, E.: An inequality on guessing and its application to sequential decoding. IEEE Transactions on Information Theory 42(1), 99–105 (1996)

[3] Arora, M.: How secure is aes against brute force attacks? https://www.eetimes.com/document.asp?doc_id=1279619, Accessed: 2017-10-1

[4] Bellare, M., Coppersmith, D., Hastad, J., Kiwi, M., Sudan, M.: Linearity testing in characteristic two. IEEE Transactions on Information Theory 42(6), 1781–1795 (1996)

[5] Bernstein, D.J., Lange, T., van Vredendaal, C.: Tighter, faster, simpler side-channel security evaluations beyond computing power. IACR Cryptology ePrint Archive 2015, 221 (2015)

[6] Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 12–23. Springer (1999)

[7] Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. Journal of CRYPTOLOGY 4(1), 3–72 (1991)

[8] Bogdanov, A., Knudsen, L., Leander, G., Paar, C., Poschmann, A., Robshaw, M., Seurin, Y., Vikkelsoe, C.: PRESENT: An ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2007, vol. 4727, pp. 450–466. Springer (2007), http://dx.doi.org/10.1007/978-3-540-74735-2_31

[9] Bogdanov, A., Isobe, T.: How secure is AES under leakage. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 361–385. Springer (2014)

[10] Bogdanov, A., Kizhvatov, I., Manzoor, K., Tischhauser, E., Witteman, M.: Fast and memory-efficient key recovery in side-channel attacks. In: International Conference on Selected Areas in Cryptography. pp. 310–327. Springer (2015)

[11] Carlet, C.: Boolean functions for cryptography and error correcting codes. Boolean models and methods in mathematics, computer science, and engineering 2, 257–397 (2010)

[12] Daemen, J., Rijmen, V.: The design of Rijndael: AES - the advanced encryption standard. Springer Verlag, Berlin, Heidelberg, New York (2002)

[13] David, L., Wool, A.: A bounded-space near-optimal key enumeration algorithm for multi-dimensional side-channel attacks. IACR Cryptology ePrint Archive 2015, 1236 (2015)

[14] Dinur, I., Shamir, A.: Cube attacks on tweakable black box polynomials. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 278–299. Springer (2009)

[15] Dinur, I., Shamir, A.: Side channel cube attacks on block ciphers. IACR Cryptology ePrint Archive 2009, 127 (2009)

[16] Duc, A., Faust, S., Standaert, F.X.: Making masking security proofs concrete. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 401–429. Springer (2015)

[17] Ferguson, N., Kelsey, J., Lucks, S., Schneier, B., Stay, M., Wagner, D., Whiting, D.: Improved cryptanalysis of Rijndael. In: Goos, G., Hartmanis, J., van

Leeuwen, J., Schneier, B. (eds.) Fast Software Encryption, vol. 1978, pp. 213–230. Springer (2001), http://dx.doi.org/10.1007/3-540-44706-7_15

[18] Fluhrer, S.R., McGrew, D.A.: Statistical analysis of the alleged RC4 keystream generator. In: International Workshop on Fast Software Encryption. pp. 19–30. Springer (2000)

[19] Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic analysis: Concrete results. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 251–261. Springer (2001)

[20] Glowacz, C., Grosso, V., Poussier, R., Schueth, J., Standaert, F.X.: Simpler and more efficient rank estimation for side-channel security assessment. In: International Workshop on Fast Software Encryption. pp. 117–129. Springer (2015)

[21] intel: Intel core i9-7960x processor (2017), https://www.intel.co.jp/content/www/jp/ja/products/processors/core/x-series/i9-7960x.html, Accessed: 2017-10-1

[22] Ishai, Y., Sahai, A., Wagner, D.: Private circuits: Securing hardware against probing attacks. In: Annual International Cryptology Conference. pp. 463–481. Springer (2003)

[23] Islam, S., Afzal, M., Rashdi, A.: On the security of LBlock against the cube attack and side channel cube attack. In: International Conference on Availability, Reliability, and Security. pp. 105–121. Springer (2013)

[24] ISO/IEC 29192-2: Information technology -security techniques - lightweight cryptography -part 2: Block ciphers (2012)

[25] Knudsen, L., Wagner, D.: Integral cryptanalysis. In: International Workshop on Fast Software Encryption. pp. 112–127. Springer (2002)

[26] Knudsen, L.: CLEFIA. In: van Tilborg, H., Jajodia, S. (eds.) Encyclopedia of Cryptography and Security, pp. 210–211. Springer US (2011), http://dx.doi.org/10.1007/978-1-4419-5906-5_561

[27] Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Annual International Cryptology Conference. pp. 388–397. Springer (1999)

[28] Köpf, B., Basin, D.: An information-theoretic model for adaptive side-channel attacks. In: Proceedings of the 14th ACM conference on Computer and communications security. pp. 286–296. ACM (2007)

[29] Kullback, S., Leibler, R.A.: On information and sufficiency. The annals of mathematical statistics 22(1), 79–86 (1951)

[30] Li, Z., Zhang, B., Fan, J., Verbauwhede, I.: A new model for error-tolerant side-channel cube attacks. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 453–470. Springer (2013)

[31] MacKay, D.J.: Information theory, inference and learning algorithms. Cambridge university press (2003)

[32] Mantin, I.: Predicting and distinguishing attacks on RC4 keystream generator. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 491–506. Springer (2005)

[33] Manzoor, K., et al.: Efficient practical key recovery for side-channel attacks. Master's thesis, Aalto University, June 2014. http://cse. aalto. fi/en/personnel/antti-yla-jaaski/msc-thesis/2014-msc-kamran-manzoor. pdf (2014)

[34] Martin, D.P., O ' connell, J.F., Oswald, E., Stam, M.: Counting keys in parallel after a side channel attack. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 313–337. Springer (2014)

[35] Massey, J.L.: Guessing and entropy. In: Information Theory, 1994. Proceedings., 1994 IEEE International Symposium on. p. 204. IEEE (1994)

[36] Matsui, M.: Linear cryptanalysis method for DES cipher. In: Workshop on the Theory and Application of of Cryptographic Techniques. pp. 386–397. Springer (1993)

[37] Poussier, R., Standaert, F.X., Grosso, V.: Simple key enumeration (and rank estimation) using histograms: An integrated approach. In: International Conference on Cryptographic Hardware and Embedded Systems. pp. 61–81. Springer (2016)

[38] Renauld, M., Standaert, F.X.: Algebraic side-channel attacks. In: International Conference on Information Security and Cryptology. pp. 393–410. Springer (2009)

[39] Sasaki, Y., Wang, L.: Comprehensive study of integral analysis on 22-round LBlock. In: Kwon, T., Lee, M.K., Kwon, D. (eds.) Information Security and Cryptology, vol. 7839, pp. 156–169. Springer (2013), http://dx.doi.org/10.1007/978-3-642-37682-5_12

[40] Schramm, K., Wollinger, T., Paar, C.: A new class of collision attacks and its application to DES. In: International Workshop on Fast Software Encryption. pp. 206–222. Springer (2003)

[41] Strohmaier, E., Dongarra, J., Simon, H., Meuer, M.: Performance development (2017), https://www.top500.org/statistics/perfdevel/, Accessed: 2017-10-1

[42] Strohmaier, E., Dongarra, J., Simon, H., Meuer, M.: Top 10 sites for june 2017 (2017), https://www.top500.org/lists/2017/06/, Accessed: 2017-10-1

[43] Suzaki, T., Minematsu, K., Morioka, S., Kobayashi, E.: TWINE: A lightweight block cipher for multiple platforms. In: Knudsen, L., Wu, H. (eds.) Selected

Areas in Cryptography, vol. 7707, pp. 339–354. Springer (2013), http://dx.doi.org/10.1007/978-3-642-35999-6_22

[44] Todo, Y.: Structural evaluation by generalized integral property. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015, vol. 9056, pp. 287–314. Springer (2015), http://dx.doi.org/10.1007/978-3-662-46800-5_12

[45] Todo, Y., Aoki, K.: FFT key recovery for integral attack. In: International Conference on Cryptology and Network Security. pp. 64–81. Springer (2014)

[46] Todo, Y., Morii, M.: Bit-based division property and application to SIMON family. In: International Conference on Fast Software Encryption. pp. 357–377. Springer (2016)

[47] Tsunoo, Y., Saito, T., Suzaki, T., Shigeri, M., Miyauchi, H.: Cryptanalysis of DES implemented on computers with cache. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 62–76. Springer (2003)

[48] Veyrat-Charvillon, N., Gérard, B., Renauld, M., Standaert, F.X.: An optimal key enumeration algorithm and its application to side-channel attacks. In: International Conference on Selected Areas in Cryptography. pp. 390–406. Springer (2012)

[49] Veyrat-Charvillon, N., Gérard, B., Standaert, F.X.: Security evaluations beyond computing power. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 126–141. Springer (2013)

[50] Wald, A.: Sequential tests of statistical hypotheses. In: Breakthroughs in Statistics, pp. 256–298. Springer (1992)

[51] Williams, H.P.: Logic and integer programming, vol. 130. Springer (2009)

[52] Wu, W., Zhang, L.: LBlock: A lightweight block cipher. In: Lopez, J., Tsudik, G. (eds.) Applied Cryptography and Network Security, vol. 6715, pp. 327–344. Springer (2011), http://dx.doi.org/10.1007/978-3-642-21554-4_19

[53] Xiang, Z., Zhang, W., Bao, Z., Lin, D.: Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers. In: Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22. pp. 648–678. Springer (2016)

[54] Yang, L., Wang, M., Qiao, S.: Side channel cube attack on PRESENT. In: International Conference on Cryptology and Network Security. pp. 379–391. Springer (2009)

[55] Zheng, Y., Matsumoto, T., Imai, H.: On the construction of block ciphers provably secure and not relying on any unproved hypotheses. In: Brassard, G. (ed.) Advances in Cryptology - CRYPTO'89 Proceedings, vol. 435, pp. 461–480. Springer New York (1990), http://dx.doi.org/10.1007/0-387-34805-0_42

[56] 金子敏信: 共通鍵暗号の安全性評価. 電子情報通信学会基礎・境界ソサイエティ Fundamentals Review 7(1), 14–29 (2013)

付　　録

A　用語の定義

ブロック暗号 共通鍵暗号に分類され，データを一定の大きさに区切って暗号化する．鍵拡大関数と暗号化関数で構成される．

鍵拡大関数 鍵を入力とし，段鍵を出力する．

暗号化関数 平文と段鍵を入力し，暗号文を出力する関数．段関数を繰り返すことで構成される．

平文 暗号化関数に鍵とともに入力されるデータ．平文のビット長を平文長という．

鍵 送信側と受信側が共有している固定長の情報．鍵のビット長を鍵長という．

段鍵 鍵拡大関数で鍵から生成される固定長 (鍵長の定数倍) の秘密情報．一般的に各段関数で等ビット長の段鍵が使用される．

段関数 中間値と段鍵を入力し，中間値を出力する関数 (1 段目の入力は平文に，最終段の出力は暗号文となる).

段数 段関数を処理した回数.

***r* 段目出力** r 回段関数を処理した出力

副鍵 鍵を分割したもの.

ワード 複数ビットをまとめたもので，暗号化関数や段関数における演算の単位．ワードのビット長をワード長という．

中間値 段関数の出力で，次の段関数の入力となる (段鍵は中間値に含まない).

選択平文 攻撃者が任意に選択し，暗号化関数に入力する平文．選択平文の集合を選択平文集合という．

変数 選択平文集合を入力し，暗号化処理を複数回実行する際，平文 (選択平文集合の要素) によって変化する値．変数であるビット及びワードを変数ビット及び変数ワードという．

定数 選択平文集合を入力し，暗号化処理を複数回実行する際，平文 (選択平文集合の要素) によって変化しない値．定数であるビット及びワードを定数ビット及び定数ワードという．

多重集合 集合に同じ値をとる要素が複数含まれる集合．

列 含まれる要素の順序が定義される多重集合．

計算量 攻撃に必要な暗号化関数の処理回数．

データ量 選択平文攻撃においては，攻撃者が得られる平文と暗号文の組数 (選択平文集合の要素数)．サイドチャネル攻撃においては，漏洩値を測定した回数．

正しい鍵 攻撃対象である暗号化関数で使用されている鍵．攻撃者は正しい鍵を推定することを攻撃の目的とする．

ブール関数 $\mathbb{F}_2$ 上で定義される関数 ($\mathbb{F}_2^n \to \mathbb{F}_2$)．本論文では Algebraic Normal Form (ANF) で AND 及び XOR による多変数多項式として表現する．

S-box 非線形な置換関数．本論文で対象とするブロック暗号は全て S-box を使用する．

B 記号の定義

ボールド体 x (小文字アルファベット)：ベクトルを意味する．複数ビットは各ビットの値を成分とするベクトルで表現する．

カリグラフィー体 $\mathcal{X}$ (大文字アルファベット)：集合，多重集合及び列を意味する．集合は $\{x|x \in \mathcal{X}\}$ のように要素と要素が満たすべき条件を示す．

サンセリフ体 f, F：関数を意味する．

$|\mathcal{X}|$：集合，多重集合または列の要素数

$\mathcal{X} \times \mathcal{Y}$：集合 $\mathcal{X}$ 及び $\mathcal{Y}$ の直積集合

$\mathcal{X} \cap \mathcal{Y}$: 集合 $\mathcal{X}$ 及び $\mathcal{Y}$ の共通集合

$\mathcal{X} \cup \mathcal{Y}$: 集合 $\mathcal{X}$ 及び $\mathcal{Y}$ の和集合

$\mathcal{X} \backslash \mathcal{Y}$: 集合 $\mathcal{X}$ から $\mathcal{Y}$ を引いた差集合

$(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m)$: 括弧内の全ベクトルの成分を成分とするベクトル

$\mathbb{F}_2^n$: 有限体 $\mathbb{F}_2 (= \{0, 1\})$ の n 次元のベクトル

$+$: 実数の加算.

$\oplus$: $\mathbb{F}_2$ または $\mathbb{F}_2^n$ 上の XOR(排他的論理和). $\mathbb{F}_2^n$ 上ではベクトルの各成分で XOR を行う.

$\sum$: 実数の総和.

$\bigoplus$: XOR の総和. 総和 $\sum$ の加算 $+$ を XOR $\oplus$ に置き換える.

$\cdot$: 一般的には実数の積を示すが, ブール関数内では AND(論理積) を示す. 自明な場合は省略する.

$\prod$: 一般的には実数の総乗を示すが, ブール関数内では AND の総乗を示す.

$\mathbf{v}$: 平文

$\mathbf{k}$: 鍵

n_{all} : 平文長

n_{key} : 鍵長

n_{ward} : ワード長

n_{var} : 変数ビット数

n_{const} : 定数ビット数

m_{all} : 平文に含まれるワード数

m_{key} : 鍵に含まれる副鍵数

r_{all} : ブロック暗号の仕様として規定されている段数

t：計算量

q：データ量

$\mathcal{I}$：ビットの添字集合．同じ段のビットを示す場合ビットの添字 i を要素とし，複数段のビットを示す場合段数 r を加えて (r, i) を要素とする．

$\mathcal{J}$：ワードの添字集合．同じ段のワードを示す場合ワードの添字 i を要素とし，複数段のワードを示す場合段数 r を加えて (r, i) を要素とする．

$\mathbf{w}_r$：r 段目出力である中間値．暗号文は $\mathbf{w}_{r_{all}}$ とする．

$\mathbf{v}_i$：平文の i 番目のワード

$\mathbf{k}_i$：i 番目の副鍵

$\mathbf{w}_{r,i}$：r 段目出力である中間値の i 番目のワード

$\mathbf{w}_{\mathcal{J}}$：添字集合 $\mathcal{J}$ に含まれる添字で示される全中間値のビットを成分とする中間値

$\mathcal{V}_{\mathcal{I}}$：変数ビットの添字集合が $\mathcal{I}$ である選択平文集合

$\mathcal{V}_{\mathcal{J}}$：変数ワードの添字集合が $\mathcal{I}$ である選択平文集合

$\mathcal{W}_r$：r 段目出力である中間値多重集合

$\mathcal{W}_{r,i}$：r 段目出力である中間値の i 番目のワードの多重集合

$\mathcal{W}_{\mathcal{J}}$：添字集合 $\mathcal{J}$ に含まれる添字で示される全中間値のビットを成分とする中間値の多重集合

$\mathbf{k}^o$：鍵 $\mathbf{k}$ の候補

$\lambda_{\mathbf{k}^o}$：鍵 $\mathbf{k}$ の候補の尤度，尤度比または対数尤度

$\mathcal{T}_{\mathbf{k}}$：鍵 $\mathbf{k}$ の候補の集合 $(\mathbf{k}^o \in \mathcal{T}_{\mathbf{k}})$

$\mathcal{U}_{\mathbf{k}}$：鍵 $\mathbf{k}$ の候補とその (対数) 尤度を要素とする集合 $((\mathbf{k}^o, \lambda_{\mathbf{k}^o}) \in \mathcal{U}_{\mathbf{k}})$

C　ブロック暗号

ブロック暗号の概要を図 D.1 に示す．鍵拡大関数において鍵 $\mathbf{k}$ から段鍵 $\mathbf{rk}_r$ ($r \in \{1, 2, ..., r_{all}\}$) を出力する．段鍵を使用し，暗号化関数では平文 $\mathbf{v}$ から暗号文 $\mathbf{w}_{r_{all}}$ を出力する．暗号化関数の概要を図 D.2 に示す．各段関数の出力を中間値とよび $\mathbf{w}_r$ と表記する．$\mathbf{w}_r$ は r 段目の段関数の出力であり，かつ，$r+1$ 段目の入力となる．平文を段関数で r_{all} 回繰り返し処理することで暗号文が出力される．段関数は段鍵 $\mathbf{rk}_r$ が未知であるならば出力が推定できないように構成される．なお，暗号化関数の構造は Feistel 構造と SPN 構造に大別される．

D　AES[12]

AES (Advanced Encryption Standard) は NIST (アメリカ国立標準技術研究所) で採択された標準暗号であり，現在最も普及しているブロック暗号である．AES は平文長 128 ビットで SPN 構造を持つ段関数で構成される．鍵長は 128，192 及び 256 ビットであり，それぞれの鍵長に対し段数は 10，12 及び 14 段となっている．鍵長 n_{key} の AES を AES-n_{key} と表現する．図 D.3 に AES の暗号化関数を示す．AES の全演算は図 D.4 に示す 8 ビットのワード $\mathbf{x}_i$ ($i \in \{1, 2, ..., 16\}$) を 1 単位として処理される．平文 $\mathbf{v}$，中間値 $\mathbf{w}_r$，秘密鍵 $\mathbf{k}$ 及び段鍵 $\mathbf{rk}_r$ といった全ての変数が図 D.4 の表現に従う．AES の段関数は，以下の 4 関数から構成される (最後の r_{all} 段目においては，MixColumns の処理は行わない)．なお，各関数の入力を $\mathbf{x}$，出力を $\mathbf{y}$ とする ($\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^{128}$).

AddRoundkey (AR): 鍵スケジュール部において生成された r 段目の段鍵 $\mathbf{rk}_{r,i}$ ($r \in \{1, 2, ..., r_{all}\}, i \in \{1, 2, ..., 16\}$) との XOR $\mathbf{y}_i = \mathbf{x}_i \oplus \mathbf{rk}_{r,i}$ を行う ($i \in \{1, 2, ..., 16\}$).

SubBytes (SB): 全ワードに対し，それぞれ非線形な置換関数 (S-box) $\mathsf{S} : \mathbb{F}_2^8 \to \mathbb{F}_2^8$ の処理 $\mathbf{y}_i = \mathsf{S}(\mathbf{x}_i)$ を行う ($i \in \{1, 2, ..., 16\}$)．なお，S は全単射である．

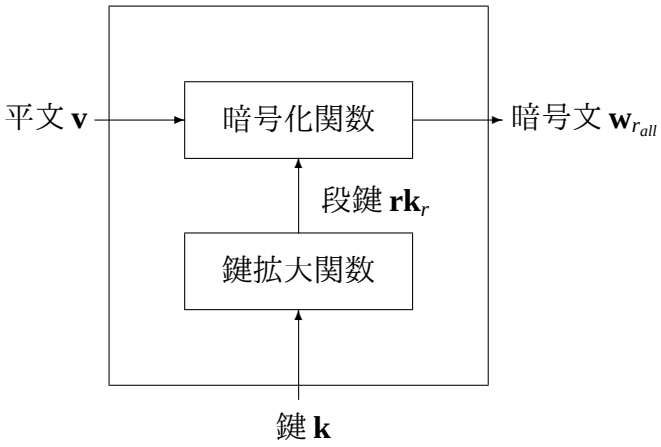


図 D.1　ブロック暗号の概要

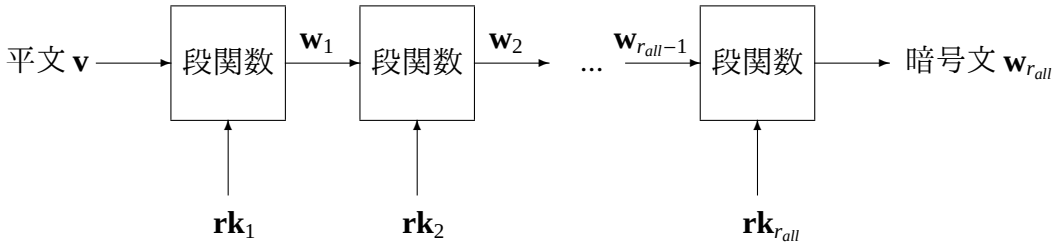


図 D.2　暗号化関数の概要

ShiftRows (SR): 図 D.4 の各行に対して循環シフトを行う．具体的には，i 行目は $(i-1)$ ワード分だけ左に循環シフトを行う．つまり，

$$\begin{aligned}
&\mathbf{y}_1 = \mathbf{x}_1, && \mathbf{y}_5 = \mathbf{x}_5, && \mathbf{y}_9 = \mathbf{x}_9, && \mathbf{y}_{13} = \mathbf{x}_{13}\\
&\mathbf{y}_2 = \mathbf{x}_6, && \mathbf{y}_6 = \mathbf{x}_{10}, && \mathbf{y}_{10} = \mathbf{x}_{14}, && \mathbf{y}_{14} = \mathbf{x}_2\\
&\mathbf{y}_3 = \mathbf{x}_{11}, && \mathbf{y}_7 = \mathbf{x}_{15}, && \mathbf{y}_{11} = \mathbf{x}_3, && \mathbf{y}_{15} = \mathbf{x}_7\\
&\mathbf{y}_4 = \mathbf{x}_{16}, && \mathbf{y}_8 = \mathbf{x}_4, && \mathbf{y}_{12} = \mathbf{x}_8, && \mathbf{y}_{16} = \mathbf{x}_{12}
\end{aligned} \tag{D.1}$$

という処理がなされる．

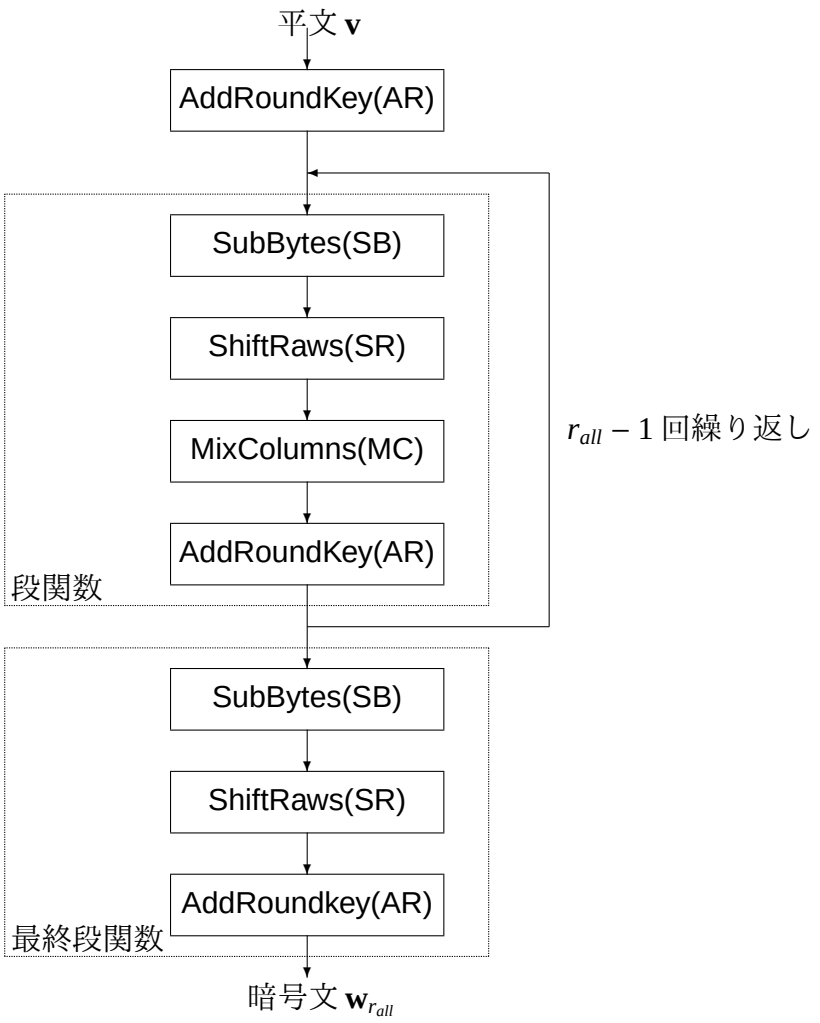


図 D.3　AES の暗号化関数の構成

$\mathbf{x}_1$	$\mathbf{x}_5$	$\mathbf{x}_9$	$\mathbf{x}_{13}$
$\mathbf{x}_2$	$\mathbf{x}_6$	$\mathbf{x}_{10}$	$\mathbf{x}_{14}$
$\mathbf{x}_3$	$\mathbf{x}_7$	$\mathbf{x}_{11}$	$\mathbf{x}_{15}$
$\mathbf{x}_4$	$\mathbf{x}_8$	$\mathbf{x}_{12}$	$\mathbf{x}_{16}$

図 D.4　AES の平文，段鍵または中間値の表現法.

MixColumns (MC): MC 関数内では変数に対する積を行う．その際，各ワードの値は $\mathbb{F}_2$ 上の多項式 $\mathbb{F}_2[X]/(X^8+X^4+X^3+X+1)$ として扱われる．なお，簡単のため多

項式はベクトル $\mathbb{F}_2^8$ で表現する．関数内では各ワードに対して $(0,0,0,0,0,0,1,0)$ または $(0,0,0,0,0,0,1,1)$ の積を計算する．これらのベクトルをそれぞれ 02 及び 03，$(0,0,0,0,0,0,0,1)$ を 01 と表現する．その際のワードの値 $\mathbf{x} = (x_8, x_7, .., x_1)$ に対する演算は

$$02 \cdot \mathbf{x} = \begin{cases} \mathbf{x} << 1 & x_8 = 0 \text{ の場合 (演算前)} \\ (\mathbf{x} << 1) \oplus (0,0,0,1,1,0,1,1) & \text{その他} \end{cases}$$
$$03 \cdot \mathbf{x} = \begin{cases} (\mathbf{x} << 1) \oplus \mathbf{x} & x_8 = 0 \text{ の場合 (演算前)} \\ (\mathbf{x} << 1) \oplus (0,0,0,1,1,0,1,1) \oplus \mathbf{x} & \text{その他} \end{cases} \quad \text{(D.2)}$$

となる (<< 1 は 1 ビットの左シフトを示す)．なお，$01 \cdot \mathbf{x} = \mathbf{x}$ である．MC 内の関数は以下に示す MDS 関数 (MDS 行列) を 4 つ並列に処理したものである ($i \in \{1,2,3,4\}$)．

$$\begin{pmatrix} \mathbf{y}_{4\cdot(i-1)+1} \\ \mathbf{y}_{4\cdot(i-1)+2} \\ \mathbf{y}_{4\cdot(i-1)+3} \\ \mathbf{y}_{4\cdot(i-1)+4} \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{x}_{4\cdot(i-1)+1} \\ \mathbf{x}_{4\cdot(i-1)+2} \\ \mathbf{x}_{4\cdot(i-1)+3} \\ \mathbf{x}_{4\cdot(i-1)+4} \end{pmatrix} \quad \text{(D.3)}$$

例えば，$\mathbf{y}_{4\cdot(i-1)+1} = 02 \cdot \mathbf{x}_{4\cdot(i-1)+1} \oplus 03 \cdot \mathbf{x}_{4\cdot(i-1)+2} \oplus 01 \cdot \mathbf{x}_{4\cdot(i-1)+3} \oplus 01 \cdot \mathbf{x}_{4\cdot(i-1)+4}$ のように，式 (D.2) を用いた積と XOR で演算を行う．なお，MDS 関数 ($\mathbb{F}_2^{32} \to \mathbb{F}_2^{32}$) は全単射である．

E PRESENT[8]

PRESENT は Bogdanov らが提案したブロック暗号である．計算リソースの限られたデバイスへの実装に適した軽量ブロック暗号として開発されており，CLEFIA[26] と並び ISO の軽量暗号の標準として採用されている (ISO/IEC 29192)[24].

PRESENT は SPN 構造を有する段関数を 31 段有し，鍵長 80 ビットと 128 ビッ

トの 2 つのバージョンが存在する．図 E.1 に PRESENT の段関数を示す．なお，$\mathbf{rk}_r \in \mathbb{F}_2^{64}$ ($r \in \{1, 2, ..., 32\}$) は段鍵を示す．PRESENT は段鍵の XOR，非線形層及び線形層で構成される．ただし，最終段 (31 段目) においては非線形層の前に $\mathbf{rk}_{31}$ が，線形層の後に $\mathbf{rk}_{32}$ が XOR される．非線形層は 16 個の $\mathbb{F}_2^4 \to \mathbb{F}_2^4$ なる全単射である S-box S で構成される．線形層はビット単位の置換を行うため，PRESENT はビット単位処理の段関数を持つとみなす．

F LBlock[52]

LBlock は Wu らが提案した軽量ブロック暗号であり，Feistel 構造の段関数を 31 段有し鍵長は 80 ビットである．図 F.1 に LBlock の段関数を示す．段関数では入力を 4 ビットずつの 16 ワードに分割して処理を行う．なお，$\mathbf{rk}_{r,i} \in \mathbb{F}_2^4$ ($r \in \{1, 2, ..., 31\}, i \in \{1, 2, ..., 8\}$) は段鍵を示す．$\mathbb{F}_2^4 \to \mathbb{F}_2^4$ なる 8 つの異なる全単射である S-box S_i ($i \in \{1, 2, ..., 8\}$) を有する．記号“<<< 8”は 8 ビット左巡回シフトである．

G TWINE[43]

TWINE は Suzaki らが提案した軽量ブロック暗号であり，Type-II 一般化 Feistel 構造 [55] の段関数を 36 段有し，鍵長 80 ビットと 128 ビットの 2 つのバージョンが存在する．図 G.1 に TWINE の段関数を示す．段関数では入力を 4 ビットずつの 16 ワードに分割して処理を行う．なお，$\mathbf{rk}_{r,i} \in \mathbb{F}_2^4$ ($r \in \{1, 2, ..., 36\}, i \in \{1, 2, ..., 8\}$) は段鍵を示す．$\mathbb{F}_2^4 \to \mathbb{F}_2^4$ なる全単射である S-box S を並列に処理する．

H 選択平文攻撃の仮定

選択平文攻撃は攻撃者が任意に選択した平文とそれに対応する暗号文を手に入れて鍵の推定を行う．ここで，計算量 (鍵の推定に必要な暗号化処理の回数) t とデータ量 (平文と暗号文組の数) q には制限がある．計算量は鍵の全数探索の計算

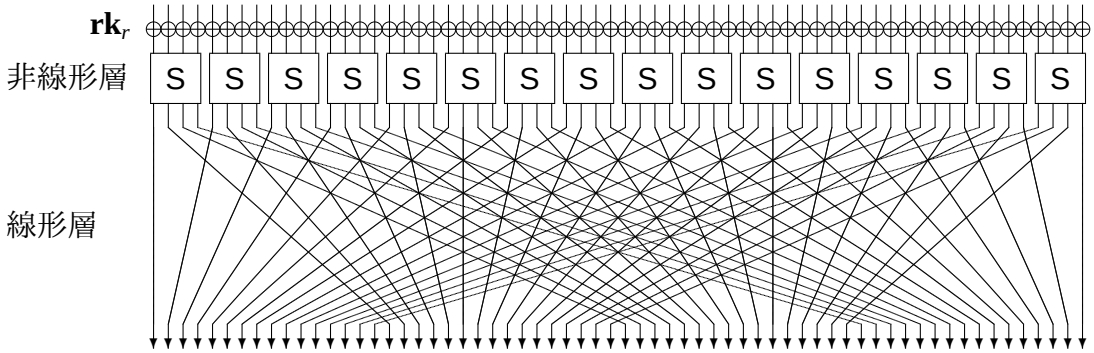


図 E.1　PRESENT の段関数

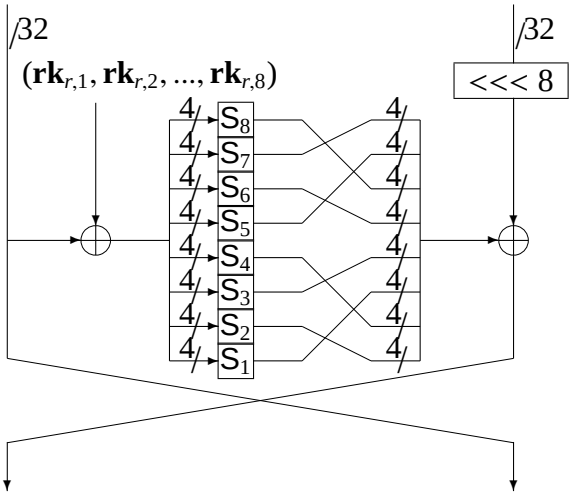


図 F.1　LBlock の段関数

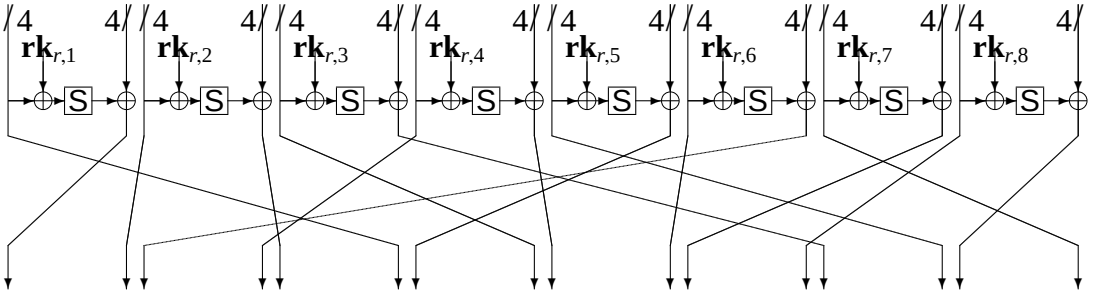


図 G.1　TWINE の段関数

量よりも小さく (ショートカット法の仮定)，データ量は全平文数より小さい必要がある (全平文と暗号文の対応が分かれば，鍵の推定は必要ない)．このような仮定において攻撃可能段数を見積もる．攻撃可能段数よりも多く段数設定すれば，鍵の全数探索が実行困難な攻撃者に対して安全となる．

攻撃者が選択する平文の集合を選択平文集合とよぶ．選択平文集合によって攻撃者が利用できる段関数等の脆弱性が異なる．なお，選択平文集合の要素数がデータ量となる．

平文 $\mathbf{v}$ において変数及び定数とするビット (変数ビット及び定数ビット) を設定する方法がある．これは Integral 攻撃 [25] や Cube 攻撃 [14] で一般的に採られる．変数ビット及び定数ビットを $\mathbf{v}_{var}$ 及び $\mathbf{v}_{const}$ とする．$\mathbf{v}_{var}$ は取りうる全ての値が入り，$\mathbf{v}_{const}$ は 1 つの値に固定される．例えば，右端の 2 ビットを変数ビットとした選択平文集合は $\{(0,0,...,0,0,0),(0,0,...,0,0,1),(0,0,...,0,1,0),(0,0,...,0,1,1)\}$ である．ただし，定数ビットは必ずしも 0 ではない．変数ビットの添字集合を $\mathcal{I} \subset \{1,2,...,n_{all}\}$ とする．変数ビットの数が $|\mathcal{I}| = n_{var}$ のとき，選択平文集合の要素数は $2^{n_{var}}$ となる．

変数をビット単位でなく，ワード単位で設定する方法がある．1 ワードが n_{ward} ビットで，平文が m_{all} ワードから成るとする ($n_{all} = m_{all} \cdot n_{ward}$)．変数ビットから成るワードを変数ワードとよび，その添字集合を $\mathcal{J} \subset \{1,2,...,m_{all}\}$ とする．同様に，定数ビットから成るワードを定数ワードとよぶ．

I　サイドチャネル攻撃の仮定

本論文ではサイドチャネル攻撃の形式的評価を行う．漏洩モデルは複数存在するが [15][22][38][40][49]，実装対象に応じてモデルを選択する．選択した漏洩モデルにおいて有効な攻撃手法を考慮し，耐性評価を行うことで攻撃が実行困難となるパラメータを導出する．このパラメータを基に実装を行うことで，未知のサイドチャネル攻撃への対策となる．

本論文では Bogdanov らが提案したランダム漏洩モデル及び Dinur らが提案した 1 ビットアクセス漏洩モデルに注目する [9][15]．ここで，選択平文攻撃とは異なるデータ量の定義を行う．形式的評価におけるデータ量を漏洩情報を測定した回数と定義する．つまり，漏洩情報を得るために複数回暗号化処理を行ったとしても，得られた漏洩情報が 1 個ならばデータ量は 1 のみ増加する．また，測定時に実行した暗号化処理の回数は計算量において考慮せず，鍵推定を行う際の処理のみ考慮する．

形式的評価では，攻撃者の仮定をパラメータとして攻撃成功率等を導出する．複数のパラメータが存在する場合は単一のパラメータに注目し，その他のパラメータは固定する．本論文では攻撃者の計算能力を限定し，攻撃者が t_{adv} 回の暗号化処理を行うことができると仮定する．1 ビットアクセス漏洩モデルではさらに得られるデータ量を限定し，最大で q_{adv} 回漏洩値を測定することができると仮定する．その場合の攻撃者をそれぞれ，$\mathsf{A}_{t \le t_{adv}}$ 及び $\mathsf{A}_{t \le t_{adv}, q \le q_{adv}}$ と表現する．

J　全単射の特性

全単射 $\mathbb{F}_2^n \to \mathbb{F}_2^n$ に $\mathbb{F}_2^n$ の任意の要素を入力すると，入力と同様に $\mathbb{F}_2^n$ の任意の要素が出力される．これを全単射の特性とよぶ．選択平文集合において，全単射の入力となる部分は変数，その他は定数とする (付録 H 参照)．これにより，全単射の出力となる中間値の集合は一意に定まる．

鍵 $\mathbf{k}$ が既知であれば復号が一意に行えるよう，暗号化関数は鍵 $\mathbf{k}$ が定数のとき全単射である．暗号化関数を構成する全単射として，Feistel 構造の関数 FS が一般的に使用される (式 (2.5) 参照)．関数 R の構成方法として，$\mathbf{y} = \mathsf{S}(\mathbf{x} \oplus \mathbf{k})$ のように $\mathbf{x}$ と $\mathbf{k}$ を XOR した後，全単射である S-box $\mathsf{S} : \mathbb{F}_2^{n_{ward}} \to \mathbb{F}_2^{n_{ward}}$ に入力する方法が方法が一般的である．このように関数 R を構成した場合，$\mathbf{k}$ が定数ならば $\mathbf{x} \mapsto \mathbf{y}$ は全単射であり，関数 FS も $\mathbb{F}_2^{2 \cdot n_{ward}} \to \mathbb{F}_2^{2 \cdot n_{ward}}$ なる全単射となる．

関数 FS の一部が全単射となる場合がある．$\mathbf{x}_1$ を定数とすると，関数 R の出力

も定数となる．すると，定数の XOR である $\mathbf{x}_2 \mapsto \mathbf{y}_1$ なる関数は全単射となる．この場合，FS は $\mathbb{F}_2^{n_{word}} \to \mathbb{F}_2^{n_{word}}$ なる全単射と恒等関数 $(\mathbf{x}_1 \mapsto \mathbf{y}_2)$ の並列処理となる．

K　ブール関数の特性

ブール関数を多項式として表現する方法として，Algebraic Normal Form (ANF) がある．ANF でブール関数 f の一般式を

$$\mathsf{f}(\mathbf{x}) = \bigoplus_{\mathcal{I} \in \mathcal{P}(\{1,2,\ldots,n\})} a_{\mathcal{I}} \prod_{i \in \mathcal{I}} x_i, \tag{K.1}$$

と表現する．ここで，$\mathcal{P}(\{1,2,...,n\})$ は添字集合 $\{1,2,...,n\}$ の冪集合である．また，$a_{\mathcal{I}} \prod_{i \in \mathcal{I}} x_i$ を項とよび，係数 $a_{\mathcal{I}}$ とモノミアル $\prod_{i \in \mathcal{I}} x_i$ で構成される．モノミアルが 1 である項 $(\mathcal{I} = \emptyset)$ が f の定数項となる．

中間値 1 ビットを計算するブール関数は平文 $\mathbf{v}$ 及び鍵 $\mathbf{k}$ で $\mathsf{f}(\mathbf{v}, \mathbf{k})$ と書ける．f の入力のうち，添字集合 $\mathcal{I}$ で示される平文の変数ビット以外は全て係数または定数項とみなしたブール関数を $\mathsf{f}_{\mathcal{I}}$ とする．例えば，$\mathsf{f}(\mathbf{v}, \mathbf{k}) = k_1 v_1 v_2 \oplus k_2 k_3 v_3$ の v_2 以外は係数または定数項とした場合 $(\mathcal{I} = \{2\})$，$\mathsf{f}_{\mathcal{I}}$ の $k_1 v_1 v_2$ という項の係数は $k_1 v_1$ となり，$k_2 k_3 v_3$ は定数項となる．関数 $\mathsf{f}_{\mathcal{I}}$ における最大次数項は $a_{\mathcal{I}} \prod_{i \in \mathcal{I}} v_i$ である．

付録 H で示した方法で選択平文集合 $\mathcal{V}_{\mathcal{I}}$ を設定し，積分値 (定義 2.2 参照) を計算する．すると，$\mathsf{f}_{\mathcal{I}}$ の最大次数項の係数 $a_{\mathcal{I}}$ は

$$a_{\mathcal{I}} = \bigoplus_{\mathbf{v} \in \mathcal{V}_{\mathcal{I}}} \mathsf{f}(\mathbf{v}, \mathbf{k}), \tag{K.2}$$

で求まる [11]．これをブール関数の特性とよぶ．なお，$a_{\mathcal{I}}$ は定数ビット $\mathbf{v}_{const}$ 及び鍵 $\mathbf{k}$ を入力とするブール関数 $a_{\mathcal{I}} = \mathsf{a}_{\mathcal{I}}(\mathbf{v}_{const}, \mathbf{k})$ とみなすことができる．式 (K.2) から $\mathsf{a}_{\mathcal{I}}(\mathbf{v}_{const}, \mathbf{k})$ の値を得る．なお，$\mathbf{v}_{const}$ 及び $\mathbf{k}$ は f の入力では定数とみなし，$\mathsf{a}_{\mathcal{I}}$ の入力では変数とみなしている．$\mathsf{a}_{\mathcal{I}}(\mathbf{v}_{const}, \mathbf{k})$ の値は $\mathbf{v}_{const}$ 及び $\mathbf{k}$ によって変化する．Integral 攻撃では，積分値が鍵 $\mathbf{k}$ に依らず定数となるものを用いる．また，サ

イドチャネル Cube 攻撃では $a_{\mathcal{I}}$ が鍵 $\mathbf{k}$ を変数として線形となるものを用いる．

L　鍵列挙とランク評価

限られたサイドチャネル情報から計算能力を活かして鍵を推定する手法として鍵列挙がある．なお，鍵列挙は分割統治法を行うサイドチャネル攻撃全般に適用が可能である．

まず，サイドチャネル情報から副鍵 $\mathbf{k}_i$ ($i \in \{1, 2, ..., m_{key}\}$) の候補を事後確率順に列挙した列を作成する．得られた列を統合することで，鍵 $\mathbf{k}$ の候補の列を作成する．最適な鍵列挙 [48] であれば，鍵 $\mathbf{k}$ の候補は事後確率順に正しく列挙される．なお，事前確率を一様分布であると仮定した場合，事後確率と尤度による推定は同じ結果になるため [31]，事後確率の代わりに尤度を用いることができる．以下では直感的に理解が容易な事後確率を用いる．

鍵列挙を用いたサイドチャネル攻撃に対する計算量の評価を効率的に行う手法として，ランク評価がある [49]．鍵列挙と同様，複数の手法が提案されている [5][16][20][49]．ランク評価では正しい鍵を与えられ，鍵 $\mathbf{k}$ の列の中で正しい鍵のランクを見積もる．ランクは正しい鍵が見つかるまでに攻撃者が検証する候補数を示し，計算量と等しい．

本論文では，DBA 及び SCCA の攻撃手法において最適な鍵列挙 [49] を，これらに対する耐性評価においてヒストグラムを用いたランク評価 [20] を用いる．それぞれ付録 M 及び N に示す．

M　最適な鍵列挙 [48]

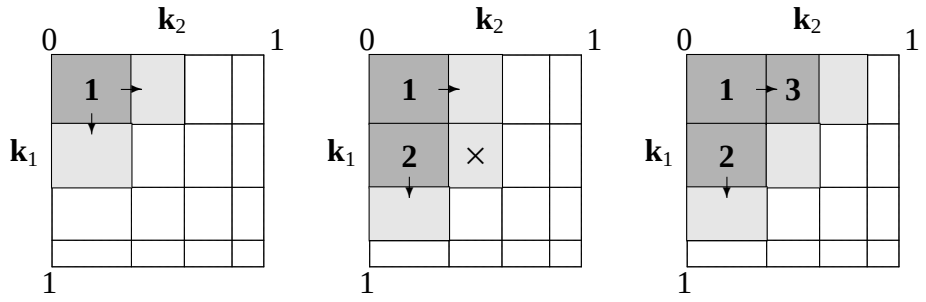
最適な鍵列挙は与えられたデータに対して，任意の鍵候補が厳密に事後確率の順番で列挙される．まず，Algorithm M.1 に 2 個の副列を統合する関数を示す．図 M.1 を用いて例示する．2 個の副列 $\{\mathbf{k}_1^1, \mathbf{k}_1^2, \mathbf{k}_1^3, \mathbf{k}_1^4\}$ 及び $\{\mathbf{k}_2^1, \mathbf{k}_2^2, \mathbf{k}_2^3, \mathbf{k}_2^4\}$ を統合して 1 個の列にする．縦横の線分が各副鍵の候補の事後確率を表現している．線分の

Algorithm M.1 最適な鍵列挙 [48]

function KeyEnum($\mathcal{T}_{\mathbf{k}_1}$,$\mathcal{T}_{\mathbf{k}_2}$)
  前線集合の初期化 $\mathcal{F} \leftarrow (\mathbf{k}_1^1, \mathbf{k}_1^2)$
  while $\mathcal{F} \neq \emptyset$ **do**
    $(\mathbf{k}_1^o, \mathbf{k}_2^{o'}) \leftarrow \mathcal{F}$ で最も同時事後確率が高い候補
    $\mathcal{T}_{(\mathbf{k}_1,\mathbf{k}_2)} \leftarrow \mathcal{T}_{(\mathbf{k}_1,\mathbf{k}_2)} \cup \{(\mathbf{k}_1^o, \mathbf{k}_2^{o'})\}$
    $\mathcal{F} \leftarrow \mathcal{F} \backslash \{(\mathbf{k}_1^o, \mathbf{k}_2^{o'})\}$
    if $o + 1 \leq |\mathcal{T}_{\mathbf{k}_1}|$ かつ $\mathcal{F}$ の (o+1) 番目の行に他の候補がない **then**
      $\mathcal{F} \leftarrow \mathcal{F} \cup \{(\mathbf{k}_1^{o+1}, \mathbf{k}_2^{o'})\}$
    end if
    if $o' + 1 \leq |\mathcal{T}_{\mathbf{k}_2}|$ かつ $\mathcal{F}$ の (o'+1) 番目の列に他の候補がない **then**
      $\mathcal{F} \leftarrow \mathcal{F} \cup \{(\mathbf{k}_{1,2}^o, \mathbf{k}_1^{o'+1})\}$
    end if
  end while
  return $\mathcal{T}_{(\mathbf{k}_1,\mathbf{k}_2)}$
end function

間隔が広い候補はより大きな事後確率が与えられており降順になっている．縦横の線分で定義される長方形の領域は同時事後確率を示し，面積順に候補を列挙する．領域内の整数が列挙される順番であり，図 M.1 は左から順に 3 番目の候補を指定するまでの手順を示す．図 M.1 (a) では，1 番目の候補として $(\mathbf{k}_1^1, \mathbf{k}_2^1)$ を指定する．なお，暗い灰色で既に列挙された候補を示す．これに隣接した候補を前線集合とよび，明るい灰色で示す．図 M.1 (b) では，前線集合から $(\mathbf{k}_1^2, \mathbf{k}_2^1)$ を 2 番目の候補として指定する．次の前線集合を設定する際，一部の候補を事前に除外する．$\mathbf{k}_2$ の軸上には $(\mathbf{k}_1^1, \mathbf{k}_2^2)$ 及び $(\mathbf{k}_1^2, \mathbf{k}_2^2)$ が前線集合に含まれるが，面積の小さい $(\mathbf{k}_1^2, \mathbf{k}_2^2)$ を除外する (図 M.1 中 × で示している)．つまり，同一軸上で複数の候補が前線集合に含まれる場合，最も左上側にある候補以外は除外する．図 M.1(c) では，$(\mathbf{k}_1^1, \mathbf{k}_2^2)$ 及び $(\mathbf{k}_1^3, \mathbf{k}_2^1)$ を比較して面積の大きい $(\mathbf{k}_1^1, \mathbf{k}_2^2)$ が 3 番目の候補として選ばれている．

次に，3 個以上の副列を生成してこれらを 1 個の列に統合する方法を示す．単



(a) 1 番目の選択後　(b) 2 番目の選択後　(c) 3 番目の選択後

図 M.1　最適な鍵列挙の適用例 [48]

純に Algorithm M.1 を繰り返して統合を行う場合，メモリ量が計算環境の限界を超えてしまう．例えば，鍵長が 128 ビットの場合，候補数が少なくとも 2^{64} 個の副列が Algorithm M.1(最終的な処理) の入力となる．そこで，上位の副列を再帰的に下位の副列に分解しながら生成する方法がある (再帰分解法)．再帰分解法はメモリ量を削減できるが，再帰呼び出しを階層的に行うため計算量は増大する．計算量は列挙する候補数に依存して増加することが分かっている [10].

N　ヒストグラムを用いたランク評価 [20]

Algorithm N.1 にヒストグラムを用いたランク評価 HistRankEst を示す．サイドチャネル情報の集合を $\mathcal{L}$ とする．これらから，副鍵の全候補の対数事後確率分布 $\log_2(Pr[\mathbf{k}_i|\mathcal{L}])$ を計算する ($i \in \{1, 2, ..., m_{key}\}$)．事後確率を基準とし，ビンの数が n_{bin} であるヒストグラム H_i を作成する．$\mathsf{H}_i(b)$ をヒストグラム H_i の b 番目のビンに含まれる副鍵の候補数とする．対数事後確率から $b = \lfloor \log_2(Pr[\mathbf{k}_i|\mathcal{L}])/l_{bin} \rfloor$ と求まる (l_{bin}:ビンの幅)．2 個のヒストグラムは畳み込み処理によって統合される．図 N.1 に示す例では，H_1 と H_2 の畳み込みで $\mathsf{H}_{1:2}$ が生成されている．H_1 から H_i までのヒストグラムの畳み込みで生成されるヒストグラムを $\mathsf{H}_{1:i}$ とする．$\mathsf{H}_{1:i}$ におけるビンの数は $i \cdot n_{bin} - (i-1)$ となる．ヒストグラムを用いたランク評価では $\mathsf{H}_{1:m_{key}}$ を生成し，このヒストグラムを用いて正しい鍵のランクを評価する．正しい鍵が属

Algorithm N.1 ヒストグラムを用いたランク評価 [20]

function HistRankEst($\{\mathsf{H}_1, \mathsf{H}_2, ..., \mathsf{H}_{m_{key}}\}, b^\dagger$)

ヒストグラムの初期化 $\mathsf{H}_{1:1} \leftarrow \mathsf{H}_1$.

for $i = 2 \rightarrow m_{key}$ **do**

for $b = (i-1) \rightarrow i \cdot n_{bin} - (i-1)$ **do**

for $b' = 1 \rightarrow n_{bin}$ **do**

$\mathsf{H}_{1:i}(b+b') \leftarrow \mathsf{H}_{1:i}(b+b') + \mathsf{H}_{1:i-1}(b) \cdot \mathsf{H}_i(b')$

end for

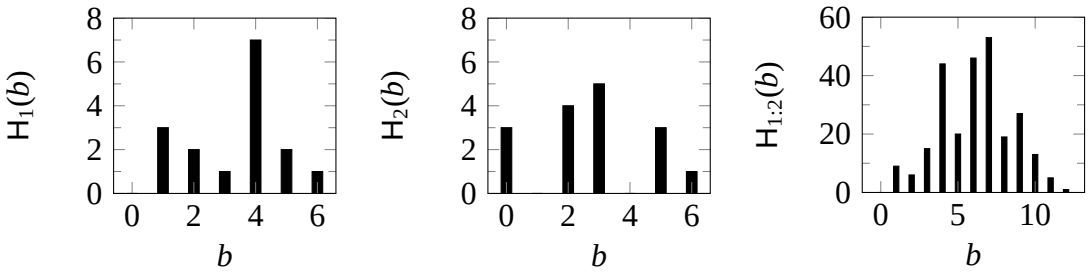
end for

end for

ランクの推定値を計算 $t_k \leftarrow \sum_{b=b^\dagger}^{m_{key} \cdot n_{bin} - (m_{key}-1)} \mathsf{H}_{1:m_{key}}(b)$

return t_k

end function



(a) H_1 (b) H_2 (c) $\mathsf{H}_{1,2}$

図 N.1 ヒストグラムの畳み込みの例 [37].

するビンの添字を $b^\dagger$ としたとき，ランクの推定値は $t_k = \sum_{b=b^\dagger}^{m_{key} \cdot n_{bin} - (m_{key}-1)} \mathsf{H}_{1:m_{key}}(b)$ となる．得られるランクには誤差が生じるが，ランクの下界 t_{lower} と上界 t_{upper} は

$$t_{lower} = \sum_{b=b^\dagger + \lceil m_{key}/2 \rceil}^{m_{key} \cdot n_{bin} - (m_{key}-1)} \mathsf{H}_{1:m_{key}}(b) \tag{N.1}$$

$$t_{upper} = \sum_{b=b^\dagger - \lceil m_{key}/2 \rceil}^{m_{key} \cdot n_{bin} - (m_{key}-1)} \mathsf{H}_{1:m_{key}}(b) \tag{N.2}$$

と求まる．上界及び下界の差が小さいければ，正確な評価を行うことができる．n_{bin} を増やすことで各ビンに含まれる候補の数は減るため，$\pm\lceil m_{key}/2 \rceil$ 個のビンの

差によって生じるランクの差は減少し，より正確な評価が行える．HistRankEst の主な計算は畳み込みであり，その計算量は

$$t_{re} = \sum_{i=2}^{m_{key}} \sum_{b=i-1}^{i \cdot n_{bin}-(i-1)} n_{bin} \tag{N.3}$$

となることから，n_{bin} に対して多項式時間の計算量であることが分かる．Glowacz らは AES-128 に対して計算機実験を行った [20]．その結果，Intel i7 を搭載した汎用コンピュータで 5 秒程度で実行が可能であり，$\log_2(t_k) - \log_2(t_{lower}) \leq 1$ 及び $\log_2(t_{upper}) - \log_2(t_k) \leq 1$ という小さい誤差となることを示した．

O　攻撃者の計算能力の基準

本論文では攻撃者が実行できる暗号化処理の回数を 2^{60} または 2^{80} と仮定している．計算機のランキングである TOP500 で 2017 年 12 月時点で 1 位である神威・太湖之光 [42] を基準とする．神威・太湖之光の FLOPS(1 秒間の浮動小数点演算回数) が 93.01[PFLOPS] であり，1 回の暗号化処理が 1,000[FLOPS] 要すると仮定する [3]．この仮定で，2^{60} 回の暗号化処理は約 3.5 時間，2^{80} 回は約 412 年要する．

まず，2^{80} 回の暗号化処理は現存する全ての計算機で現実的な時間内で実行できない．なお，TOP500 の 1 位である計算機の性能はここ 10 年で約 331 倍となっている [41]．10 年後に予想される神威・太湖之光の 331 倍の性能がある計算機であっても 2^{80} 回の暗号化処理に約 1 年を要する．したがって，$\mathsf{A}_{t \leq 2^{80}}$ を仮定することで，10 年後の攻撃者を考慮した長期的な安全性を保証することができる．

次に，2^{60} 回の暗号化処理は TOP500 の上位に入るような計算機ならば実行可能だが，一般的な計算機では困難である．例えば，Intel 社製では最新の Intel Core i9-7960X (2.8GHz) の性能を理論的に換算すると 1,433[FLOPS] であり [21]，2^{60} 回の暗号化処理に約 25 年要する．この処理を現実的な時間内に実行するコストは一般的な攻撃者にとっては過大である．したがって，$\mathsf{A}_{t \leq 2^{60}}$ を行える攻撃者を仮定することで，短期的な安全性を保証することができる．

研 究 業 績

学術論文

(1) Haruhisa Kosuge, Keisuke Iwai, Hidema Tanaka, Takakazu Kurokawa.: Search Algorithm of Precise Integral Distinguisher of Byte-based Block Cipher. In: The 11th International Conference on Information Systems Security (ICISS2015), Lecture Note in Computer Science, vol. 9478, pp. 303-323. Springer (Dec. 2015)

(2) Haruhisa Kosuge, Hidema Tanaka.: Security Evaluation of Light-Weight Block Ciphers by GPGPU. In: Advanced Computing: An International Journal (ACIJ), vol.7, No 3. AIRCC Publishing Corporation (May. 2016)

(3) Haruhisa Kosuge, Hidema Tanaka.: Improvement of Search Algorithm for Integral Distinguisher in Subblock-Based Block Cipher. In: International Journal on Cryptography and Information Security (IJCIS), vol. 6, No. 1. AIRCC Publishing Corporation (Jun. 2016)

(4) Haruhisa Kosuge, Hidema Tanaka.: Algebraic Degree Estimation for Integral Attack by Randomized Algorithm. In: The 17th International Workshop on Information Security Applications (WISA2016), Lecture Notes in Computer Science, vol. 10144, pp. 292-304. Springer (Aug. 2016)

(5) Haruhisa Kosuge, Hidema Tanaka.: Algebraic Degree Estimation of Block Ciphers Using Randomized Algorithm; Upper-Bound Integral Distinguisher. In: International Journal on Cryptography and Information Security (IJCIS), vol. 6, No. 3. AIRCC Publishing Corporation (Dec. 2016)

(6) Haruhisa Kosuge, Hidema Tanaka.: Differential Bias Attack for Block Cipher under Randomized Leakage with Key Enumeration. In: The 9th International Conference on Cryptology and Information Security (AFRICACRYPT2017), Lecture Notes in Computer Science, vol. 10239, pp. 95-116. Springer (May. 2017)

(7) Haruhisa Kosuge, Hidema Tanaka.: Theoretical Security Evaluation against Side-channel Cube Attack with Key Enumeration. In: The 5th International Conference on Cryptology and Information Security in Latin America (Latincrypt 2017), Lecture Notes in Computer Science, Springer (Sep. 2017 印刷中)

(8) Haruhisa Kosuge, Hidema Tanaka.: Improvements on Security Evaluation of AES against Differential Bias Attack. In: IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol.E100-A, No.11, pp.2398-2407. The Institute of Electronics, Information and Communication Engineers (Nov. 2017)

国際会議（査読あり）

(1) Haruhisa Kosuge, Keisuke Iwai, Hidema Tanaka and Takakazu Kurokawa.: Computational security evaluation of light-weight block cipher against integral attack by GPGPU. IEEE 2nd International Conference on Cyber Security and Cloud Computing (CSCloud2015), pp. 439-444, IEEE (Nov. 2015)

(2) Haruhisa Kosuge, Keisuke Iwai, Hidema Tanaka and Takakazu Kurokawa.: Integral attack on reduced-round RECTANGLE. IEEE 2nd International Conference on Cyber Security and Cloud Computing (CSCloud2015), 68-73, IEEE (Nov. 2015)

国内学会（口頭発表）

(1) 小菅悠久, 岩井啓輔, 田中秀磨, 黒川恭一: 軽量型ブロック暗号 TWINE の高階差分特性. コンピュータセキュリティシンポジウム 2014 (CSS2014), 2E1-1. 情報処理学会 (2014 年 10 月)

(2) 小菅悠久, 岩井啓輔, 田中秀磨, 黒川恭一: 平文制御を用いた確率的高階差分特性による TWINE の安全性評価. 2015 年暗号と情報セキュリティシンポジウム (SCIS2015), 1D2-1. 電子情報通信学会 (2015 年 1 月)