IMES DISCUSSION PAPER SERIES

共通鍵暗号を取り巻く現状と課題
― DESからAESへ ―

宇根正志・太田和夫

Discussion Paper No. 98-J-27

IMES

INSTITUTE FOR MONETARY AND ECONOMIC STUDIES
BANK OF JAPAN

日本銀行金融研究所

〒100-8630 東京中央郵便局私書箱 203 号

IMES Discussion Paper Series 98-J-27
1998 年 11 月

共通鍵暗号を取り巻く現状と課題

― DES から AES へ ―

宇根正志*1・太田和夫*2

要　旨

共通鍵暗号は、暗号化と復号に同一の鍵を用いる暗号であり、情報の秘匿・改ざん防止技術として、金融分野を中心に従来から幅広く利用されている。共通鍵暗号の中でもブロック暗号と呼ばれる方式が主要な商用暗号として利用されており、1977 年に米国政府標準暗号に認定された DES（Data Encryption Standard）が事実上の標準として利用されてきた。

しかし、DES は、鍵長が 56 bit であることから、近年のコンピューターのコストパフォーマンス向上等によって安全性が徐々に低下している。このため、現在金融分野を中心に、DES の代替暗号として Triple DES を利用する動きが広がっている。Triple DES は、DES のアルゴリズムを 3 回繰り返す方式であり、①DES からの移行が比較的容易である、②全数探索法に対する安全性が向上する等の利点を有している。

一方、米国政府は、次世代の標準暗号として AES（Advanced Encryption Standard）の標準化を進めている。AES は、標準化完了後、一般に利用可能となるまでにはさらに数年はかかるとみられているが、Triple DES の次の主要な暗号方式として位置付けられている。

DES から Triple DES、さらには AES への移行に象徴されるように、共通鍵暗号を取り巻く環境は急速に変化している。本稿では、共通鍵暗号の機能、構造、主要な解読法のほか、主要なブロック暗号に関するこれまでの安全性評価結果について整理するとともに、DES から Triple DES、そして AES への移行の経緯と現状について説明する。

キーワード：AES、DES、Triple DES、共通鍵ブロック暗号、米国政府標準暗号

JEL Classification：L86、L96、Z00

*1 日本銀行金融研究所研究第 2 課（E-mail: masashi.une@boj.or.jp）
*2 日本電信電話株式会社情報通信研究所（E-mail: ohta@sucaba.isl.ntt.co.jp）

本論文は、1998 年 11 月 4 日に日本銀行で開催された「金融分野における情報セキュリティ技術に関するシンポジウム」への提出論文に加筆・修正を加えたものである。

目　次

頁

Ⅰ．はじめに

共通鍵暗号は、暗号化と復号に同一の鍵を用いる暗号であり、情報の秘匿・改ざん防止技術として、金融分野を中心に従来から幅広く利用されている。共通鍵暗号には、ブロック暗号とストリーム暗号という 2 種類の暗号方式が存在するが、ストリーム暗号は軍事目的に利用される場合が多いことから、アルゴリズムが非公開となるケースが多い。このため、ブロック暗号が主要な商用暗号として利用されており、1977 年に米国政府標準暗号に認定された DES（Data Encryption Standard）がこれまで事実上の標準とされてきた。

しかし、DES は鍵長が 56 bit であることから、近年のコンピューターのコストパフォーマンス向上等によって、Brute Force Method に対する安全性低下が深刻化している。このため、現在金融分野を中心に、DES の代替暗号として Triple DES を利用する動きが広がっている。Triple DES は、2 つもしくは 3 つの異なる鍵を用いて、DES のアルゴリズムを 3 回繰り返して暗号化するという方式であり、①DES からの移行が比較的容易である、②2 つもしくは 3 つの異なる鍵を利用することによって、鍵長がそれぞれ 112 bit および 168 bit に拡張する効果を有しており、Brute Force Method に対する安全性が向上する等の利点を有している。

一方、米国政府は、次世代の標準暗号として、AES（Advanced Encryption Standard）の標準化を進めている。AES は、鍵長として 128、192、256 bit、ブロック長として 128 bit が利用可能なブロック暗号とされており、現在 15 の候補アルゴリズムの分析・評価が実施されている。米国政府は、AES の標準化を完了した後、AES を米国政府標準暗号として 20～30 年の間利用する方針を発表しており、次世代の代表的な暗号方式として幅広い分野において普及する可能性が高い。現在発表されている AES の標準化スケジュールによれば、早ければ 2000 年内にも候補アルゴリズムの 1 つが AES として認定される予定となっているものの、標準化完了後、AES がその安全性について高い信頼を得て、AES を利用した暗号製品が広く普及するまでには、さらに数年程度はかかるものとみられている。このため、AES が一般に利用可能になるまでの間は、Triple DES が DES の後継暗号として利用されるとの見方が多い。

DES から Triple DES、さらには AES への移行に象徴されるように、共通鍵暗号を取り巻く環境は急速に変化している。今後共通鍵暗号を利用するに際しては、共通鍵暗号に関する研究動向について理解を深めることが重要となっている。

本稿では、まず第Ⅱ章において、共通鍵暗号の機能、種類、構造等について説明する。続いて、第Ⅲ章において、商用暗号の主流となっているブロック暗号の主要な解読法について説明し、第Ⅳ章で、主要なブロック暗号の構造とこれまでの安全性評価結果について説明する。第Ⅴ章では、DES から Triple DES への移行の背景について説明し、Triple DES の構造や安全性評価結果について説明する。最後に、第Ⅵ章において、次世代の主要なブロック暗号と目されている AES の標準化動向について説明する。

Ⅱ．共通鍵暗号の概要と機能

共通鍵暗号は、暗号化と復号に同一の鍵を利用する暗号方式である。共通鍵暗号は、ネットワーク上でやり取りされるデータや外部記憶媒体に保管されるデータを秘匿するために用いられるほか、無権限者によるデータの改ざんを防止・検出するための技術としても利用されている。共通鍵暗号を利用して暗号通信を行う場合には、暗号化に利用した鍵を安全に受信者に配送する必要があるものの、鍵の配送が不要な公開鍵暗号よりも高速で暗号化・復号を行うことができる。このため、一般的に、データの暗号化手段として共通鍵暗号が利用され、共通鍵暗号の鍵を配送する手段として公開鍵暗号が利用されるケースが多い。

1. 共通鍵暗号の機能

共通鍵暗号の主な機能として、（1）守秘、（2）認証、（3）ハッシュ関数が挙げられる。

（1）守秘機能

インターネット等オープンなネットワーク上でデータをやり取りする場合、データ送信の途中で、送受信者以外の第三者によってデータの内容を覗き見られる可能性がある。また、磁気ディスク等の記憶媒体にデータを記録・保管する場合、無権限者がその媒体から無断でデータを読み出す可能性がある。こうした無権限者によるデータの覗き見を防止する機能が守秘であり、共通鍵暗号はデータの守秘を実現する技術の 1 つである。共通鍵暗号では、暗号化に利用された鍵を有する者だけが、その鍵で暗号化されたデータを復号することができる。信頼できる共通鍵暗号アルゴリズムが利用されている限り、鍵を知らない者が暗号化されたデータを解読することは計算量的に困難となる[1]。

（2）認証機能

データの認証は、オープンなネットワーク上でやり取りされるデータの正当性を確保する機能である。共通鍵暗号によって実現される認証機能としては、主に①アクセスしてきた利用者の正当性を確認する「ユーザー認証」、②通信の途中で第三者によってデータが改ざんされていないことを確認する「メッセージ認証」、が挙げられる。

①ユーザー認証

ユーザー認証は、ネットワーク等を経由してアクセスしてきた通信相手が正当な権限を有しているかどうかを確認する機能である。共通鍵暗号では、通信を行う者同士の間で鍵

[1] 計算量的に困難であるとは、その計算を行うことは理論的には可能であるものの、実際にその計算を実行するには計算量が非常に大量となり、膨大な費用と時間を必要とすることから、事実上不可能であることを意味する。どの程度の計算量が「事実上不可能」であるかは、その時々の技術条件等によって左右される。

を共有して安全に管理する必要があるが、一旦鍵が共有されると、その鍵を使って正しい演算を行うことが可能か否かによってユーザー認証を実現することができる。例えば、以下の手順によってユーザー認証が実現される（図 1 参照）。

（ステップ 1）検証者が乱数 r を生成する。
（ステップ 2）検証者は被検証者に r を送信する。
（ステップ 3）被検証者は r を受信し、検証者との間で既に共有している鍵 K を使って r を暗号化する。
（ステップ 4）被検証者は、r を暗号化したデータ E(K, r)を検証者に送信する。
（ステップ 5）検証者は、鍵 K で r を暗号化したデータと受信した E(K, r)を照合し、一致した場合には被検証者が正当なユーザーであることを確認する。

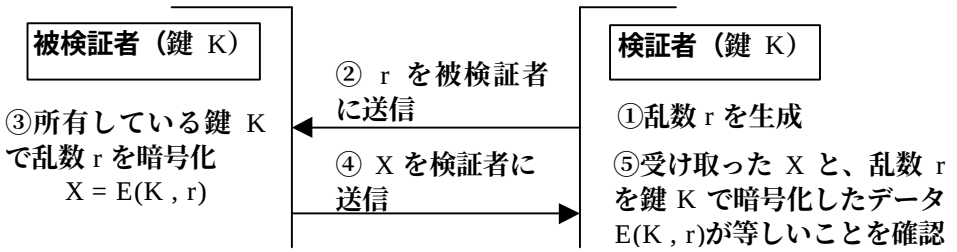


図 1　共通鍵暗号を利用したユーザー認証の例

②メッセージ認証

共通鍵暗号を利用したメッセージ認証としては、メッセージの認証子 MAC（Message Authentication Code）を利用する方式が一般的である[2]。MAC は、メッセージを送信者の鍵を利用して圧縮変換することによって生成されるデータであり、メッセージの真正性と送信者確認を同時に行うことを目的としてメッセージに付されるものである。MAC は、メッセージが 1 bit でも変更されると、そのメッセージに対応する MAC も変化するという性質を有しているほか、MAC を生成する際には、MAC からもとのデータを復元することが計算量的に困難な圧縮変換が利用される。MAC を利用したメッセージ認証の方法は、以下の通り（図 2 参照）。

（ステップ 1）被検証者は、予め共有している鍵 K を使ってメッセージ M に対する MAC を作成する。
（ステップ 2）被検証者は、生成した MAC とともに M を検証者に送付する。
（ステップ 3）検証者は、受信したメッセージ M から鍵 K を利用して MAC を生成し、メッセージとともに受信した MAC と一致するかどうかを確認する。一

[2] ISO 8731-1（Banking – Approved algorithms for message authentication – Part 1: DEA）では、金融業務で利用される MAC を生成するアルゴリズムとして、CBC モードを利用した DES（後述、図 3 参照）が規定されている。

致した場合、メッセージの送信者がその鍵を共有している被検証者であり、かつ送信されたメッセージが改ざんされていないことが確認される。万一通信の途中で第三者によってメッセージや MAC が改ざんされた場合、MAC の検証は成功しないため、改ざんの検出が可能となる。

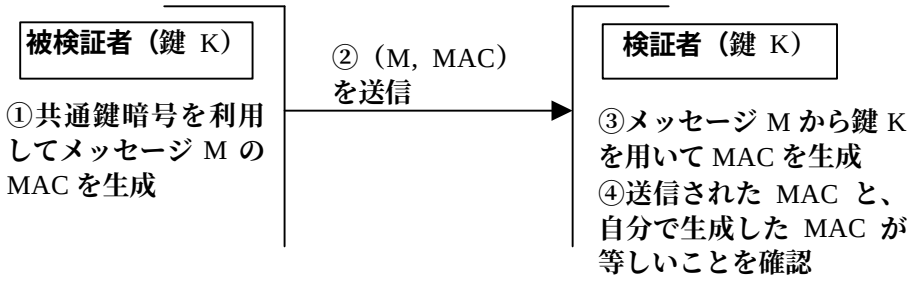
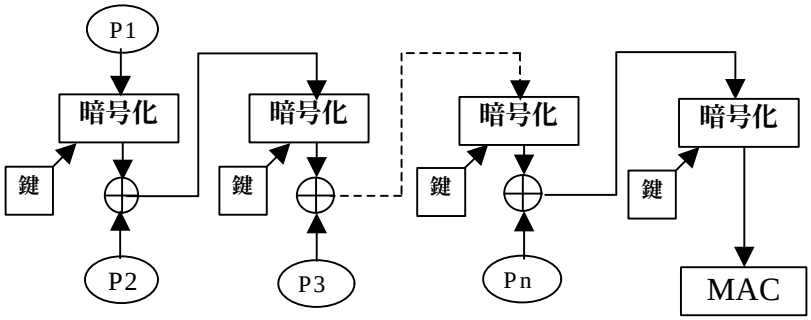


図 2　MAC **を利用したメッセージ認証方式**

なお、共通鍵暗号を利用した典型的な MAC **の生成方法は、以下の図** 3 **の通り。**



（ ⊕ **：排他的論理和、**Pi**：平文ブロック、**i = 1,2,..,n）

図 3　**共通鍵暗号を利用した典型的な** MAC **の生成方法**

③ハッシュ関数

ハッシュ関数は、任意のデータを一定の bit **長のデータに圧縮変換する関数であり、前述の** MAC **生成のようにメッセージ認証等に利用される。ハッシュ関数としては、メッセージダイジェストタイプと呼ばれている** MD5 **や** SHA-1 **等が広く利用されているが、共通鍵暗号を利用する方式も提案されている。共通鍵暗号を利用する方式の中でも代表的なものとしては、**Matyas-Meyer-Oseas **方式が挙げられる**[3]**。この方式によるハッシュ値生成方**

[3] Matyas-Meyer-Oseas **方式は、ブロック暗号を利用したハッシュ関数の利用方法について規定されている** ISO/IEC 10118-2（Hash-functions Part 2: Hash-functions using an n-bit block cipher algorithm）**に記載されている。本標準には、この方式のほかに、**Matyas-Meyer-Oseas **方式のハッシュ関数を** 2 **つ並列に配置し、ブロック長の** 2 **倍のサイズのハッシュ値を生成する** MDC-2 **と呼ばれる方式も記載されている。なお、この標準には、**Matyas-Meyer-Oseas **方式**

法は、以下の通り（図 4 参照）。

（ステップ 1）ハッシュ化するデータを一定長のブロック単位（利用するブロック暗号のブロック長に対応）に分割する。分割後のデータを Di（i = 1,…,n）とする。

（ステップ 2）初期値 K_0 を鍵として D_1 をブロック暗号によって変換し、その変換後のブロックを D_1 との排他的論理和によって変換して H_1 を生成する。H_1 のサイズはブロック暗号のブロック長に等しい。

（ステップ 3）H_1 を鍵生成関数 U によって変換して鍵 K_2 を生成し、鍵 K_2 を用いて次のブロック D_2 をブロック暗号によって変換する。変換後のデータは、D_2 との排他的論理和によって変換され、H_2 が生成される。

（ステップ 4）ステップ 3 の手続きを最後のブロック D_n の変換が終了するまで継続し、その結果生成される H_n がハッシュ値となる。

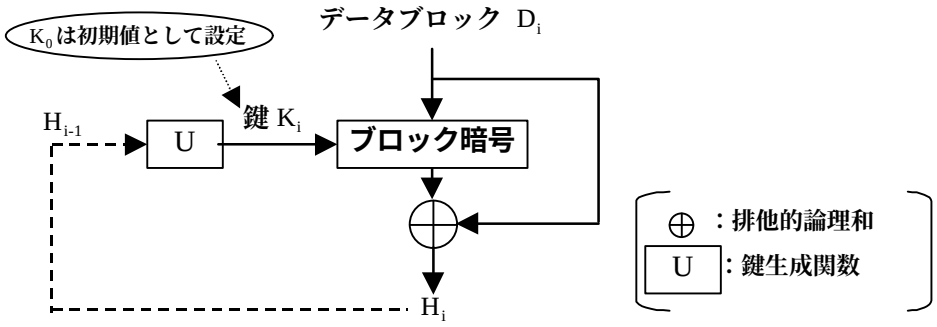


図 4　Matyas-Meyer-Oseas 方式によるハッシュ値生成方法

2. 共通鍵暗号の種類

共通鍵暗号は、大別すると、ストリーム暗号とブロック暗号に分類される（図 5 参照）。

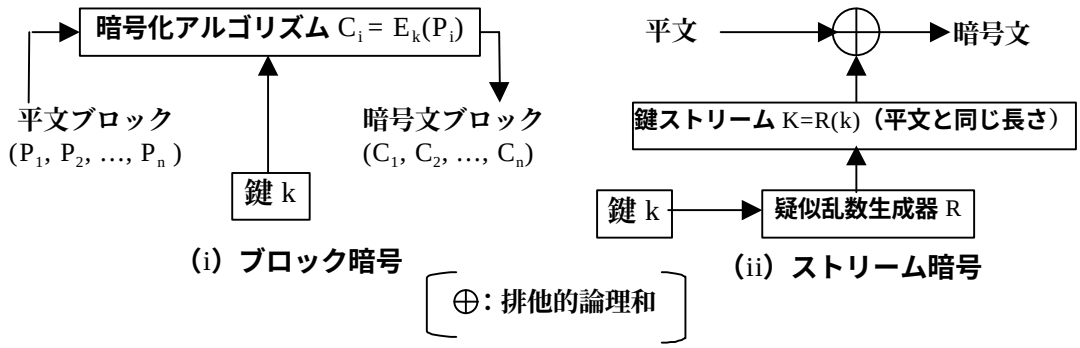


図 5 ブロック暗号とストリーム暗号の一般的な暗号化手順

に利用するブロック暗号の例として、DES が紹介されている。

ストリーム暗号は、暗号化する平文と同じ bit 長の鍵ストリームを疑似乱数生成器等によって生成し、平文と鍵ストリームの排他的論理和を計算することによって暗号文を生成する方式である。ストリーム暗号は、軍事用に利用されているものが多く、具体的なアルゴリズムの仕様が公開されている暗号方式は比較的少ない。

一方、ブロック暗号は、暗号化するデータをある一定長のブロックごとに分割し、各ブロックごとに同一の鍵で暗号化する方式である。DES をはじめとして、多くのブロック暗号のアルゴリズムが公開されており、ブロック暗号が現代の商用暗号の主流となっていることから、以下ではブロック暗号について説明することとする。

3. ブロック暗号の構造

ブロック暗号は、データランダム化部と鍵スケジューリング部によって構成される（図 6 参照）。

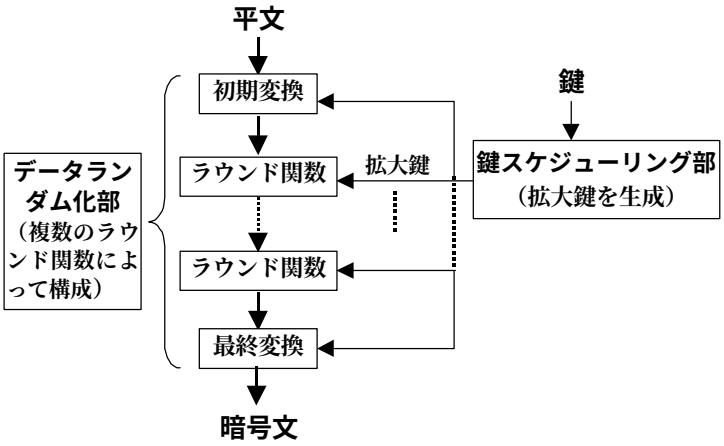


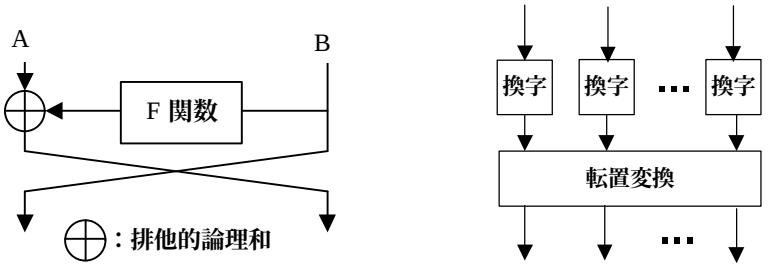
図 6 一般的なブロック暗号のアルゴリズムの構造

(1) データランダム化部

データランダム化部は、鍵スケジューリング部において生成された拡大鍵を利用して平文ブロックを暗号化する部分である。多くの暗号方式において採用されているデータランダム化部は、①初期変換、②繰り返し利用される基本変換（ラウンド関数と呼ばれる）、③最終変換によって構成されている。暗号方式によっては、初期変換や最終変換がないものも存在する。

ラウンド関数は構造上いくつかのタイプに分類されるが、よく利用されるものとして、Feistel 構造と SPN 構造が挙げられる（図 7 参照）。Feistel 構造は、米 IBM 社の Feistel が開発した Lucifer のラウンド関数に初めて採用されたものであり、2 つのサブブロックの一方を F 関数と呼ばれる非線形変換によって変換した後、その変換されたサブブロックとの排他的論理和によって他方のサブブロックを変換し、2 つのサブブロックの位置を入れ換える、という手順で変換が行われる。Feistel 構造には、①構造が単純であり、安全性

に関する分析が容易である、②暗号化変換と復号変換が同一である（同じ鍵を利用した暗号化変換で暗号文を変換すると平文が得られる）、等の特徴点があり、DES を始めとする多くの暗号方式において採用されている。



（Feistel 構造）　　（SPN 構造）

図 7　Feistel 構造と SPN 構造

一方、SPN（Substitution-Permutation Network）構造は、換字変換（ある数値を別のある数値に規則的に置き換える変換）と転置変換（データブロックの bit の位置を規則的に入れ替える変換）が連続的に配置された構造であり、DES の F 関数等に利用されている[4]。SPN 構造には、①暗号化変換と復号変換が同一にはならず、暗号化変換を復号に利用するためには別の変換手段を適宜付加する必要がある、という短所が存在するものの、②データブロックの攪拌が速い（Feistel 構造においてはデータブロック全体を非線形変換（F 関数）するためにはラウンド関数 2 段が必要となる一方、SPN 構造ではラウンド関数 1 段で変換できる）等の長所がある。

なお、これらの構造以外にも、一般形 Feistel 構造や 2-round SPN 構造等様々なタイプのラウンド関数が考案されている（詳細は巻末の別紙表 9 を参照）。

(2) 鍵スケジューリング部

鍵スケジューリング部は、鍵からデータランダム化部での変換に必要な拡大鍵を生成する部分である。

4. ブロック暗号の利用モード

ブロック暗号によって暗号化・復号を行う場合、いくつかの種類の利用モードが存在する。主要な利用モードとして、①ECB モード、②CBC モード、③OFB モード、④CFB

[4] SPN 構造は、Shannon が提案した合成関数（product cipher）の安全性に関する考え方に依拠して開発されたといわれている。合成関数は、2 つ以上の異なる種類の関数を繰り返し利用することでデータを変換する関数を指すが、Shannon は「単純な変換を単独で暗号化関数に利用してもその暗号の安全性は大して向上しないが、複数の変換を組合せて暗号化関数を構成することで、その暗号の安全性を著しく向上させることができる」ことを情報理論的に示した。

モードが挙げられる[5]（図 8 参照）。

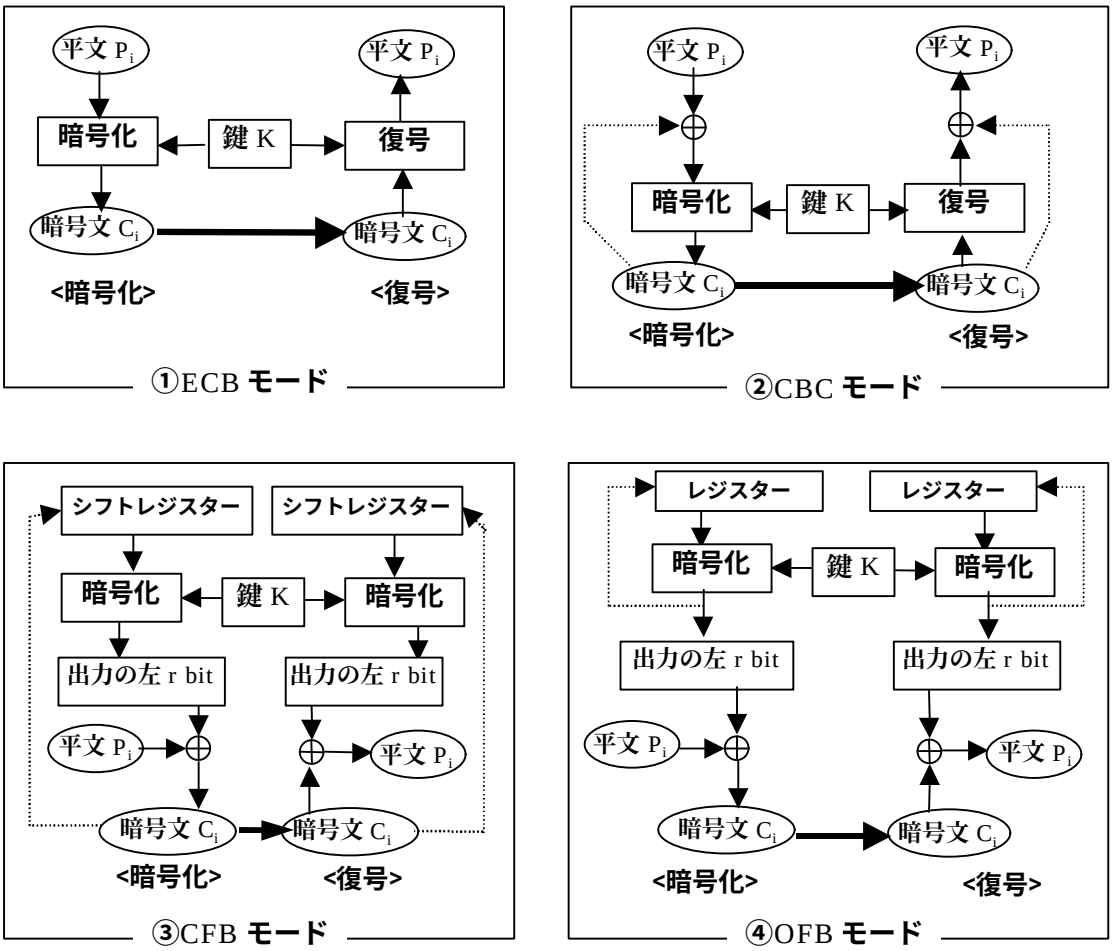


図 8　主要な 4 つの利用モード

（1）ECB モード

ECB（Electronic Codebook）モードは、各ブロックを他のブロックとは独立に暗号化・復号するモードであり、暗号化する場合には、各平文ブロックをそれぞれ暗号化アルゴリズムによって変換することで暗号文ブロックが生成される。このため、暗号文を通信相手に送信する途中で、ある暗号文ブロック 1 個にエラーが発生したとしても、復号の際にはその暗号文ブロックに対応した平文ブロック 1 個にエラーの影響が及ぶだけである。ECB モードでは、同じ鍵を利用して同じ平文ブロックを暗号化すると、同一の暗号文ブロックが生成される。

[5] これらの 4 つの利用モードは、ブロック暗号の利用モードに関する標準規格 ISO/IEC 10116（Information technology ― Security techniques ― Modes of operation for n-bit block cipher）において規定されている。

(2) CBC モード

CBC モード（Cipher Block Chaining）は、各平文ブロックを直前の暗号文ブロックとの排他的論理和によって変換した後、その変換結果を暗号化アルゴリズムによって再度変換することで暗号文ブロックを生成するモードである。最初の平文ブロックを CBC モードで暗号化する場合には、直前の暗号文ブロックとして初期値が与えられる。このため、同じ平文ブロックを同じ鍵で暗号化しても、初期値が異なっている場合には、異なる暗号文ブロックが生成される。復号の際には、まず同一の鍵を用いた復号アルゴリズムによって暗号文ブロックを変換した後、直前の暗号文との排他的論理和を計算して平文ブロックを生成する。

暗号文を通信相手に送信する途中に、ある暗号文ブロック 1 個にエラーが発生した場合には、復号の際、その暗号文ブロックに対応した平文ブロックと次の平文ブロックにエラーの影響が及ぶ。なお、初期値は暗号通信者間で秘密にする必要はないが、配送される初期値の真正性を確保する必要がある。

(3) CFB モード

CFB（Cipher Feedback）モードは、利用する暗号アルゴリズムのブロック長 n bit よりも短い r bit（r < n）の平文ブロックを暗号化する際に利用されるモードの 1 つである。CFB モードにおける暗号化の手順は、次の通り。

①n bit の初期値がシフトレジスターに入力・記録される。
②シフトレジスターから初期値が出力され、暗号化アルゴリズムによって変換される。
③変換後のデータの左から r bit 分のデータが抽出される。
④第 1 番目の平文ブロック（r bit）は、抽出された r bit データとの排他的論理和によって変換されて第 1 番目の暗号文ブロック C_1（r bit）となる。
⑤第 1 番目の暗号文ブロック C_1 はシフトレジスターにフィードバックされた後、シフトレジスターに記録されているデータ（最初は初期値）の右側に結合される。結合されたデータ（n+r bit）は、左 r bit が切り捨てられて n bit データに変換され、新たにシフトレジスターに記録されるとともに出力される。
⑥シフトレジスターから出力された n bit データは、暗号化アルゴリズムによって変換される。
⑦③に戻って、③～⑥の手続きが、最後の平文ブロックの変換が終了するまで繰り返される。

復号の際には、初期値と各暗号文ブロックを基にシフトレジスターから n bit データが順々に出力され、同一の鍵による暗号化アルゴリズムによって変換された後、左 r bit のデータと暗号文ブロックとの排他的論理和によって平文ブロックが生成される。このように、CFB モードでは、復号の際にも暗号化アルゴリズムが利用される。

暗号文を通信相手に送信する途中に、ある暗号文ブロック 1 個にエラーが発生した場合、復号の際には最大で T + 1 個（T は (n/r)以上の最小の整数）の平文ブロックにエラーの影響が及ぶこととなる。

(4) OFB モード

OFB（Output Feedback）モードは、CFB モードと同様に、暗号アルゴリズムのブロック長 n bit よりも短い r bit（r < n）の平文ブロックを暗号化する際に利用することができる。OFB モードにおける暗号化の手順は、次の通り。

①n bit の初期値がレジスターに入力・記録される。
②レジスターから初期値が出力され、暗号化アルゴリズムによって変換される。
③変換後のデータはレジスターにフィードバックされて記録される。
④変換後のデータの左から r bit 分のデータが抽出され、第 1 番目の平文ブロック（r bit）との排他的論理和によって変換されたデータが第 1 番目の暗号文ブロック（r bit）となる。
⑤レジスターに記録されているデータが出力され、暗号化アルゴリズムによって変換される。
⑥変換後のデータはレジスターにフィードバックされて記録されるとともに、左から r bit 分のデータが抽出され、平文ブロックとの排他的論理和によって暗号文ブロックが生成される。
⑦⑤に戻り、⑤と⑥の手続きが繰り返される。

復号の際には、通信相手から送信された初期値を暗号化アルゴリズムによって順々に変換し、各変換後のデータの左から r bit 分のデータと各暗号文ブロックとの排他的論理和によって平文ブロックが生成される。OFB モードにおいても、CFB モードと同様に復号の際には暗号化アルゴリズムが利用される。

なお、暗号文を通信相手に送信する途中で、ある暗号文ブロック 1 個にエラーが発生した場合、復号の際には、エラーが発生した暗号文ブロックに対応する平文ブロックのみに影響が及ぶこととなる。

Ⅲ. ブロック暗号の主要な解読法

ブロック暗号の解読法は、Brute Force Method と Short Cut Method の 2 つに分類される。Brute Force Method は、全ての鍵の候補をしらみつぶしに試してみる方法である。代表的な Brute Force Method としては、全数探索法、タイムメモリートレードオフ法、暗号文一致攻撃、辞書攻撃が挙げられる。

一方、Short Cut Method は、暗号アルゴリズムの構造上の特徴や平文・暗号文の統計的偏りを利用して、候補となる鍵の集合全体から真の鍵の候補を絞り込む方法である。代表的な Short Cut Method としては、差分解読法、線形解読法、高階差分攻撃、補間攻撃、分割攻撃、関連鍵攻撃が挙げられる。最近では、差分解読法や線形解読法に対して十分な安全性を有していることが数値的に証明可能な暗号方式も提案されているが、その中のいくつかに対しては、新しく提案された高階差分攻撃や補間攻撃が有効であることが示されている。

なお、本節で取り上げる攻撃法以外にも、各暗号方式に固有の弱点を利用した攻撃法等多くの方法が存在する。

1. Brute Force Method

(1) 全数探索法

全数探索法は、入手した暗号文を候補となる鍵で順々に復号することによって真の鍵を探索する方法である。したがって、全数探索法に対する安全性は鍵長のみに依存し、鍵長が長いほど安全性は高まる。鍵長が n bit の場合、候補となる鍵の個数は 2^n 個となることから、平均的にみて 2^{n-1} 回の暗号化処理によって真の鍵を発見することができる。

なお、暗号化処理回数と処理時間の関係を、DES を想定して一定の条件の下で試算した結果は、以下の表 1 の通り。この試算結果は、対象となるアルゴリズムや利用可能な費用等によって大きく変化する点に留意する必要がある。

表 1　DES の暗号化処理回数と処理時間の関係

	2^{40}	2^{56}	2^{64}	2^{80}	2^{128}
処理時間	約 12 秒	約 225 時間（9.4 日）	約 80 ヶ月（6.6 年）	4.3×10^5 年	1.2×10^{20} 年

＜前提条件＞
(1) 暗号方式として DES を想定。DES の暗号化を各回数実行するために必要な時間を試算。
(2) 暗号化装置として、EFF が約 25 万ドルで製作した DES 解読専用装置を利用した場合を想定（詳細は後述）。本解読専用装置の一秒間の暗号化処理回数は約 888 億回（ 2^{36} 回）。

(2) タイムメモリートレードオフ法

タイムメモリートレードオフ法は、1980 年に Hellman によって最初に提案された解読法であり、送信される平文 1 個を予想し得る場合に、予め暗号文と平文の対応が分かるような索引表を作成しておくことにより、暗号文を入手した後に短時間で鍵の検索を可能と

する方法である。タイムメモリートレードオフ法では、予め作成しておく索引表が大きいほど鍵検索の時間を短縮することが可能となるほか、一旦検索表を作成すれば、鍵が変更されたとしても予め想定した平文が入手可能である限り適用できる。タイムメモリートレードオフ法に対する安全性は鍵長に依存し、鍵長が長いほど安全性は高まる。

(3) 暗号文一致攻撃

暗号文一致攻撃は、「n bit のサイズのデータをランダムに $2^{n/2}$ 個集めたときに、その中に同じデータが 2 個以上存在する確率が約 0.5 になる」という性質（バースデー・パラドックスと呼ばれる）を利用した攻撃法であり、同一の鍵によって生成された複数の暗号文の中から一致するものを見つけ出し、それらの暗号文に対応する平文の情報を入手する方法である。これまで提案されてきた多くのブロック暗号はブロック長が 64 bit であるが、暗号文一致攻撃を適用すれば 2^{32} 個程度の暗号文を集めると約 0.5 の確率で同じ暗号文を見つけることができる。このように、暗号文一致攻撃に対する安全性は暗号文のブロック長に依存し、ブロック長が長いほど安全性は高まる。

(4) 辞書攻撃

辞書攻撃は、ある鍵によって暗号化された暗号文と平文のペアを予め大量に集めて適当な外部記憶媒体に記録しておき、それを辞書のように利用することによって、盗聴等によって入手した暗号文に対応する平文を得る、という攻撃法である。ブロック長が n bit の場合、この攻撃を実行するためには 2^n 個の平文・暗号文ペアを記録できる容量をもつ媒体が必要となる。したがって、ブロック長が長いほど、辞書攻撃に対する安全性は高くなる。

2. Short Cut Method

(1) 差分解読法

差分解読法は、1990 年に Biham と Shamir によって提案された解読法（Biham and Shamir[1991]）であり、ある特定の差分を有する平文のペアに対して、特定の差分を有する暗号文のペアが生じる確率が高くなる場合に、それらの平文・暗号文のペアを利用して候補となる鍵を絞り込む方法である。例えば、まず F 関数において、ある一定の入力データの差分 ΔX に対して高い確率で生成される出力データの差分 ΔY を予め調べておき、次に鍵を K に固定した F 関数に ΔX の差分を有する入力データのペアを入力して、その出力データの差分が ΔY に偏っていないかどうかを調べる。固定した鍵 K が真の値であれば出力データの差分は高い確率で ΔY となるはずであり、差分が ΔY に偏っている場合、そのときの鍵 K が真の鍵である確率が高くなる。差分解読法では、このようにして候補となる鍵を絞っていく[6]。

---

[6]このように、攻撃者が自分にとって都合のよい平文とそれに対応する暗号文を利用できる場合の攻撃は選択平文攻撃と呼ばれている。

差分解読法に対する厳密な安全性評価尺度として、平均差分確率が提案されている（Lai, Massey and Murphy[1991]）。非線形関数の入出力データにおいて各差分のペアが発生する確率をすべての鍵について計算し、その中で最も発生確率が高くなる差分のペアを見つける。このような差分ペアの発生する確率が平均差分確率である。平均差分確率が小さいほど、差分解読法に対する安全性が高くなる。平均差分確率は以下のように定義される。

【平均差分確率】 t bit の鍵 k を変数とする非線形関数 $F_k(x)$（入出力データ x、y はいずれも n bit）の平均差分確率は、

$$\frac{1}{2^t}\sum_{k}\max_{\Delta x(\neq 0),\Delta y}\left(\frac{\#\{x \mid F_k(x)\oplus F_k(x\oplus\Delta x)=\Delta y\}}{2^n}\right)$$

ただし、Δx は x の差分、⊕は排他的論理和演算、#{x}は x の個数を表す。

差分解読法に対する安全性を評価する場合、データランダム化部全体の平均差分確率を計算することは困難であり、平均差分確率によって安全性が証明されている暗号方式は MISTY など一部に限られている。そこで、厳密な意味での安全性を証明するものではないが、差分解読法に対する安全性の必要条件として、最大差分特性確率と呼ばれる指標が利用されている。例えば、まず F 関数等データランダム化部の一部となっている非線形関数の平均差分確率を計算する。次に、各非線形関数の平均差分確率がすべて独立であると仮定した上で、これらの平均差分確率を基に、データランダム化部全体もしくはその一部における確率を計算する。このような条件付き確率が最大差分特性確率となる。最大差分特性確率の計算方法は、データランダム化部の構造に依存する。

なお、差分解読法による解読に必要な平文・暗号文ペアの数は、平均差分確率の逆数として求めることができる。例えば、ある 64 bit 鍵長・ブロック長のブロック暗号において平均差分確率が 2^{-60} であった場合、差分解読法を利用してこのブロック暗号を解読するためには少なくとも 2^{60} の選択平文・暗号文ペアが必要であることを意味する。

(2) 線形解読法

線形解読法は、1993 年に松井によって提案された解読法（Matsui[1994]）であり、平文と暗号文の bit 値の間に線形関係が発生する確率が 1/2 から乖離している場合、その線形関係を利用することによって候補となる鍵の数を絞り込む方法である。線形解読法を適用するためには、任意の平文・暗号文のペアを入手する必要がある[7]。

線形解読法に対する厳密な安全性評価尺度として、平均線形確率が提案されている（Nyberg[1995]）。非線形関数の入出力データの bit 間で各線形関係が発生する確率をすべての鍵について計算し、その中で最も発生確率が高くなる線形関係を見つける。このような線形関係が生じる確率が平均線形確率である。平均確率確率が小さいほど、線形解読

[7]攻撃者が任意の平文とそれに対応する暗号文を利用する攻撃は既知平文攻撃と呼ばれている。

法に対する安全性が高くなる。平均線形確率は以下のように定義される。

【平均線形確率】t bit の鍵 k を変数とする非線形関数 $F_k(x)$（入出力データ x、y はいずれも n bit）の平均線形確率は、

$$\frac{1}{2^t}\sum_k \max_{\Gamma x,\Gamma y(\neq 0)}\left(2\frac{\#\{x \mid x\bullet\Gamma x = F_k(x)\bullet\Gamma y\}}{2^n}-1\right)^2$$

ただし、Γx は x の特定のマスク値、•は内積、#{x}は x の個数を表す。

線形解読法に対する安全性を評価する場合、データランダム化部全体の平均線形確率を計算することは困難であり、平均線形確率によって安全性が証明されている暗号方式は一部に限られている。そこで、厳密な意味での安全性を証明するものではないが、線形解読法に対する安全性の必要条件として、最大線形特性確率と呼ばれる指標が利用されている。例えば、まず、F 関数等データランダム化部の一部となっている非線形関数の平均線形確率を計算する。次に、各非線形関数の平均線形確率がすべて独立であると仮定した上で、これらの平均線形確率を基に、データランダム化部全体もしくはその一部の確率を計算する。このような条件付き確率が最大線形特性確率となる。最大線形特性確率の計算方法は、データランダム化部の構造に依存する。

なお、線形解読法による解読に必要な平文・暗号文ペアの数は、線形確率の逆数として定まる。例えば、ある 64 bit 鍵長のブロック暗号において平均線形確率が 2^{-60} であった場合、線形解読法を利用してこのブロック暗号を解読するためには少なくとも 2^{60} の既知平文・暗号文ペアが必要であることを意味する。

(3) 高階差分攻撃

高階差分攻撃は、1994 年に Lai によって最初にそのアイデアが提案され（Lai[1994]）、その後 Jakobsen と Knudsen や下山・盛合・金子らによって、高階差分攻撃による既存の暗号方式への攻撃に関する研究成果が発表されており（Jakobsen and Knudsen[1997]、下山・盛合・金子[1997]）、現在ではブロック暗号の有力な攻撃法の 1 つとなっている。高階差分攻撃は、暗号化関数（例えば、F 関数）の入力データに関して出力データの高い階数の差分を取ると、鍵に依存しない定数が得られることを利用して候補となる鍵を絞り込む攻撃法である。例えば、F 関数に適用する場合、F 関数の出力データの各 bit 値は入力データと拡大鍵を変数とするブール多項式によって表現されるが、そのブール多項式の次数を n とすると、出力データの n 階差分値は定数となる。この n 階差分値を利用して拡大鍵を変数とする線形連立方程式を立てて解くことにより、拡大鍵を求めることができる。この拡大鍵の情報を利用して、真の鍵の候補を絞り込むことが可能となる。ただし、高階差分攻撃を適用するためには、攻撃対象となるラウンド関数あるいは F 関数のブール多項式次数を計算する必要がある。高階差分攻撃は選択平文攻撃の 1 つである。

高階差分攻撃は、差分解読法や線形解読法に対する安全性が証明されている暗号方式の

いくつかに対しても、有効となるケースが存在することが示されている。例えば、Nyberg and Knudsen の KN 暗号は、差分解読法に対して十分な安全性を有していることが証明されているが、高階差分攻撃によって攻撃に必要な選択平文数を大幅に減少させることが可能となるほか（6 段の KN 暗号の場合、選択平文・暗号文ペア数：2^{60}→2^{8}）、計算量も大幅に削減できる（ワークステーションを利用して 0.02 秒）ことが示されている（下山・盛合・金子[1997]）。こうした研究成果から、差分解読法や線形解読法に対して十分な安全性が確保されているとみられる暗号方式でも、高階差分攻撃のような新しい解読法に対して必ずしも安全であるとは言えないことが示唆される。

高階差分攻撃に対する安全性を評価するための厳密な指標はこれまでのところ提案されていないが、暗号化関数のブール多項式表現における次数が大きいほど攻撃に必要な選択平文数や計算量が増加することから、近似的な評価指標としてブール多項式の次数が利用されている。しかし、この次数が大きい値であったとしても、必ずしも高階差分攻撃に対する安全性が保証されるわけではない点には留意が必要である。

(4) 補間攻撃

補間攻撃は、1997 年に Jakobsen と Knudsen によって提案された解読法（Jakobsen and Knudsen[1997]）である。鍵を固定した場合に、暗号化関数は平文を変数とする n 次関数によって表現されるとすれば、暗号化関数は最大（n+1）項の多項式で表されるため、異なる（n+1）組の平文・暗号文のペアを入手し、（n+1）個の係数を未知とする連立方程式を立てて解くことによって、ある 1 つの鍵に対する暗号化関数を構成することができる。このようにして導出した暗号化関数を利用して真の鍵の候補を絞っていく攻撃法が補間攻撃である。補間攻撃を適用するためには、攻撃対象となるラウンド関数等が多項式によって表現できること、そして多項式の項数が比較的少ないことが必要である。補間攻撃では、暗号化関数の次数を下げるように平文を選択して攻撃に利用することができるほか、既知平文を利用することも可能である。

高階差分攻撃と同様に、補間攻撃を利用することによって、差分解読法・線形解読法に対して安全性が証明されている暗号方式のいくつかを解読することができるとの研究成果が発表されている。例えば、Lee と Cha によって提案された SNAKE 暗号は、差分解読法や線形解読法に対して十分な安全性を有しているとされている（Lee and Cha[1997]）が、補間攻撃を利用することによって、より少ない既知・選択平文数と計算量によって解読可能であることが示されている（盛合・下山・金子[1998]）。

補間攻撃に対する厳密な安全性評価指標は、高階差分攻撃同様、これまでのところ提案されていないが、鍵を固定したときの暗号化関数を入力データの多項式で表現した場合、その項数が大きいほど攻撃に必要な既知平文や計算量が増加することから、近似的な指標として多項式の項数が利用されている。しかし、この項数が大きな値であったとしても、必ずしも補間攻撃に対する安全性が保証されるわけではない。

(5) 分割攻撃

分割攻撃は、線形解読法を応用した攻撃法であり、1997 年に Harpes と Massey によって提案された（Harpes and Massey[1997]）。分割攻撃では、まず鍵を固定した上で複数の入出力データのペアを入手し、暗号化関数の入力データと出力データをそれぞれ特定の関数によって入力集合と出力集合に写像する。これらの集合を複数の部分集合に分割したときに、入力集合の特定の要素と出力集合の特定の要素との間に強い相関がないかどうかを調べる。強い相関がある場合には、その情報に基いて真の鍵の候補を絞り込むことが可能となる。分割攻撃は既知平文攻撃の 1 つである。

分割攻撃に対する厳密な安全性評価指標はこれまでのところ提案されていない。分割攻撃に対して安全であるための必要条件は、入力集合の要素と出力集合の要素との間に強い相関が発生しないことである。そのため、分割攻撃の近似的な評価指標として、入出力集合における要素間の偏りの大きさが利用されている。しかし、要素間での偏りが小さい場合であったとしても、必ずしも分割攻撃に対する安全性が保証されるわけではない。

(6) 関連鍵攻撃

関連鍵攻撃は、1993 年に Biham によって最初に提案された攻撃であり、ある特殊な関係を有する 2 つの異なる鍵による暗号化が可能な場合に、それらの鍵の関係や暗号化されたデータを手掛かりにして、真の鍵の候補を絞り込む方法である（Biham[1994]）。関連鍵攻撃は、適用可能な環境が非常に限定されていることから、他の攻撃法に比べて実現可能性は低いとみられている。

Ⅳ. 主要なブロック暗号と安全性評価

本節では、アルゴリズムが公開されている方式のうち、従来から安全性に関する研究が盛んに行われてきた主要なブロック暗号として、DES、FEAL、IDEA、SAFER、LOKI、RC5、MISTY の 7 つを取り上げ、それらのアルゴリズムの概要や安全性に関する分析結果について整理する（安全性評価結果については、本章末の表 2 参照）。

1. DES

(1) アルゴリズムの概要と構造

DES（Data Encryption Standard、NIST[1993]）は、初めての商用暗号として 1977 年に米国政府標準暗号に制定されたブロック暗号であり、DES の特許の所有者である IBM 社が標準化に際してロイヤリティフリーでの使用を認めたこと等によって、世界で最も幅広く利用されてきた。DES は、金融業務に利用される情報セキュリティ技術として多くの ISO の標準規格において参照されている[8]ほか、米国では金融業務用の暗号アルゴリズムとして標準化されており（ANSI X3.92）、多くの欧米の金融機関が DES を利用して自行システムのセキュリティ対策を講じてきた。これまで Fedwire、CHAPS、日銀ネット等の世界の主要な大口決済ネットワークにおいても、DES が事実上の標準暗号として利用されてきた。

しかし、近年のコンピューターのコストパフォーマンスの向上等によって、全数探索法に対する安全性低下が深刻になってきており、米国内の金融業務における標準規格を策定している ANSI X9 は、1998 年 8 月に Triple DES（後述）を新たな金融業務用の標準暗号（ANSI X9.52）として認定しており、public comment の期間を経て 10 月には正式に標準として認定される予定である。また、ISO は、利用する暗号アルゴリズムとして DES を参照している標準規格を Triple DES に移行する方針を決定しているほか、米国政府でも、1998 年中に DES に代わって Triple DES を新たに米国政府標準暗号に認定する意向を表明している。

①データランダム化部

DES は、平文を 64 bit 単位のブロックに分割し、各ブロックを 56 bit の鍵[9]を利用して暗号化する構造となっている。暗号化は以下の手順で実行される（図 9 参照）。

---

[8] DES が参照されている金融業務において利用される情報セキュリティ技術に関する ISO の標準規格としては、ISO 8731-1、ISO 8732、ISO 9564-2、ISO/IEC 10116、ISO/IEC 10118-2 等が挙げられる。

[9] 実際にアルゴリズムに利用される鍵は 64 bit であるが、このうち 8 bit はパリティビットであり、鍵として利用可能なのは実質 56 bit であることから、一般的に DES の鍵長は 56 bit とされている。

（ステップ 1）64 bit 平文ブロックを転置変換する。
（ステップ 2）64 bit の平文ブロックを 2 つの 32 bit サブブロックに分割する。
（ステップ 3）サブブロックと 48 bit 拡大鍵を変数とする F 関数を含むラウンド関数を 16 回繰り返す。
（ステップ 4）32 bit のサブブロックを結合して再び 64 bit のデータブロックとし、最初の転置変換の逆変換を行う。

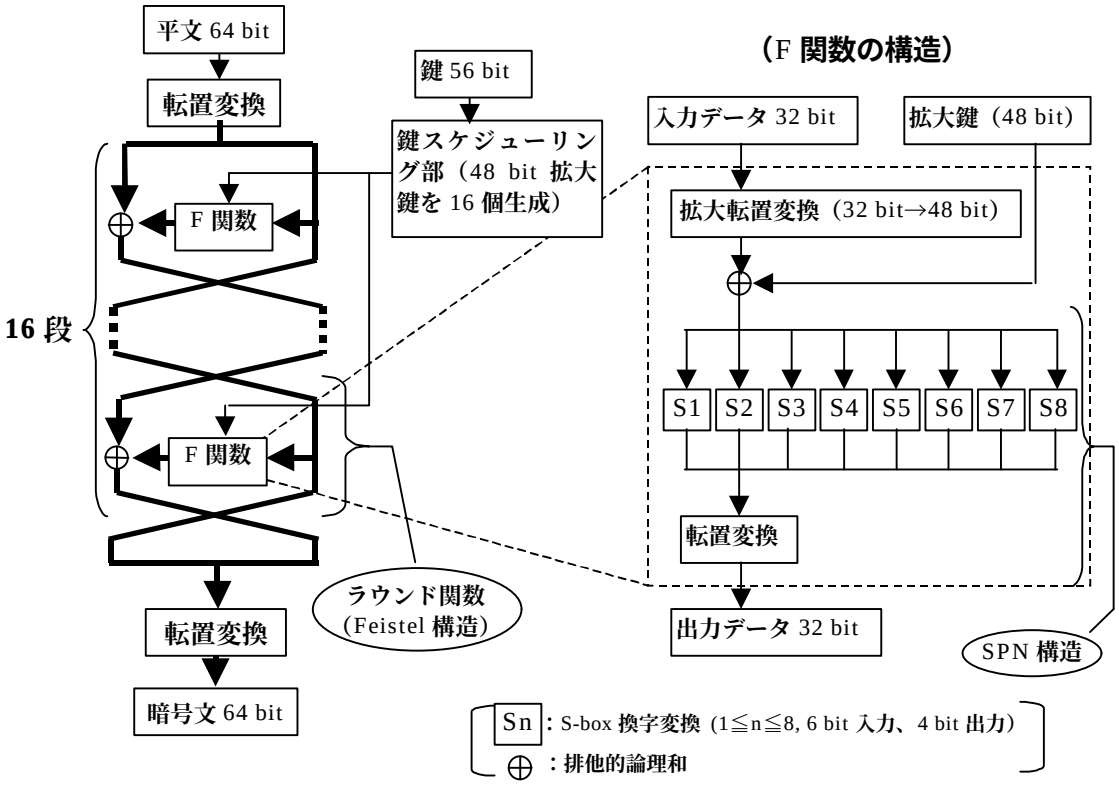


図 9　DES のデータランダム化部と F 関数の構造

復号は、同じ鍵の下で暗号化の手順をちょうど逆に辿ることにより実現される。DES のラウンド関数には Feistel 構造が採用されている。

ラウンド関数に含まれる F 関数は、DES のアルゴリズムにおける唯一の非線形変換手段であり、内部構造には SPN 構造が採用されている。F 関数の変換手順は以下の通り。

（ステップ 1）32 bit サブブロックに対し、一部の bit 値をコピーしてサブブロックを 48 bit に拡大するとともに転置変換を行う。
（ステップ 2）48 bit サブブロックを 48 bit 拡大鍵との排他的論理和によって変換する。
（ステップ 3）48 bit サブブロックを 8 個の 6 bit データに分割し、8 種類の S-box と呼ばれる換字変換（換字表を利用）によって 4 bit データに変換する。
（ステップ 4）8 個の 4 bit データを結合して 32 bit サブブロックとし、転置変換を行う。

②鍵スケジューリング部

鍵スケジューリング部では、56 bit の鍵から 48 bit の拡大鍵が 16 個生成され、データランダム化部の各ラウンド関数に 1 個ずつ入力される（図 10 参照）。

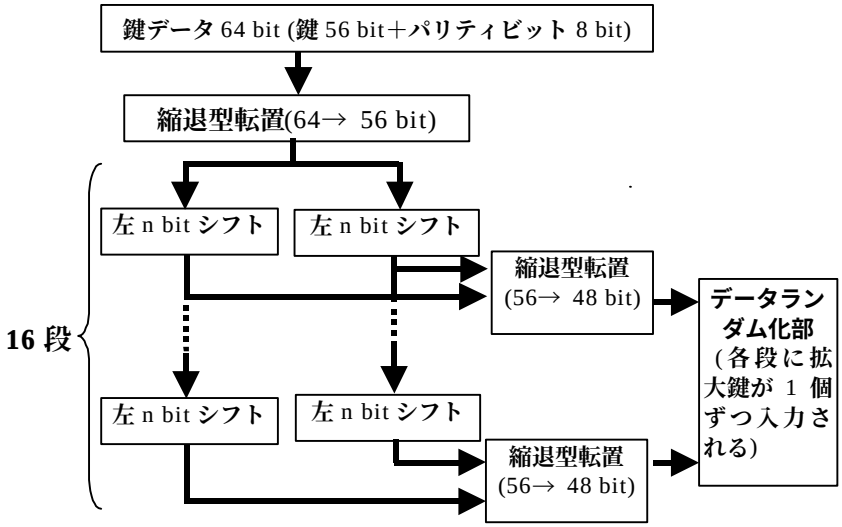


図 10　DES の鍵スケジューリング部の構造

拡大鍵は以下の手順で生成される。

（ステップ 1）当初 56 bit の鍵には 8 bit のパリティビットが付加されているが、縮退型転置変換によってパリティビットが削除されて 56 bit に圧縮されると同時に、転置変換が行われる。

（ステップ 2）56 bit の鍵は 28 bit ずつに分割され、それぞれ左 bit シフト変換が行われる。bit のシフト数は段数に依存し、1、2、9、16 段目では左 1 bit シフト、それ以外の段では左 2 bit シフトとなる。

（ステップ 3）2 つの 28 bit データは結合され、縮退型転置変換によって 48 bit に圧縮されると同時に、転置変換が行われる。

（ステップ 4）48 bit データは拡大鍵となり、データランダム化部の F 関数に入力される。

(2) 安全性に関する評価結果

これまで、様々な解読法に対する DES の安全性が研究・評価されてきたが、現在最も脅威となっている攻撃法は Brute Force Method である。1998 年 7 月に行われた RSA 社主催の DES 解読コンテストでは、約 25 万ドルの予算で製作された DES 解読専用装置によって、56 時間で解読されている（詳細は後述）。

Short Cut Method に対する安全性の理論的研究も数多く行われており（33 頁の表 2 参

照)、差分解読法を適用する場合、2^{47} 個の選択平文と 2^{37} 回の暗号化処理に相当する計算量が必要である（Biham and Shamir[1993]）ほか、線形解読法を適用する場合には、2^{43} 個の既知平文と 2^{42} 回の暗号化処理に相当する計算量が必要であることが示されている（Matsui[1994]）。また、分割攻撃によって 13 段の DES を解読するためには、1.34×2^{40} 個の既知平文が必要であるとの試算結果も発表されている（浜出他[1998]）。

2. FEAL

(1) アルゴリズムの概要と構造

FEAL（Fast Data Encipherment Algorithm）は、1988 年に NTT が開発したブロック長 64 bit のブロック暗号（Miyaguchi, Shiraishi and Shimizu[1988]）であり、64 bit の鍵を利用する FEAL-N と 128 bit の鍵を利用する FEAL-NX が提案されている（図 11 参照）。現在では、CAFIS[10]のデータ暗号化手段等に利用されているほか、ANSWER-WEB にも利用されている[11]。

①データランダム化部

FEAL のラウンド関数には、DES 同様、Feistel 構造が採用されており、段数は 4、8、16、32 段が利用可能となっているが、安全性の観点から 32 段の FEAL-32X が推奨されている。FEAL-N の暗号化手順は次の通り。

（ステップ 1）最初に、64 bit の平文ブロックを、4 つの 16 bit 拡大鍵との排他的論理和によって変換し、2 つの 32 bit サブブロックに分割する。
（ステップ 2）右 32 bit サブブロックを、左 32 bit サブブロックとの排他的論理和によって変換する（ステップ 1 と 2 が初期変換）。
（ステップ 3）1 つの 16 bit 拡大鍵を変数とする F 関数を含むラウンド関数による変換を N 回繰り返す。
（ステップ 4）右 32 bit サブブロックを、左 32 bit サブブロックとの排他的論理和によって変換する。
（ステップ 5）2 つの 32 bit サブブロックを結合して 64 bit ブロックとし、4 つの 16 bit 拡大鍵との排他的論理和によって変換する（ステップ 4 と 5 が最終変換）。これが暗号文ブロックとなる。

---

[10] CAFIS（Credit And Financial Information System）は、クレジットカード会社、金融機関、クレジットカード加盟店等を結び、クレジットカードの決済情報や信用照会情報、銀行 POS の資金決済情報等を扱うネットワークシステムである。NTT データが管理・運営しており、1998 年 5 月現在で、クレジットカード会社や金融機関等約 1500 社が加盟しているほか、CAT 端末等の設置台数は約 50 万台となっている。

[11] ANSWER（Automatic Answer Network System for Electronical Request）は、NTT データが運営・管理している金融サービス専用ネットワークシステムであり、顧客端末と金融機関とを結び、各端末から金融機関への口座残高照会、振込・振替請求、取引照会等の金融ネットワークサービスを提供している。ANSWER-WEB は、ANSWER を用いてインターネットバンキングを実現するサービスである。

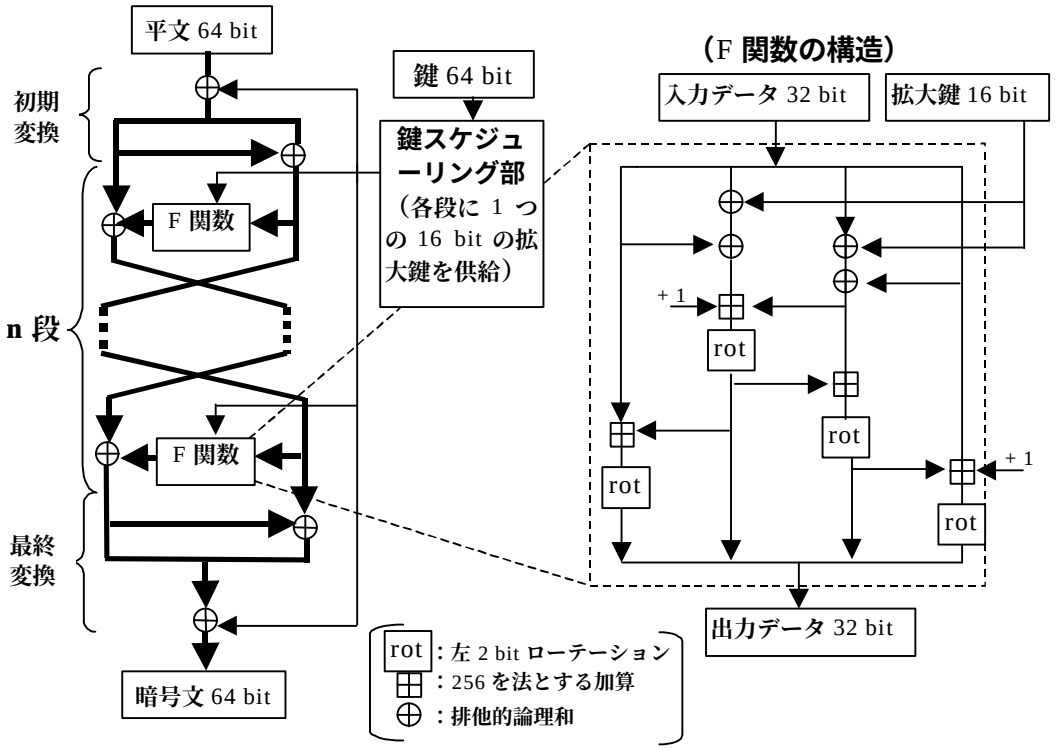


図 11　FEAL-N のデータランダム化部と F 関数の構造

DES ではラウンド関数での変換の前後に転置変換が行われているが、FEAL の場合、拡大鍵との排他的論理和による変換が採用されている。また、FEAL の F 関数内部の変換には、算術演算（左 2 bit ローテーション、256 を法とする加算）が利用されており、DES の F 関数で利用されているテーブル参照型の S-box に比べて少ないメモリー量での実装が可能となっている。F 関数では、まず 32 bit のサブブロックが 4 つの 8 bit データに分割され、それぞれの 8 bit データは 2 種類の算術演算および排他的論理和によって変換された後、再び 32 bit サブブロックに結合される、という手順で変換される。16 bit の拡大鍵は 1 つの F 関数に 1 つ入力され、2 つの 8 bit データに分割された後、排他的論理和変換に利用される。

②鍵スケジューリング部

鍵スケジューリング部は、入力される鍵 64 bit を基にして 16 bit の拡大鍵を（n+8）個生成する。構造はデータランダム化部と類似しており、FK 関数を含むラウンド関数が (n+8)/2 回繰り返される（図 12 参照）。FK 関数も、データランダム化部の F 関数と類似の構造を有しており、左 2 bit ローテーションと法 256 の加算という 2 種類の算術演算を変換手段として利用している。

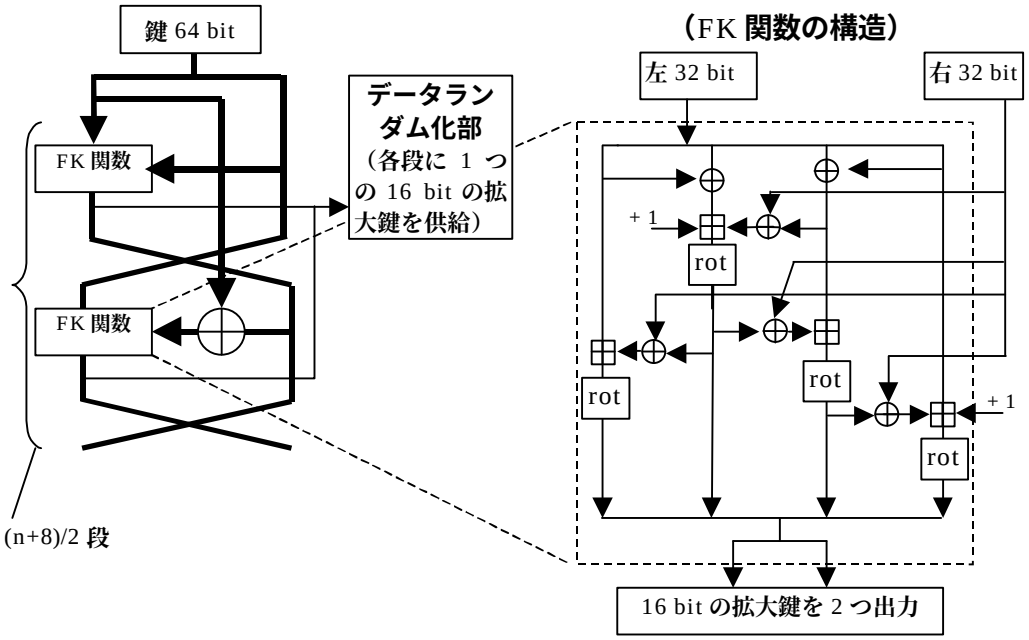


図 12　FEAL-N **の鍵スケジューリング部**

（2**）安全性に関する評価**

FEAL-N **は、鍵長とブロック長が** 64 bit **であり、全数探索法によって真の鍵を見つけるためには平均** 2^{63} **回の暗号化処理が必要となる。**

差分解読法に対する安全性については、差分特性確率を利用した近似的な評価により、FEAL-16 **の解読には** 2^{29} **個の選択平文が必要であるほか、**FEAL-32 **の解読には** 2^{66} **個の選択平文が必要となることが示されている（**Biham and Shamir[1993]**）。**

線形解読法については、FEAL-8 **は、** 2^{24} **個の既知平文によって解読可能であることが示されている（**Biham[1995]**）。しかし、**FEAL-16 **は、近似的な評価から** DES **とほぼ同等の安全性を有していることが示されており（**Ohta and Aoki[1994]**）、**FEAL-32 **は線形解読法に対して十分な安全性を有しているとみられている。**

3. LOKI91

（1**）アルゴリズムの概要と構造**

LOKI91 **は、**1991 **年に** Brown **らによって提案された鍵長とブロック長が** 64 bit **のブロック暗号（**Brown, Kwan and Pieprzyk[1993]**）である。**LOKI91 **は** 1990 **年に提案された** LOKI89 **と呼ばれる暗号方式（**Brown, Pieprzyk and Seberry[1990]**）を原形としている。** 14 **段以下の** LOKI89 **が差分解読法に対して十分な安全性を有しているとはいえないことが示された（**Biham and Shamir[1992]**）ため、アルゴリズムの改良が行われ、その結果開発されたのが** LOKI91 **である。**LOKI91 **では、差分解読法に対する安全性を高めるために、（**i**）データランダム化部の最初と最後の排他的論理和変換の削除、（**ii**）換字変換の一**

部変更、(iii) 鍵スケジューリング部の bit シフト量の変更、が行われた。

①データランダム化部

LOKI91 のデータランダム化部は Feistel 構造を有しており、64 bit の平文ブロックは 2 つの 32 bit サブブロックに分割された後、ラウンド関数に入力される。ラウンド関数と F 関数の構成は DES と類似している（図 13 を参照）。F 関数（32 bit 入出力）は SPN 構造を有しており、以下の手順で変換される。

（ステップ 1）32 bit の拡大鍵との排他的論理和によって変換された後、拡大転置変換によって 32 bit から 48 bit に拡大される。
（ステップ 2）48 bit サブブロックは、4 つの 12 bit データに分割された後、S-box（12 bit 入力、8 bit 出力）によって換字変換される。
（ステップ 3）4 つの 8 bit データは結合されて 32 bit サブブロックとなり、転置変換を経た後出力される。

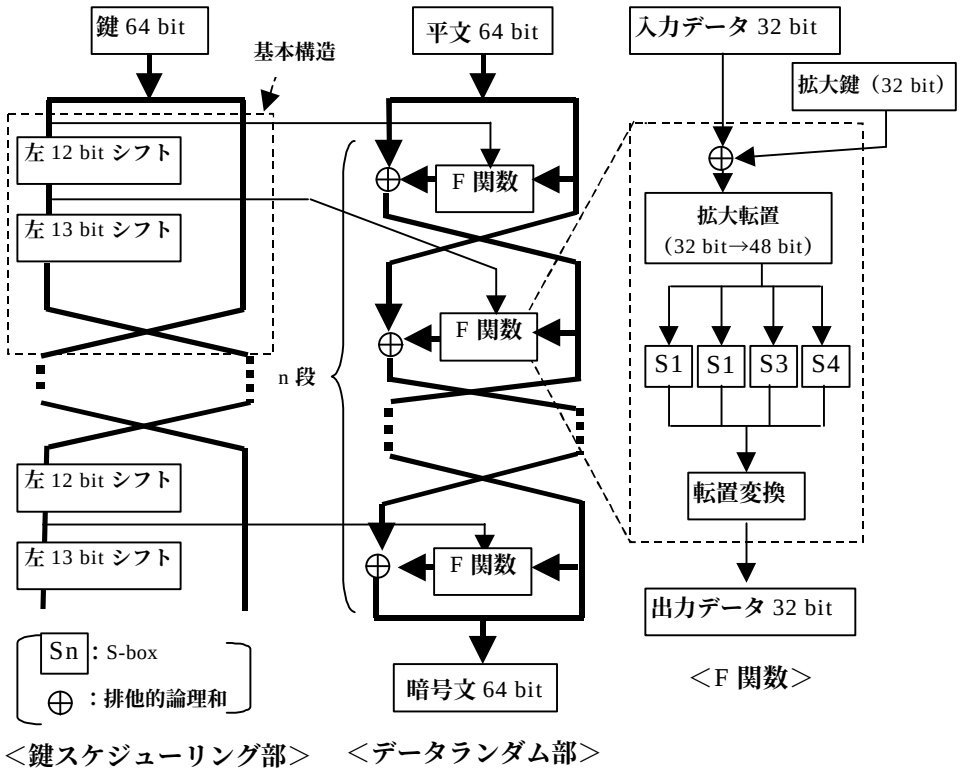


図 13　LOKI91 の暗号化アルゴリズムの構造

LOKI91 で採用されている S-box は 12 bit 入力 8 bit 出力となっており、DES に比べて大きいほか、4 つとも同一となっている。利用されている演算は、加算、乗算、排他的論理和、べき乗演算である。

②鍵スケジューリング部

LOKI91 の鍵スケジューリング部は、64 bit 鍵から 16 個の 32 bit 拡大鍵を生成する（図 13 参照）。64 bit 鍵は 2 つの 32 bit サブブロックに分割され、左 32 bit のサブブロックが 2 種類の bit シフト（左 12 bit シフト、左 13 bit シフト）によって変換される過程で 2 つの拡大鍵が生成された後、2 つのサブブロックの位置が入れ替えられる。これが鍵スケジューリング部の基本構造となっており、この基本構造が 6 回繰り返かえされる。

（2）安全性に関する評価

全数探索法に対しては、LOKI91 の鍵長が 64 bit であることから、解読には 2^{64} 回の暗号化処理が必要となる。

差分解読法や線形解読法に対しては、14 段以上の LOKI91 は近似的な評価から十分な安全性を有しているとみられている（Tokita, Sorimachi and Matsui[1995]）。13 段の LOKI91 を差分解読法によって解読するためには 2^{64} 個以上の選択平文が必要であることが示されているほか、14 段の LOKI91 を線形解読法によって解読するためには 2^{86} 個以上の既知平文が必要であることが示されている。一方、関連鍵攻撃によって、全数探索法よりも少ない計算量（1.375×2^{61} 回の暗号化処理）で解読が可能となることが示されているが（Biham[1994]）、関連鍵攻撃が適用できるケースは非常に限定されていることから実際の脅威にはならないとみられている。

4. IDEA

（1）アルゴリズムの概要と構造

IDEA（International Data Encryption Algorithm）は、1991 年に Lai と Massey がスイスの Ascom 社と共同開発したブロック暗号（Lai, Massey and Murphy[1991]）であり、ブロック長 64 bit、鍵長 128 bit、段数 8 段のブロック暗号である。

①データランダム化部

IDEA のデータランダム化部は、DES や FEAL 等の Feistel 構造とは異なる構造となっており、64 bit 平文ブロックを４つの 16 bit サブブロックに分割した上で、8 段のラウンド関数と最終変換によって暗号化を行う仕組みとなっている（図 14 参照）。ラウンド関数は、16 bit のサブブロック単位で変換を実行し、変換手段には 2^{16} を法とする加算、$(2^{16}+1)$ を法とする乗算、排他的論理和が利用されている。最終変換には、ラウンド関数の一部分が利用されている。

②鍵スケジューリング部

鍵スケジューリング部では、1 つのラウンド関数に 16 bit 拡大鍵 6 個が入力されるほか、最終変換には 4 個の拡大鍵が利用されることから、128 bit 鍵から 16 bit の拡大鍵が 52 個

（=6 個×8 段＋4 個）が生成される。拡大鍵は、次の手順で生成される。

（ステップ 1）128 bit 鍵を 16 bit ごとに分割し、まず 8 個の拡大鍵を生成する。
（ステップ 2）128 bit 鍵を左に 25 bit だけ巡回シフトし、変換後のブロックを 16 bit ごとに分割して 8 個の拡大鍵を生成する。この手続きを 6 回繰り返して、44 個の拡大鍵を生成する。

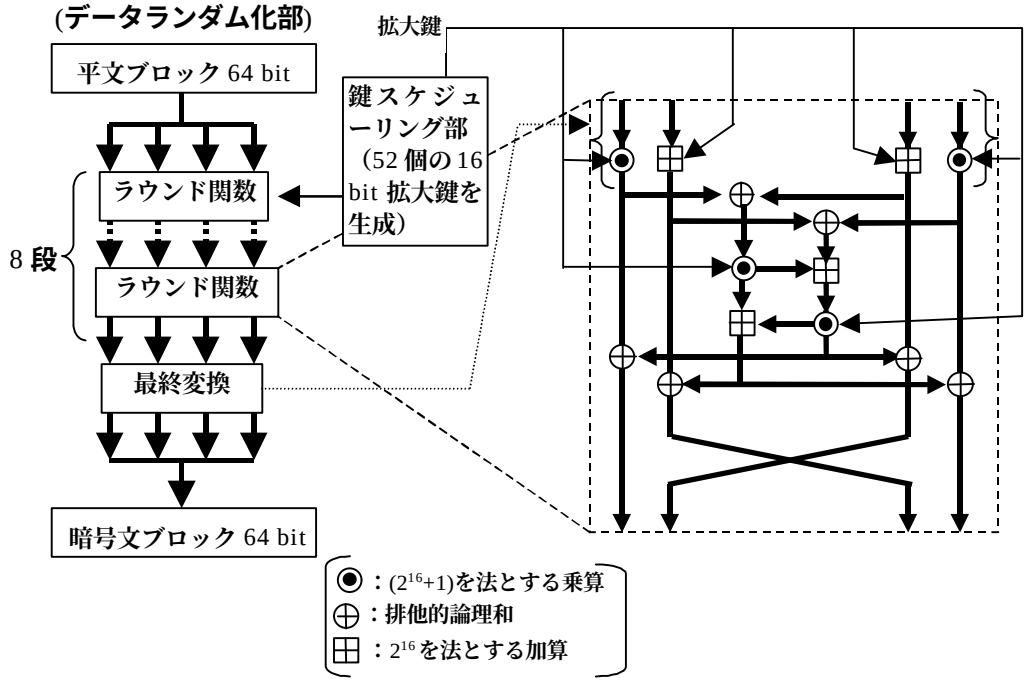


図 14　IDEA 暗号のラウンド関数の構造

なお、復号の際には、データランダム化部には変更は不要だが、鍵スケジューリング部に若干の変更が必要となる。

(2) 安全性に関する評価

IDEA は鍵長が 128 bit であることから、全数探索法で真の鍵を見つけるためには 2^{128} 回の暗号化処理が必要となる。

IDEA は差分解読法に対する安全性を考慮して設計されており、ラウンド関数 1 段の差分特性確率は高々2^{-18} 程度となるほか、3 段の差分特性確率は 2^{-64} を下回ることから、厳密な評価ではないものの、差分解読法に対して十分な安全性を有しているとみられている（Lai, Massey and Murphy[1991]）。最近では、3.5 段の IDEA に対して、2^{56} 個の特殊な選択平文と 2^{67} 回の暗号化処理によって 80%以上の確率で解読可能であるほか、3 段の IDEA に対しては、2^{29} 個の選択平文と 2^{44} 回の暗号化処理によって解読可能であるとの研究成果も示されている（Borst, Knudsen and Rijmen[1997]）。もっとも、8 段の IDEA に

対して適用可能かどうかは示されていない。一方、線形解読法に対しては、近似的な評価から十分な安全性を有するとみられている（Harpes, Cramer and Massey[1995]）。

また、IDEA には、鍵の特定の 69 bit がすべて 0 となる 2^{59} 個の弱鍵が存在し、弱鍵の推定には、2^{39} 個の選択平文と 2^{45} 回程度の暗号化処理量が必要であることが示されている（三上・金子[1997]）。

5. SAFER

（1）アルゴリズムの概要と構造

SAFER（Secure And Fast Encryption Routine）は、1994 年に Massey と米 Cylink によって共同開発されたブロック暗号であり、鍵長とブロック長が 64 bit の SAFER K-64（Massey[1994]）、128 bit の SAFER K-128（Massey[1995]）のほか、これらの暗号方式の鍵スケジューリング部に改良が加えられた SAFER SK-64 と SAFER SK-128 が提案されている。以下では、SAFER K-64 を例に説明する。

①データランダム化部

SAFER のデータランダム化部は、複数回繰り返されるラウンド関数と最終変換によって構成されており、段数は 6 段、8 段、10 段、12 段のいずれかが利用可能となっているが、8 段以上での利用が推奨されている（図 15 参照）。64 bit 平文ブロックは、最初に 8 個の 8 bit サブブロックに分割された後にラウンド関数に入力され、ラウンド関数内部では以下の 6 つのステップによってサブブロック単位で変換される。

（ステップ 1）64 bit の拡大鍵は 8 つの 8 bit データに分割され、排他的論理和と 256 を法とする加算による変換が行われる。

（ステップ 2）2 種類の換字変換（$S(x)=45^X \bmod 257$ と $S'(x)=\log_{45}X \bmod 257$）が行われる。変換には換字表が利用されている。

（ステップ 3）ステップ 1 と同様の変換であり、排他的論理和と加算の位置が入れ替えられている。

（ステップ 4～6）関数 f による変換と 4 bit サブブロック単位での転置変換が、それぞれ 3 回行われる。関数 f は、入力される左サブブロックを L、右サブブロックを R とすれば、f(L,R)=(2L+R, L+R)と表される（Pseudo-Hadamard 変換と呼ばれている）。

所定の回数のラウンド関数による変換終了後、最終変換が行われるが、これはラウンド関数のステップ 1 の部分に対応しており、排他的論理和と加算を組み合わせた構造となっている。

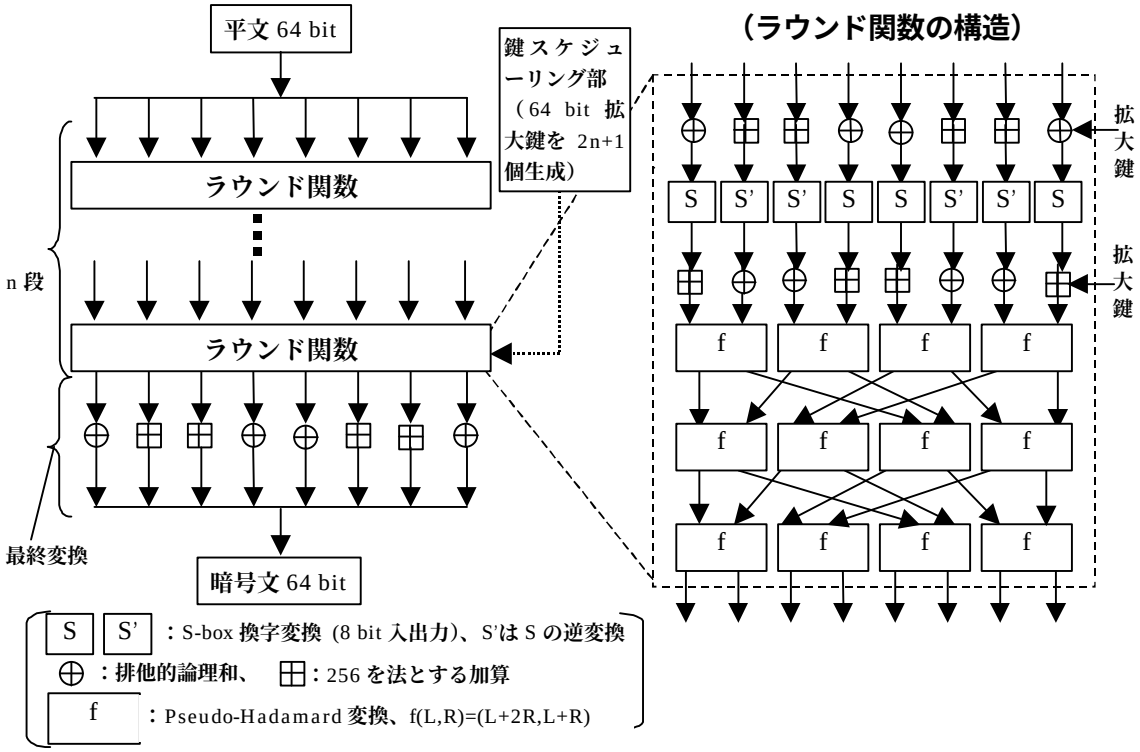


図 15　SAFER K-64 **のデータランダム化部とラウンド関数の構造**

②鍵スケジューリング部

鍵スケジューリング部は、段数を n **とすると、**64 bit **の鍵から（**2n+1**）個の** 64 bit **拡大鍵を生成する。拡大鍵の生成手順は次の通り。**

（ステップ 1**）最初に** 64 bit **鍵を第一の** 64 bit **拡大鍵とする。**
（ステップ 2**）**64 bit **鍵を** 8 **つの** 8 bit **データに分割する。**
（ステップ 3**）**8 bit **データをそれぞれ左** 3 bit **シフトした後に、予め用意された定数との法** 256 **の加算によって変換し、変換結果を結合して拡大鍵とする。**
（ステップ 4**）上記手続きを** 2n **回繰り返し、**2n+1 **個の拡大鍵を生成する。**

SAFER SK-64 **は、上記の** SAFER K-64 **の鍵スケジューリング部を改良した方式である。改良の内容は、①ステップ** 2 **と** 3 **の間に、**8 **個の** 8 bit **データの排他的論理和を計算して** 9 **番目の** 8 bit **データを生成するステップを挿入する、②ステップ** 3 **において定数との加算を行う前に、変換に利用する各** 8 bit **データの位置を入れ替えるという手続きを加える、という** 2 **点である。**

(2) 安全性に関する評価

SAFER **には、ブロック長・鍵長が** 64 bit **および** 128 bit **の** SAFER K-64 **と** SAFER K-

128 のほか、それぞれの鍵スケジューリング部に改良が加えられた SAFER SK-64 と SAFER SK-128 の合計 4 種類がある。SAFER K-64 と SK-64 を全数探索法によって解読するためには 2^{64} 回の暗号化処理が必要となるほか、SAFER K-128 と SK-128 を全数探索法によって解読するためには 2^{128} 回の暗号化処理が必要となる。

差分解読法に対しては、5 段の SAFER K-64 が、選択平文 2^{46} 個と 2^{35} 回の暗号化処理によって解読可能であることが示されている（Knudsen and Berson[1996]）。しかし、8 段以上の SAFER K-64 や K-128 は、差分解読法に対して十分な安全性を有しているとみられているほか、線形解読法に対しても近似的な評価から十分な安全性を有しているとみられている（Harpes, Cramer and Massey[1995]）。

また、SAFER K-64 に対して、ある特別の関係を有する 1 組の鍵ペアと 2^{47} 個の選択平文を利用した関連鍵攻撃によって、鍵の 8 bit 分のデータを入手できるという分析結果が示されている（Knudsen[1996]）ほか、2^{8} 個の鍵ペアと 2^{29} 個の選択平文によって鍵の 28 bit のデータを入手可能であるとの分析結果も示されている（Kelsey, Schneier and Wagner[1996]）。これらの分析結果を基に鍵スケジューリング部に改良が加えられた SAFER SK-64 に対して、同様の攻撃が成立するかどうかに関する研究成果はこれまでのところ発表されていない。

6. RC5

（1）アルゴリズムの概要と構造

RC5 は、1994 年に Rivest によって提案されたブロック暗号（Rivest[1995]）であり、ブロック長、鍵長、段数とも可変となっている。ブロック長は 32、64、128 bit が利用可能であるほか、鍵長は 8 bit 単位で上限 2,048 bit までが利用可能となっている。段数は上限 255 段までが利用可能であり、ブロック長 64 bit、鍵長 128 bit、段数 12 段の組み合わせ、もしくはブロック長 128 bit、鍵長 128 bit、16 段の組み合わせが一般的とされている。以下では、ブロック長 64 bit、鍵長 128 bit、段数 12 段の RC5 を例に説明する。なお、RC5 では、ブロック長 2w bit、鍵長 b byte、段数 r の場合、「RC5-w/r/b」との標記が一般的に使われており、以下で説明する RC5 は RC5-32/12/16 と表される。

①データランダム化部

データランダム化部においては、平文ブロックを 2 個の 32 bit サブブロックに分割し、初期変換を行った後、ラウンド関数を 12 回繰り返すことによって暗号化を実施する（図 16 参照）。初期変換では、2 つのサブブロックは、2 個の 32 bit の拡大鍵との排他的論理和によって変換される。ラウンド関数は、Feistel 構造でも SPN 構造でもなく、排他的論理和、2^{32} を法とする加算、データ依存型 bit シフトによって構成されており、加算とデータ依存型 bit シフトによって、ラウンド関数の出力データの非線形性を高めている。RC5 で利用されているデータ依存型 bit シフトは、「入力データの下位 5 bit の値だけ左 bit シフトさ

せる」というものである。

②鍵スケジューリング部

鍵スケジューリング部では、128 bit の鍵から、2 つの定数との加算、bit シフト等の演算によって 32 bit 拡大鍵が合計 26 個生成される。特に、ある段の拡大鍵が次の段の拡大鍵の生成に利用される点が特徴である。拡大鍵は初期変換部に 2 個入力されるほか、各ラウンド関数に 2 個ずつ入力される。

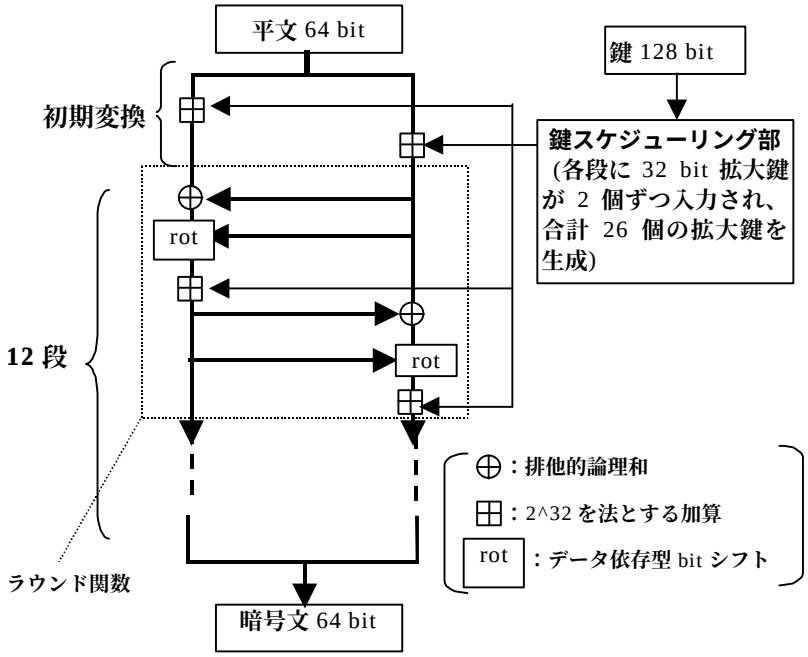


図 16　RC5-32/12/16 のデータランダム化部の構造

(2) 安全性に関する評価

RC5 の差分解読法に対する安全性については、近似的な評価において、RC5-32/12/16 を解読するために必要な選択平文数が 2^{44} 個であると見積もられており、これだけの数の選択平文を利用すればワークステーション 1 台を使って 1 分以内で解読可能であることが示されている（Biryukov and Kushilevitz[1998]）。また、RC5-32/12/16 に対して、条件付きの差分確率の最大値が $2^{-55.15}$ であることが示されている（反町・松井[1997]）。

線形解読法に対しては、RC5-32/6/16 における条件付きの線形確率の最大値が $2^{-49.15}$ となるほか、線形確率が 2^{-64} 以下となる最大の段数が 7.5 となることが示されており、7 段までであれば、全数探索法よりも線形解読法の方が有効となる可能性があることが示されている（反町・松井[1997]）。また、RC5-32/6/16 に対して線形解読法を適用するためには、2^{57} 個の既知平文が必要との研究成果も示されている（Biryukov and Kushilevitz[1998]）。

7. MISTY

(1) アルゴリズムの概要と構造

MISTY は、1995 年に三菱電機の松井らが発表したブロック暗号（松井[1996]）であり、ブロック長 64 bit、鍵長 128 bit、段数は可変（4 の倍数）となっている。MISTY の特徴は、差分解読法と線形解読法に対する安全性が数値的に保証されるように設計されている点である。

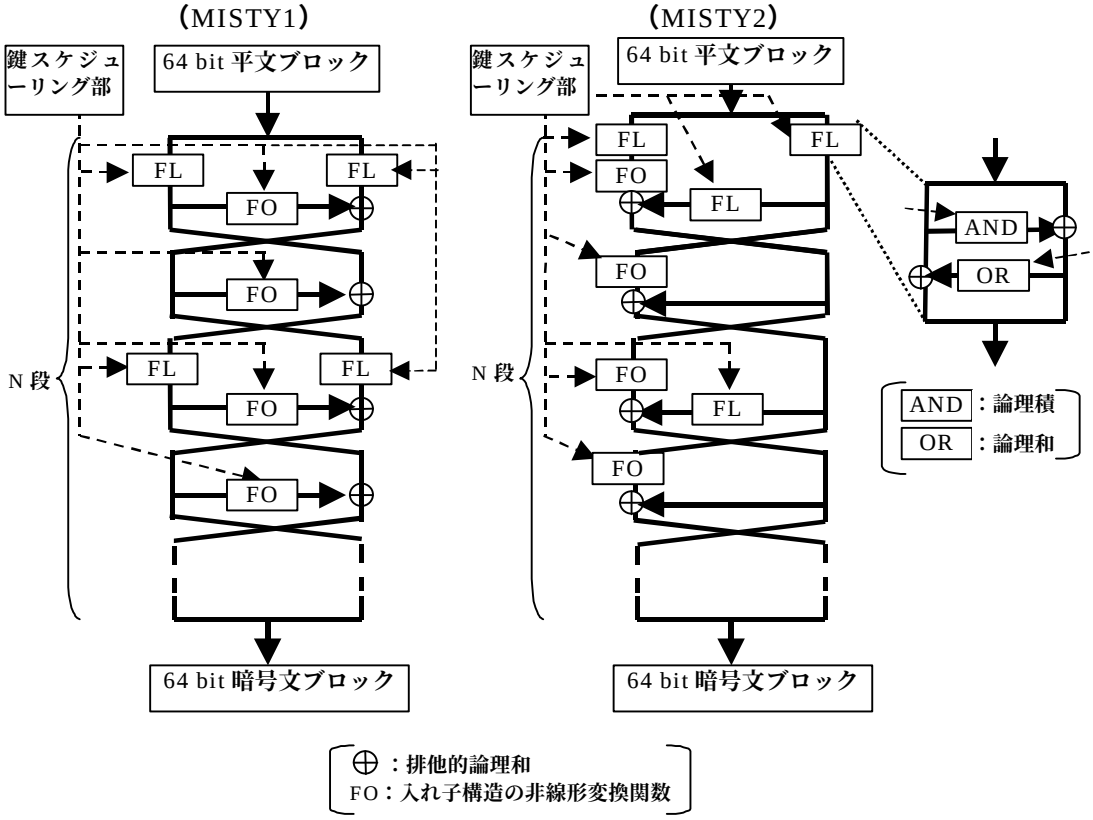


図 17　MISTY のデータランダム化部とラウンド関数

①データランダム化部

データランダム化部では、64 bit の平文ブロックが 32 bit のサブブロック 2 個に分割されて変換される。MISTY には、ラウンド関数の異なる 2 種類の暗号方式 MISTY1 と MISTY2 が提案されており、MISTY1 のラウンド関数には、FO 関数（F 関数に相当）と FL 関数を含む Feistel 構造が採用されている。MISTY2 のラウンド関数は、FO 関数の並列処理が可能なように設計されており、Feistel 構造とは異なっている（図 17 参照）。MISTY1 は 8 段、MISTY2 は 12 段で利用することが推奨されている。

FL 関数では、32 bit のサブブロックがさらに 16 bit の 2 つのデータに分割され、16 bit

拡大鍵との bit 単位での論理和および論理積によって変換された後、再び 32 bit のサブブロックに結合されて出力される。

また、FO 関数は、差分解読法および線形解読法に対して十分な安全性を確保できるように、MISTY2 のラウンド関数に類似した FO ラウンド関数を 3 段有する構造となっており、FO ラウンド関数内の FI 関数も FI ラウンド関数を 3 段有する構造となっている（図 18 参照）。このように、構造が類似している 3 段のラウンド関数が再帰的に利用されており、この点については MISTY1 と MISTY2 で共通となっている。FI 関数では、安全性に配慮するために 16 bit の入力データを 7 bit と 9 bit のデータに分割し、それぞれを 7 bit 入出力と 9 bit 入出力の換字変換 S-box によって変換する仕組みになっている。換字変換にはべき乗演算が採用されており、換字表が利用されている。

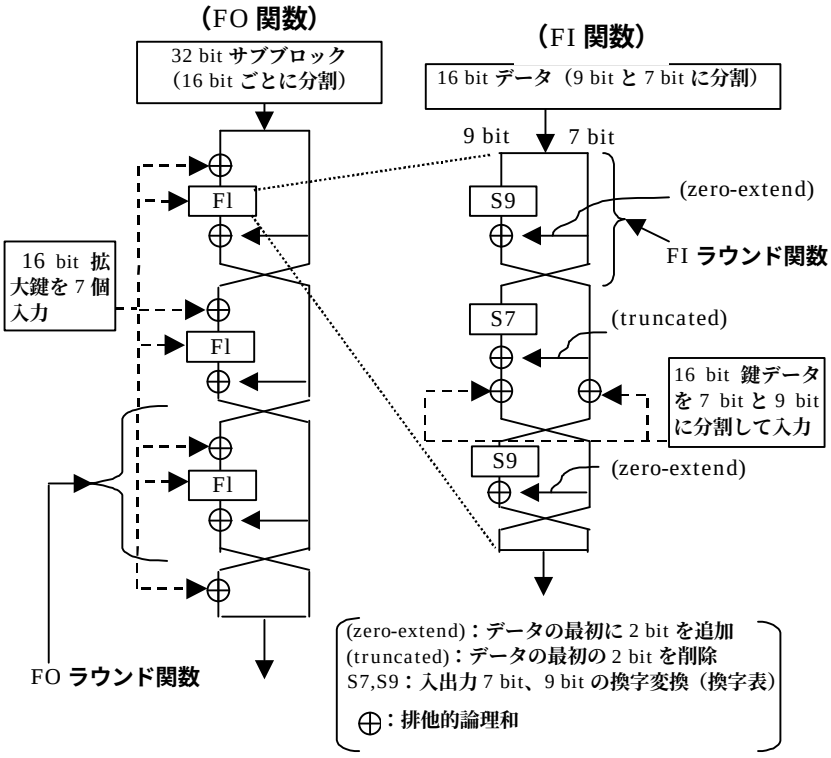


図 18　MISTY の FO 関数と FI 関数

②鍵スケジューリング部

鍵スケジューリング部は、実用的な観点からできるだけ簡素な構造となるように設計されており、FI 関数の構造をそのまま利用する形となっている。拡大鍵は、128 bit の鍵から段数に応じた数だけ生成される。

(2) 安全性に関する評価

MISTY の鍵長は、MISTY1、MISTY2 ともに 128 bit であり、全数探索法を使って解

読するためには 2^{128} 回の暗号化処理が必要となる。

また、MISTY は差分解読法と線形解読法に対する安全性を考慮して設計されており、平均差分確率および平均線形確率が 2^{-56} 以下となることが示されている（松井[1996]）。

なお、MISTY そのものではないものの、MISTY のアルゴリズムの一部に対する攻撃方法が提案されている。まず、FL 関数のない 4 段の MISTY1 に対して、36 個の選択平文を利用した高階差分攻撃によって解読が可能になることが示されている（久松・金子[1998]）ほか、5 段の MISTY に対しては、1,708 個の選択平文を用いた高階差分攻撃によって 2^{17} の計算量で解読可能であるとの研究成果が発表されている（田中・久松・金子[1998]）。また、FO 関数 3 段で構成される MISTY のアルゴリズムを解読するためには、2^{28} 個の選択平文が必要であることが証明されているが、FO 関数の入出力データを統計的に分析することによって、選択平文 222 個で解読に成功したとの実験結果も示されている（角尾・山田[1998]）。

表 2　既存の攻撃法に対する安全性評価結果

暗号方式	鍵長 (bit)	ブロック長 (bit)	全数探索法 (必要計算量)	差分解読法 （解読に必要な①選択平文数と②計算量、もしくは評価指標）	線形解読法 （解読に必要な①既知平文数と②計算量、もしくは評価指標）	高階差分攻撃	分割攻撃	弱鍵、関連鍵攻撃等その他の攻撃法
DES	56	64	2^56	・16 段：①2^47、②2^37	・16 段：①2^43、②2^42	———	13 段 ① 1.34 × 2^40	4 個の弱鍵、6 組の準弱鍵が見つかっている
FEAL-N（N:段数）	64	64	2^64	・32 段：①2^66、②2^64 以上 ・16 段：①2^29	・16 段：①2^39	———	———	———
LOKI91	64	64	2^64	・13 段：①2^64 以上	・14 段：①2^86	———	———	関連鍵攻撃 ①選択平文数：—— ②計算量：1.375×2^61
IDEA	128	64	2^128	・3.5 段：①2^56、②2^67 ・3 段：①2^29、②2^44 ・1 段の差分特性確率： 2^(-18)程度	———	———	———	弱鍵： 2^{59} 個存在 ①選択平文数：2^39 ②計算量：2^45
SAFER	64 or 128	64	K,SK-64: 2^64 K,SK-128: 2^128	・K64・5 段：①2^46、②2^35	———	———	———	K-64 への関連鍵攻撃 ①選択平文数：2^47
RC5-w/r/b[3]	可変 8～ 2,084	32, 64, or 128	2^64	・12 段：①2^44 ・12 段の最大差分確率[1]： 2^(-55.15)	・6 段：①2^57 ・6 段の最大線形確率[1]： 2^(-49.15)	———	———	———
MISTY 1,2	128	64	2^128	・平均差分確率：2^(-56)以下	・平均線形確率：2^(-56)以下	4 段[2]： 選択平文 36	———	———

（注）1. 探索した F 関数の差分値をある一定の範囲に限定して計測。
2. FL 関数のない 4 段の MISTY 1 に対する解読研究。
3. RC5 の安全性評価は、RC5-32/r/16（鍵長 128 bit、ブロック長 64 bit のバージョン）を想定したもの。

Ⅴ. DES の安全性を高めるための方策

1. DES の安全性低下を実証する試み

DES が標準化された当初、DES の安全性の高さは米国政府によって強く印象付けられた。標準化を担当した NBS[12]は、「1 秒間に 2 万 5000 個の鍵を検証可能なチップを開発できたとしても、並列処理に利用することができるのは高々1000 個程度であり、この計算機を利用しても全数探索法によって真の鍵を見つけるためには約 91 年かかる」との試算を発表した。また、DES の認定に際して安全性評価を担当した NSA[13]は、「DES のアルゴリズムについては、現時点（1976 年）で知り得る限り、いかなる統計的・解析的弱点も見つからなかった」との分析結果を発表した。こうした安全性評価によって、DES は米国政府標準暗号に認定され、金融分野を中心として幅広い分野において利用されるようになった。

一方、1977 年に、Diffie と Hellman は、NBS や NSA とは異なる分析結果を発表した（Diffie and Hellman[1977]）。その内容は、「1 秒間に 100 万個の鍵を検証可能なチップを設計し、このチップを 100 万個用いて並列処理を行えば 1 日で候補となるすべての鍵を探索することができる。こうした解読装置を構築するための費用は、現時点で約 2,000 万ドルとなるが、今後の技術進歩によって、近い将来現実的な費用で解読装置を製造できるようになると考えられる」というものである。解読装置の製造コスト逓減は、近年のコンピューターのコスト・パフォーマンス向上に伴って現実のものとなり、Brute Force Method に対する DES の安全性低下が多くの暗号研究者によって指摘されてきた。例えば、1993 年には、約 100 万ドルの予算によって平均 3.5 時間で解読できる専用装置を開発可能であることが Wiener によって示されている（Wiener[1993]）ほか、1996 年には、Blaze らによって、約 30 万ドルの予算によって平均 3 時間で解読に成功する専用装置を開発できることが示されている（Blaze et al.[1996]）。さらに、Kusuda と Matsumoto は、タイムメモリートレードオフ法を利用して、1 時間以内に 50%の確率で真の鍵を発見することが可能な解読装置を、現実的な費用で製作することが可能であることを示している（Kusuda and Matsumoto[1996]）。

こうした中、RSA 社は、全数探索法に対する DES の安全性を定量的に検証することを目的に、1997 年から 1998 年にかけて 3 回の懸賞金付き DES 解読コンテストを実施している（表 3 参照）[14]。これらの解読コンテストの内容は、CBC モードの DES によって変換された暗号文と初期値が RSA 社によって公表された後、暗号化に利用された鍵をどれだ

---

[12] NBS（National Bureau of Standard）：米国内における科学技術全般の標準規格の策定を担当する米商務省の下部組織であったが、1988 年に名称変更され、現在の NIST（National Institute of Standards and Technology）となっている。NBS は、米国政府内で利用する情報通信技術に関する標準規格 FIPS（Federal Information Processing Standards）の認定も担当しており、DES 等の暗号アルゴリズムの認定を行っていた。

[13] NSA（National Security Agency）：米国防総省の下部組織であり、国家安全保障の観点から国内外で通信情報の諜報活動を行っている。暗号技術についても高度な技術力を有しているとされており、FIPS 等の標準策定の際には NIST に対して助言を行っている。

[14] RSA 社の関連サイトの URL は、http://www.rsa.com/rsalabs/des2/。

け早く見つけることができるかを競う、というものである。候補となる鍵を用いて暗号文を復号し、与えられた平文が得られた場合、その鍵が真の鍵であることがわかる[15]。賞金は、第 1 回解読コンテストでは 1 万ドルとなっていたほか、第 2 回と第 3 回のコンテストでは、それぞれ①前回の解読時間の 1/4 未満で解読できた場合には 1 万ドル、②前回の解読時間の 1/4 以上 1/2 未満の場合には 5 千ドル、③前回の解読時間の 1/2 以上 3/4 未満の場合には 1 千ドル、となっていた。

表 3　全数探索法による DES の解読時間と費用（試算と実証）

	DES 認定当時の安全性評価		RSA 社・DES 解読コンテスト		
	NBS の主張（試算）	Diffie-Hellman の主張（試算）	第 1 回（解読結果）	第 2 回（解読結果）	第 3 回（解読結果）
試算発表時期（コンテスト開始日）	1976 年	1977 年 1 月	1997 年 1 月 28 日	1998 年 1 月 13 日	1998 年 7 月 13 日
解読方法	1 秒間に 2 万 5000 個の鍵を検証可能な専用チップを 1000 個用いて並列処理できる装置を想定	1 秒間に 100 万個の鍵を検証可能な専用チップを 100 万個用いて並列処理できる装置を想定	インターネット上でボランティアを募り、平均約 1 万台のコンピューターを動員して鍵を探索	インターネット上でボランティアを募り、平均約 4 万台[16]のコンピューターを動員して鍵を探索	1 秒間に約 5900 万個の鍵を検証可能な専用チップを約 1500 個用いて並列処理できる専用装置を利用
解読時間（鍵全体に対する鍵検証個数の割合）	約 91 年（100%）	約 20 時間（100%）	約 140 日（約 25%）	約 40 日（約 88%）	約 56 時間（約 25%）
1 秒あたりの平均鍵検証個数	約 2500 万個（1.5×2^{24}）	約 1 兆個（1.8×2^{39}）	約 17 億個（1.6×2^{30}）	約 184 億個（1.1×2^{34}）	約 888 億個（1.3×2^{36}）
解読に必要な費用	——	約 2000 万ドル	——	——	約 25 万ドル

第 1 回および第 2 回の解読コンテストでは、インターネットによってボランティアを募り、大量のコンピューターを同時に利用するという方法によって、それぞれ約 140 日、約 40 日で解読に成功している。さらに、第 3 回の解読コンテストにおいては、3 日足らずの間に DES を解読可能な専用装置を約 25 万ドルで構築できることが実証された（EFF[1998]）。この装置を研究・開発したのは、コンピューター・ネットワークにおける情報セキュリティやプライバシー確保等を目的とする米国の非営利団体 EFF（Electronic Frontier Foundation）の研究チームであり、有力な暗号研究者や技術者約 10 名が中心となっている。EFF は、「DES によって通信データの安全性を確保することはもはや不可能であることを実証するために、DES Cracker の研究・開発を進めてきた」と発表しており、EFF のホームページには、”EFF Builds DES Cracker that proves that Data Encryption

[15] RSA 社は平文の一部を予め公表している。平文は、”The unknown message is: ”（24 byte）というメッセージで始まり、RSA 社が考案した秘密のメッセージがこれに続く形となっている。したがって、平文の最初の 3 ブロック（1 ブロックは 8 byte）が明らかとなっていることから、候補となる鍵の 1 つで暗号文を復号したときに対応する平文が得られた場合、その鍵が真の鍵である可能性が極めて高い。

[16] これは、利用されたすべてのコンピューターの CPU が Intel Pentium 166 MHz と仮定したときの台数である。

Standard is insecure."と大きく記載されている[17]。なお、EFF は、この解読装置に関する技術的詳細を本として刊行しており、こうした技術情報は容易に入手可能となってきている（EFF[1998]）。

このように、3 回の解読コンテストによって、全数探索法に対する DES の安全性が急速に低下していることが実証されるとともに、DES に代わる暗号方式の必要性がより一層印象付けられた。

2. Triple DES の提案

(1) アルゴリズムの概要と構造

RSA 社の DES 解読コンテストの結果に象徴される DES の安全性低下の原因は、DES の鍵長が 56 bit と比較的短いことである。一方、DES のアルゴリズム自体は、これまでの研究成果によって、致命的な欠陥がなく優れた構造を有しているとの見方が一般的となっている。このため、DES のアルゴリズムを利用しながら全数探索法に対する安全性を高める方法として、1979 年に IBM の Tuchman によって DES の組み合わせ暗号 Triple DES が提案された（Tuchman[1979]）[18]。

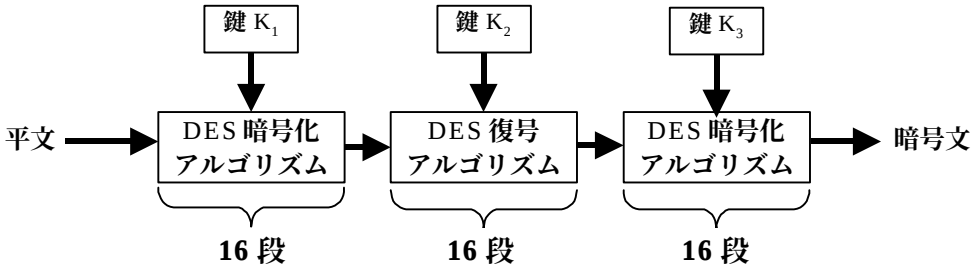


図 19　Triple DES のアルゴリズム

Triple DES は、異なる 2 つないしは 3 つの 56 bit の鍵を使って、DES のアルゴリズムを 3 回繰り返して暗号化・復号する方式である。Triple DES の変換手順は、「DES の暗号化アルゴリズム→復号アルゴリズム→暗号化アルゴリズム」という 3 段階の変換によって構成されるケースが一般的であり[19]、それぞれ異なる 3 つの鍵を利用する場合には 3-key

[17] EFF の関連サイトの URL は、http://www.eff.org/descracker.html。

[18] 組み合わせ暗号は、ある特定のアルゴリズムを複数回繰り返して暗号化する方式であり、繰り返しの回数だけ異なる鍵を利用することができるため、鍵長を長くして全数探索法に対する安全性を高めることができるとみられている。

[19] 「暗号化→復号→暗号化」という組み合わせ暗号のアルゴリズム構成は、銀行業務における鍵管理（ホールセール業務）の方法に関する標準規格 ISO 8732（Banking – Key management (wholesale)）において、管理対象の鍵や初期値の暗号化・復号方法の 1 つとして記載されてい

Triple DES、異なる 2 つの鍵を利用する場合（K_1=K_3 の場合）には 2-key Triple DES と呼ばれている（図 19 参照）。Triple DES は、3 つの鍵を同一にすると通常の DES と同じ暗号化・復号処理となることから、通常の DES との互換性を確保できるという特徴を有している。

Triple DES は、DES と比較すると、段数が 3 倍（48 段）となるほか、鍵長は 2-key Triple DES の場合には 2 倍（112 bit）となり、3-key Triple DES の場合は 3 倍（168 bit）に拡張される。ブロック長は、DES と同じく 64 bit である。

（2）Brute Force Method に対する安全性

①Merkle-Hellman 選択平文攻撃

Triple DES は、DES の鍵長を拡張する効果を有している。しかし、Triple DES は組み合わせ暗号であるため、選択平文攻撃によって実質的な安全性は鍵長が拡大するほど向上しないことが、Merkle と Hellman によって示されている（この解読法は Merkle-Hellman 選択平文攻撃と呼ばれている。Merkle and Hellman[1981]）。

Merkle-Hellman 選択平文攻撃を 2-key Triple DES に適用する場合、以下のような手順で攻撃が行われる（図 20 参照）。

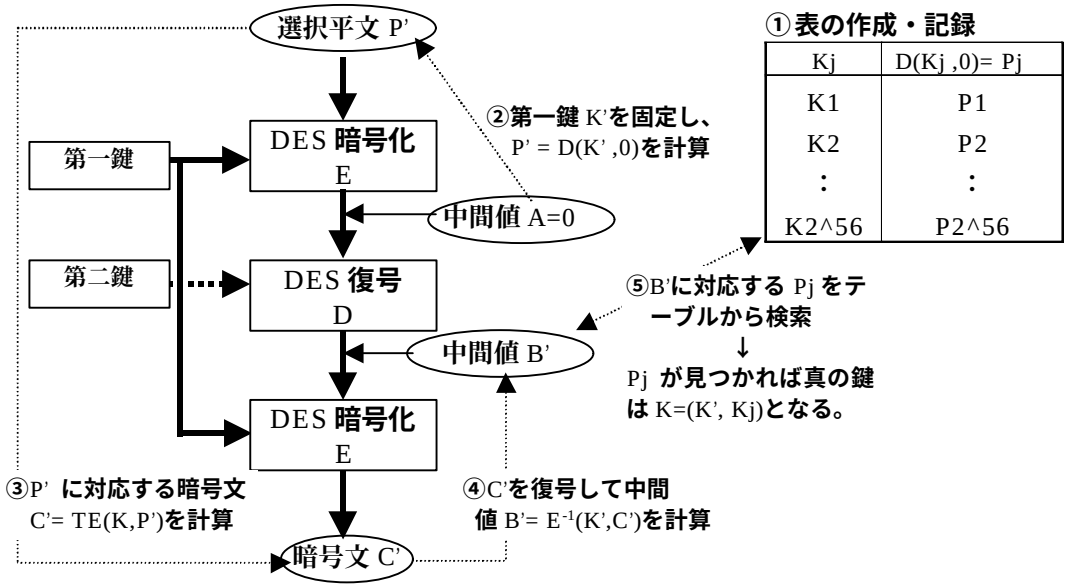


図 20　Merkle-Hellman 選択平文攻撃の概要

る。Annex C においては、例として、Triple DES を利用する方法について記載されている。

（i）最初の DES 暗号化ステップと次の DES 復号ステップとの間の値を中間値 A とし、まず A を 0 とおく。
（ii）最初と最後の DES 暗号化に利用する第一鍵 Kj（$j = 1,\ldots,2^{56}$）を用いて、DES の復号アルゴリズム D（暗号化アルゴリズムは E）によって 0 を変換し、その計算結果 $Pj = D(Kj,0)$（ただし $j = 1,\ldots,2^{56}$）を表として外部記憶媒体に記録する。したがって、必要となる計算量は 2^{56} 回の DES 復号に等しくなる。
（iii）第一鍵をある値 K'に設定し、この第一鍵に対して中間値 A が 0 となるような平文 P'を得るために、$P' = D(K',0)$を計算する。
（iv）設定した選択平文 P'に対する暗号文 $C' = TE(K, P')$を入手する（Triple DES の暗号化アルゴリズムを TE、真の鍵を $K=(K_1,K_2)$とする）。
（v）予め設定している第一鍵 K'を用いて暗号文 C'を DES のアルゴリズムで復号し、中間値 $B' = D(K', C')$を計算する。
（vi）ここで（ii）で作成した表から、$Pj = B'$を満足する Pj が存在するか否かをチェックする。このような Pj が存在する場合、真の鍵は $K= (K',Kj)$であることがわかる。一方、Pj が存在しない場合、第一鍵 K'を別の鍵に取り替えて、（iii）～（vi）の手続きを繰り返す。

この攻撃では、まず 2^{56} 回の DES の復号処理によって表を作成し、外部記憶媒体に記録した上で、2×2^{56} 回の DES の復号処理（鍵の候補は 2^{56} 個存在し、（iii）と（v）において DES の復号が必要となる）によって真の鍵を見つけることができる。したがって、必要な外部記憶媒体を用意することができれば、必要な計算量は 3×2^{56} となり、全数探索法（2^{112} 回の暗号化処理）に比べて大幅に計算量を削減することができる。

3-key Triple DES についても、同様の攻撃法により、表作成に必要な 2^{56} 回の DES 復号処理に加えて、2^{112} 回（上記例の（iii）と（v）の復号において、それぞれ独立に 2^{56} 回の復号処理が必要となる）の DES の復号処理によって解読が可能となる。したがって、必要な記憶媒体を用意できれば $2^{112}+2^{56}$ の計算量によって解読することができ、全数探索法の場合（2^{168} 回の暗号化処理）に比べて大幅に計算量を削減できる。なお、Lucks は、Merkle-Hellman 選択平文攻撃における処理回数を削減する方法を提案しており、2^{112} 回の復号処理を最大で $2^{108.2}$ 回まで削減できるとしている（Lucks[1998a]）。

Merkle-Hellman 選択平文攻撃は、Triple DES に限らず、あらゆる組み合わせ暗号に対して有効であることから、組み合わせ暗号によって鍵長を長くして Brute Force Method に対する安全性を向上させる試みには、一定の限界が存在するといえる。このことから、より高い安全性を確保するためには、組み合わせ暗号ではなく、より長い鍵長を有する単体の暗号アルゴリズムの利用が望まれる。

また、Oorschot と Wiener は、Merkle-Hellman 選択平文攻撃をベースに 2-key Triple DES に対する既知平文攻撃を提案しており（Oorschot and Wiener[1990]）、N 個の既知平文・暗号文ペアと $2^{120-\log_2 N}$ 回の DES による暗号化処理によって、2-key Triple DES を解読することが可能であることを示している。

②暗号文一致攻撃と辞書攻撃

Triple DES のブロック長は、DES と同様に 64 bit と比較的短い。したがって、暗号文一致攻撃や辞書攻撃に対する安全性は、DES と比較して全く向上していない。

暗号文一致攻撃は、バースデー・パラドックスを利用して一致する暗号文を検索し、平文に関する情報を入手するという方法である。Triple DES のブロック長は 64 bit であるから、2^{32} 個の暗号文を入手できれば約 0.5 の確率で同一の暗号文を見つけることが可能となり、それらの暗号文に対応する平文に関する情報を入手することが可能となる。また、辞書攻撃は、ある固定された鍵によって暗号化された暗号文と平文のペアを集めて「辞書」を作っておき、その後入手した暗号文に対応する平文をこの辞書を利用して見つけるという攻撃法である。したがって、ブロック長が短いほど辞書攻撃は有効となる。

こうした攻撃に対して安全性を向上させるためには、ブロック長を長くする必要がある。DES や Triple DES に限らず、FEAL、LOKI91、IDEA、SAFER、MISTY 等主要なブロック暗号のブロック長は 64 bit となっており、今後より長いブロック長を有する方式の利用が望ましい。

③Triple DES の利用モードに対する攻撃

ブロック長を変更しないで、Triple DES の暗号文一致攻撃や辞書攻撃に対する安全性を高める方法として、前述の 4 つの利用モードのほかに、これまで様々な利用モードが提案されている[20]。特殊な利用モードを適用することによって、変換手順をより複雑にして暗号文と平文の単純な関係を絶ち切り、たとえ一致する暗号文ブロックが見つかったとしても、容易に平文に関する情報が漏れないようにすることが可能となる。Triple DES の利用モードとして提案されている方式は、(i) 組み合わせ暗号の構造を生かして、変換の途中で発生する中間値を次のブロックの変換にフィードバックして利用する方式（CBCM モード等）と、(ii) Triple DES を 1 つのアルゴリズムとみなし、中間値をデータの攪拌に一切利用しない方式（TCBC-I モードや TCFB-P モード等）に大別することができる。

しかし、(i) の中間値を変換に利用するモードでは、その中間値を操作することによって、通常の Triple DES よりも大幅に少ない計算量によって解読が可能となるとする分析結果が発表されている。例えば、3-key Triple DES を CBCM モードにおいて利用する場合、2^{65} 個の選択平文を入手できれば、2^{58} 回の DES の暗号化処理によって解読可能であることが示されている[21]（Biham and Knudsen[1998]）。これらの分析結果から、Triple DES を利用する場合、その中間値を変換に利用するモードは、安全性の観点から使用しない方

[20] Triple DES の利用モードを規定している ANSI X9.52 には、Triple DES の ECB、CBC、OFB、CFB の 4 つの利用モード（それぞれ TECB、TCBC、TOFB、TCFB モードと呼ばれている）に加えて、TCBC-I、TCFB-P、TOFB-I という 3 つの利用モードが記載されている。これらの利用モードは、いずれも Triple DES を 1 つのアルゴリズムとして利用するものであり、変換途中の中間値を次のブロックの変換に利用したりフィードバックしたりする方式ではない。

[21] ANSI X9.52 には、CBCM モードは記載されていない。

がよいとの研究成果が発表されている（Biham[1998]）。

（3）Short Cut Method に対する安全性

Triple DES は、代表的な Short Cut Method である差分解読法や線形解読法に対して十分な安全性を有していることが示されている。Triple DES を 48 段の DES とみなし、最大差分特性確率と最大線形特性確率を計算すると、差分解読法を適用するためには少なくとも 2^{174} 個の選択平文が必要となるほか、線形解読法を適用するためには少なくとも 2^{118} 個の既知平文が必要となるとされている（Kusuda and Matsumoto[1997]）。これらの選択平文数および既知平文数は、メッセージ空間全体の 2^{64} 個（ブロック長が 64 bit であるため）を超えることから、差分解読法と線形解読法は理論的に適用不可能となっている。

また、関連鍵攻撃によって、3-key Triple DES が Merkle-Hellman 選択平文攻撃よりも少ない計算量によって解読可能であるとの研究成果が発表されている（Kelsey, Schneier and Wagner[1996]）。具体的には、1 組の選択平文・暗号文ペアとある特定の関係を有する 1 組の鍵ペアを利用することによって、2^{56}～2^{72} 回程度の DES の暗号化処理で解読できることが示されている。もっとも、この攻撃法は、2-key Triple DES には適用できないほか、攻撃可能な環境は極めて限定されているため、実際の脅威となるとの見方は少ない。

以上の Triple DES の安全性評価に関する分析結果をまとめると、以下の表 4 の通り。

表 4　Triple DES の主要な安全性評価結果（解読に必要な計算量）

	2-key Triple DES	3-key Triple DES
全数探索法	2^112	2^168
Merkle-Hellman **選択平文攻撃**	2^57 （選択平文数 2^56）	2^112 （選択平文数 2^56）
Lucks **による攻撃**	——	2^108.2
Oorshot-Wiener **既知平文攻撃**	$2^{(120-\log_2 N)}$ （既知平文数 N）	——
CBCM **モードにおける選択平文攻撃***	——	2^58 （選択平文数 2^65）
差分解読法	（選択平文数　2^174）	
線形解読法	（既知平文数　2^118）	
関連鍵攻撃*	——	2^56～2^72 （選択平文　1） （選択鍵ペア 1）

* CBCM モードにおける選択平文攻撃や関連鍵攻撃は、攻撃が成立する条件が非常に限定されていることから、実際の脅威にはならないとみられている。

3. Triple DES から AES へ

DES の安全性低下が深刻化する中、DES からの移行が容易である等の理由から、金融分野を中心に、幅広い分野において DES の代替暗号として Triple DES を採用する動きが増えている。一方、米国政府は、1997 年 1 月に次世代の米国政府標準暗号として AES

（Advanced Encryption Standard、詳細は後述）の策定に着手することを発表した。米国政府は、AES の標準化完了後、AES を米国政府標準暗号として 20～30 年の間利用する方針を発表しており、次世代の主要な暗号方式となる可能性が高い。米国政府の標準化スケジュールによると、AES の標準化は早ければ 2000 年にも完了する見通しであるが、AES がその安全性について高い信頼を得て、AES を利用した暗号製品が広く普及するまでには、標準化完了後さらに数年程度はかかるものとみられている。

AES が実際に利用可能になるまでの間、Triple DES が DES の後継暗号として金融分野を中心に幅広い分野において利用されるようになるとみられ、今後主要なブロック暗号は「DES→Triple DES→AES」と移行していくと考えられる。

Ⅵ. 次世代の米国政府標準暗号 AES

AES（Advanced Encryption Standard）は、現在米国政府が策定を進めている次世代の米国政府標準暗号の名称である。NISTは、これまで標準暗号として認定してきたDES（Data Encryption Standard、FIPS 46-2）の安全性低下が深刻化していることから、1997 年 1 月に AES のアルゴリズムを公募によって選定する方針を発表し、同年 9 月にはアルゴリズムの要件や評価基準等を公表している[22]。

AES のアルゴリズム選定プロセスの特徴は、候補となるアルゴリズムの設計基準や仕様のほか、NIST のアルゴリズムの評価尺度等関連情報をすべて公開するとともに、外部の暗号学者や研究者の分析・評価を参考にしてアルゴリズムの絞り込みを行うことによって、選定プロセスの透明性を高めている点である。NIST は、インターネット上に AES 関連のサイトを開設しており[23]、各アルゴリズムの詳細な仕様に関する情報や、暗号研究者による各アルゴリズムの分析結果に関する情報を入手することが可能となっている。

1. AES の候補アルゴリズムの要件、評価基準、標準化スケジュール

（1）アルゴリズムの要件

AES の候補アルゴリズムの要件[24]は、①共通鍵ブロック暗号であること、②鍵長は 128 bit、192 bit、256 bit のいずれも利用可能であること、③ブロック長は 128 bit が利用可能であること、④ロイヤリティ・フリーで使用できること、の 4 点である。

このように、鍵長とブロック長をより長くすることによって、全数探索法、暗号文一致攻撃、辞書攻撃に対して十分な安全性を確保できるように配慮されている。

（2）アルゴリズムの評価基準

アルゴリズムの評価基準は、①安全性（解読の困難性）、②コスト、③その他のアルゴリズムの特徴、の 3 点であり、この順序で優先順位が付けられている。NIST は、これらの基準に関する評価を行うにあたり、外部の暗号研究者や技術者による独自の分析結果の発表を参考にするとしている。

①安全性

安全性は最も重要な評価基準とされており、主に以下の 3 点について分析される。ただし、最終的な各アルゴリズムの安全性に関する評価は、NIST 自身による各アルゴリズムの分析結果だけではなく、外部の暗号研究者の分析結果も十分参考にして決定されることになっている。

---

[22] NIST の発表内容については、宇根[1997]、NIST[1997]を参照。

[23] NIST の AES 関連サイトの URL は、http://www.nist.gov/aes。

[24] アルゴリズムの要件、評価基準、標準化スケジュールは、NIST の公表資料（NIST[1997], Smid and Dworkin[1998]）から引用している。

（i）暗号文のランダム性[25]

（ii）安全性に関する理論的根拠

（iii）評価プロセスにおいて指摘された安全性に関する問題点

②コスト

コストについては、主に以下の 2 点について測定・評価される。

（i）鍵のセットアップ、暗号化、復号等の処理速度

（ii）実装に必要となるメモリー容量

コストに関する評価は、2 回の技術的評価ラウンドに分けて行われる。技術的評価第 1 ラウンドでは、NIST は、ソフトウェアによる実装（鍵長・ブロック長ともに 128 bit に設定）での処理速度等を測定・評価する[26]。技術的評価第 2 ラウンドでは、NIST は、ソフトウェアによる実装（上記以外の鍵長・ブロック長の組み合わせ）での処理速度等を測定・評価するほか、8 bit プロセッサーやハードウェアによる実装についても、各アルゴリズムの提案者および外部の技術者から寄せられた測定結果を参考に評価を行う。

③その他のアルゴリズムの特徴

その他の評価基準として、（i）様々なアプリケーションへの利用可能性、（ii）ハードウェアやソフトウェアへの適用性、（ii）構造の単純性等が挙げられている。様々なアプリケーションへの利用可能性については、例えば、メッセージ認証コード（MAC）、疑似乱数生成装置やハッシュ関数等に利用可能かどうか等について評価される。また、ハードウェアやソフトウェアへの適用性に関しては、そのアルゴリズムがハードウェアとソフトウェアのどちらで実装するのに向いているか等が評価されることとなっている。

（3）標準化スケジュール

アルゴリズムの標準化スケジュールは、以下の表 5 の通り。現在、NIST や暗号研究者によって各アルゴリズムの分析・評価が進められており、早ければ 2000 年内にも、AES のアルゴリズムが決定・発表される見込みである。

---

[25] 具体的には、ある平文データに対応する暗号文データと、その平文データの bit をランダムに転置変換して得られるデータを生成し、両者の間の類似性を分析することにより、そのアルゴリズムによって生成される暗号文にどの程度偏りが生じているかを評価するもの。

[26] NIST が採用するソフトウェアの実装環境は、CPU：Pentium Pro 200 MHz、メモリー：64 MB RAM、OS：Windows95、PC 機種：IBM PC 互換機、プログラム言語：Borland C++ 5.0 および JAVA JDK となっている。

表 5 アルゴリズムの標準化スケジュール

フェーズ	日程	作業内容等
アルゴリズムの募集	1998/6/15 最終締切	NIST は、提出資料に不備がないかどうかをチェックし、候補となるアルゴリズムを決定する。
技術的評価第 1 ラウンド	1998/8/20 ～1999/4/15	NIST は、暗号研究者から寄せられたアルゴリズムの分析結果を参考にして、安全性や処理速度等について分析・評価を行う。
第 1 回 AES 候補コンファレンス	1998/8/20 ～ 8/22	各アルゴリズム提案者が自分のアルゴリズムの説明を行い、参加者からコメントを得る。NIST は、各アルゴリズムの詳細に関する情報を各参加者に提供し、アルゴリズムの分析・評価を依頼する。
第 2 回 AES 候補コンファレンス	1999/3/22 ～ 3/23	技術的評価第 1 ラウンドにおけるアルゴリズムの分析結果について議論するとともに、アルゴリズムの絞り込みを行う際の留意点等について議論する。
第 2 ラウンド候補の発表	1999/4 月頃	NIST は、候補のアルゴリズムを５つ程度に絞り込み、発表する。
技術的評価第 2 ラウンド	1999 年中 (6～9 か月間)	NIST は、絞り込んだアルゴリズムについて、暗号研究者から寄せられた分析結果を参考に、より詳細に分析・評価を行う。
第 3 回 AES 候補コンファレンス	1999 年末 ～2000 年初	技術的評価第 2 ラウンドにおける分析結果について議論するとともに、アルゴリズムの一層の絞り込みの方法等について検討する。
アルゴリズムの最終選定	発表時期不明	NIST は、第 3 回コンファレンスの結果を参考にして候補アルゴリズムを 1 つに絞り込み、発表する。

2. 候補アルゴリズムの概要

NIST は、1998 年 8 月に開催された第 1 回 AES 候補コンファレンスにおいて、「全体で 21 のアルゴリズムが提案されたものの、提出されたアルゴリズムのプログラムが正常に稼動するか否か、提出書類に不備がないか否か等について審査した結果、15 のアルゴリズムが事前審査にパスした」旨を発表し、各アルゴリズムの仕様や関連論文等を公表した（表 6 参照）。15 のアルゴリズムのうち、5 件が米国から提案されているほか、カナダ（2 件）、韓国、フランス、日本、コスタリカ、オーストラリア、ドイツ、ベルギー、イギリスといった国々から提案されている。

―― 各アルゴリズムの詳細については、巻末別紙の 5 つの表を参照。各表の内容は以下の通り。本節および以下の表は、第 1 回 AES 候補コンファレンスのプロシーディングス（NIST[1998]）や各提案者が発表した資料[27]に基いて作成されている。

表 7：各アルゴリズムの設計指針
表 8：各アルゴリズムの構造上の特徴
表 9：アルゴリズムの構造に関する概念整理（Feistel 構造、SPN 構造、2-round SPN 構造等ラウンド関数や F 関数の構造について整理）
表 10：各アルゴリズムの安全性に関する分析結果
表 11：各アルゴリズムの処理速度に関する分析結果

[27] 各提案者が発表した資料については、NIST のホームページ（http://www.nist.gov/aes）からダウンロードすることが可能となっている。

表 6　AES の候補として提案されているアルゴリズム

名称	国名・代表提案者（会社名等）
CAST-256 (ｷｬｽﾄ 256)	カナダ・Entrust Technologies 社
CRYPTON (ｸﾘﾌﾟﾄﾝ)	韓国・Future Systems 社
DEAL (ﾃﾞｨｰﾙ)	カナダ・Outerbridge
DFC (ﾃﾞｨ･ｴﾌ･ｼｰ)	フランス・Centre National pour la Recherche Scientifique
E2 (ｲｰﾂｰ)	日本・NTT 社
FROG (ﾌﾛｯｸﾞ)	コスタリカ・TecApro Internacional 社
HPC (ｴｲﾁ･ﾋﾟｰ･ｼｰ)	米国・Schroeppel（アリゾナ大学）
LOKI97 (ﾛｷ 97)	豪・Brown（Australian Defense Force Academy）

名称	国・代表提案者（会社名等）
MAGENTA (ﾏｼﾞｪﾝﾀ)	ドイツ・Deutsche Telekom 社
MARS (ﾏｰｽﾞ)	米国・IBM 社
RC6 (ｱｰﾙ･ｼｰ 6)	米国・RSA Laboratories 社
RIJNDAEL (ﾗｲﾝﾃﾞｰﾙ)	ベルギー・Daemen（Banksys 社）
SAFER+ (ｾｲﾌｧｰ･ﾌﾟﾗｽ)	米国・Cylink 社
SERPENT (ｻｰﾍﾟﾝﾄ)	イギリス、イスラエル、ノルウェー・Anderson（ケンブリッジ大学）
TWOFISH (ﾄｩｰ･ﾌｨｯｼｭ)	米国・Schneier（Counterpane Systems 社）

（1）CAST-256

CAST-256 はカナダの Entrust Technologies 社によって提案されたアルゴリズムであり、ブロック長は 128 bit、鍵長は 128、192、256 bit が利用可能である（Adams[1998]）。

①設計方針

CAST-256 の主な設計方針は、同社によって既に発表されている CAST-128（ブロック長 64 bit、鍵長 128 bit）のアルゴリズムをベースにして、安全性と処理速度を向上させる、というものである。

②アルゴリズムの構造

128 bit のデータブロックは 4 つの 32 bit サブブロックに分割された後、48 段のラウンド関数によって変換される。ラウンド関数は一般形 Feistel 構造と呼ばれる構造を有しており、3 種類の F 関数によって構成されている。各 F 関数は CAST-128 で利用されているものが採用されている。F 関数は、bit シフト、排他的論理和、2^{32} を法とする加算、引算のほか、4 種類の S-box（8 bit 入力、32 bit 出力）によって構成されている。

③安全性と処理速度

安全性に関しては、提案者は、「40 段の最大差分特性確率が 2^{-140} 以下であるほか、48 段の最大線形特性確率が 2^{-122} であることから、差分・線形解読法に対して十分な安全性を有している。また、高階差分攻撃や関連鍵攻撃等他の攻撃法に対しても安全であるとみられる」との分析結果を発表している。

暗号化の処理速度については、NIST 指定の C 言語によるソフトウェア実装において約 14 Mbps、アセンブリ言語での実装において約 47Mbps と試算されている。

（2）CRYPTON

CRYPTON は、韓国の Future Systems 社によって提案されたアルゴリズムであり、ブロック長は 128 bit、鍵長は可変（64～256 bit まで 32 bit ごとに利用可能）である（Lim[1998]）。

①設計方針

CRYPTON の主な設計方針は、(i) 差分解読法や線形解読法等既存の攻撃法に対して十分な安全性を確保する、(ii) 非線形変換を並列処理できる構造を採用し、ハードウェア・ソフトウェア両方において高速処理を可能にする、(iii) 安全性評価を容易にするためにシンプルな構造とする、の 3 点である。

②アルゴリズムの特徴

CRYPTON のデータランダム化部は、SPN 構造を有するラウンド関数 12 段で構成されており、128 bit データブロックは 4 つの 32 bit サブブロックに分割され、各々が並列に処理されるように構成されている。ラウンド関数では、換字変換、bit 単位および byte 単位での転置変換、32 bit 拡大鍵との排他的論理和が採用されている。鍵スケジューリング部では、鍵から 32 bit 拡大鍵が 52 個生成される。

③安全性と処理速度

安全性に関する提案者の評価では、8 段のラウンド関数における最大差分・線形特性確率はそれぞれ 2^{-160}、2^{-128} 程度となることから、差分・線形解読法に対して十分な安全性を確保しているとされている。また、4 段のラウンド関数の出力をブール多項式で表示すると次数が 128 以上となることから、高階差分攻撃に対して十分な安全性を有していると分析されている。

暗号化の処理速度については、Pentium Pro 200MHz、32 MB RAM、Windows95、Microsoft Visual C 5.0 での実装において約 51 Mbps、アセンブリ言語での実装において約 62 Mbps との測定結果が発表されている。

（3）DEAL

DEAL（Data Encryption Algorithm with Larger Blocks）は、カナダの Outerbridge とノルウェー・ベルゲン大学の Knudsen によって提案されたアルゴリズムであり、ブロック長 128 bit、鍵長 128、192、256 bit が利用可能である（Knudsen[1998]）。

①設計方針

DEAL の主な設計方針は、(i) DES のアルゴリズムを F 関数に利用することで、これま

でに提案されている解読法に対して安全性を確保する、(ii) Triple DES と同程度の処理速度で実装可能にする、(iii) 既に DES が実装されている環境において、DES からの移行を容易にする、である。

②アルゴリズムの構造

データランダム化部では、128 bit データブロックは 2 つの 64 bit サブブロックに分割された後、Feistel 構造を有しているラウンド関数によって変換される。段数は 6 段以上が推奨されている。ラウンド関数内の F 関数には DES のアルゴリズムが利用されている点が特徴である。鍵スケジューリング部においても DES のアルゴリズムが利用されており、鍵は 64 bit 単位に分割された後、段数に等しい数の 64 bit 拡大鍵が生成される。

③安全性と処理速度

提案者によって差分解読法に対する安全性について分析がなされており、「ラウンド関数 6 段で暗号化されたデータを解読するためには、2^{70} 個以上の選択平文と 2^{121} 回の DES の暗号化処理が必要であり、差分解読法に対して安全性を確保している」と発表している。

暗号化・復号の処理速度については、C 言語によるソフトウェア実装において、暗号化・復号ともに DES のスピードの約 6 分の 1 と試算されている。

(4) DFC

DFC (Decorrelated Fast Cipher) は、フランス・CNRS によって提案されたアルゴリズムであり、ブロック長 128 bit、鍵長可変 (上限は 256 bit) である (Gilbert et. al.[1998], Vaudenay[1998])。

①設計方針

DFC の主な設計方針は、(i) decorrelation theory[28]を利用することによって、Triple DES よりも高い安全性を確保する、(ii) DES よりも高速での実装を可能とする、(iii) 様々なプラットフォームにおいて実装可能にする、である。

②アルゴリズムの特徴

DFC のデータランダム化部は Feistel 構造を有するラウンド関数 8 段で構成されており、データブロックは 2 つの 64 bit サブブロックに分割される。F 関数には、2^{64} を法とする加算や乗算のほか、非線形変換 (6 bit 入力、32 bit 出力) が利用されており、128 bit 拡大鍵がパラメーターとして入力される。鍵スケジューリング部では、128 bit 拡大鍵が 8 個生

[28] decorrelation theory は、差分解読法や線形解読法に対する安全性の証明が可能な暗号アルゴリズムを構築するための新しい技術であり、Vaudenay らによって発表されている (Vaudenay[1998])。

成される。

③安全性と処理速度

DFC の安全性に関しては、decorrelation theory によって差分・線形解読法に必要な計算量が見積もられており、差分解読法に対しては 2^{110} 個以上の選択平文が必要であるほか、線形解読法に対しては 2^{92} 個以上の既知平文が必要になるとの分析結果が示されている。

暗号化の処理速度については、Pentium Pro 200 MHz、C 言語による実装において、約 34 Mbps との測定結果が発表されている。

（5）E2

E2（Efficient Encryption Algorithm）は、日本の NTT 社によって提案されたアルゴリズムであり、ブロック長 128 bit、鍵長 128、192、256 bit が利用可能である（神田他[1998]、盛合他[1998]、NTT[1998]）。

①設計方針

E2 の主な設計方針は、(i) 既存の攻撃法に対する安全性を客観的尺度によって保証する、（ii）簡素なラウンド関数を利用することで、高速処理を実現する、（iii）あらゆるプラットフォーム上で柔軟な実装を可能にする、である。

②アルゴリズムの構造

E2 のデータランダム化部では、拡大鍵による排他的論理和と乗算、byte 単位の転置変換によって構成される初期変換部、Feistel 構造を有するラウンド関数 12 段、初期変換部の逆変換となる最終変換部によって構成される。初期変換部による変換後、128 bit データブロックは 2 つのサブブロックに分割され、ラウンド関数に入力される。F 関数は、「S-box による換字変換→線形変換（8 bit 単位の排他的論理和を利用）→S-box による換字変換」という構造を有している。S-box（8 bit 入出力）は差分・線形解読法等いくつかの既存の攻撃法に対して安全性が確保されるように設計されている。鍵スケジューリング部では、鍵から 128 bit 拡大鍵が 16 個生成される。

③安全性と処理速度

E2 の F 関数に含まれる S-box は、最大差分特性確率、最大線形特性確率等 10 項目の安全性評価基準を満足するように選択されており、ラウンド関数 9 段で既存の主な攻撃法に対して十分な安全性が確保されていると分析されている。例えば、9 段の最大差分特性確率および最大線形特性確率がそれぞれ 2^{-140}、2^{-131} 程度であることが示されている。また、鍵スケジューリング部では、データランダム化部の F 関数の一部や 8 bit 単位の転置変換が利用されており、鍵スケジューリング部への攻撃が容易に適用できない構造となってい

ると分析されている。

暗号化・復号の処理速度については、NIST 指定の C 言語によるソフトウェア実装において、暗号化・復号ともに約 36 Mbps、アセンブリ言語での実装においては、暗号化・復号ともに約 61 Mbps との測定結果が発表されている。

（6）FROG

FROG は、コスタリカの TecApro Internacional 社によって提案されたアルゴリズムであり、ブロック長（64～1,024 bit）、鍵長（40～1,000 bit）とも可変である（Georgoudis, Leroux and Chaves[1998]）。

①設計方針

FROG の主な設計方針は、（i）簡素な変換を利用することで、様々なアプリケーションにおいて高速処理を実現する、（ii）毎回異なる変換手段を利用することによって安全性を確保する、（iii）様々なブロック長と鍵長の組み合わせを利用可能にする、（iv）様々なプラットフォームでの実装を可能とする、である。

②アルゴリズムの構造

FROG では、まず鍵スケジューリング部において、鍵から 3 種類の変換表（大きさはブロック長に依存）が 8 組生成される。データランダム化部はラウンド関数 8 段によって構成されており、3 種類の変換表が 1 組ずつ各ラウンド関数の変換に利用される。ラウンド関数におけるデータブロックの変換は byte 単位での排他的論理和と換字変換のみとなっており、非常に単純な構造となっているほか、各ラウンド関数における変換方法がすべて異なっている点が特徴である。

③安全性と処理速度

安全性に関して、提案者は、「全数探索法よりも効率的な解読法はないとみられる」と評価している。

暗号化の処理速度については、Pentium 200 MHz、64 MB RAM、Windows95、C 言語での実装において、約 10 Mbps との測定結果が発表されている。

（7）HPC

HPC（Hasty Pudding Cipher）は、米国・アリゾナ大学の Schroeppel によって提案されたアルゴリズムであり、ブロック長・鍵長ともに可変である（Schroeppel[1998a][1998b]）。

①設計方針

HPC の主な設計方針は、（i）既存の解読法に対して十分な安全性を確保する、（ii）64 bit

の汎用 CPU を用いたソフトウェアでの実装で最高速度での暗号化処理が可能となる構造にする、(iii) 任意のブロック長や鍵長で利用可能とし、様々なアプリケーションに柔軟に対応できるようにする、である。

②アルゴリズムの構造

HPC の特徴は、暗号化・復号の際に、鍵のほかに SPICE と呼ばれる 256 bit のデータ（毎回変更される）を利用する点である。SPICE のデータが変更されると暗号文も変化することから、SPICE も鍵の一部といえる。データブロックは 64 bit 単位のサブブロックに分割され、段数 8 のラウンド関数では、拡大鍵や SPICE を変数とする排他的論理和、2^{64} を法とする加算や引算、bit シフトによって変換される。鍵スケジューリング部では、鍵から 256 個の 64 bit 拡大鍵を含む表が生成され、ラウンド関数に入力される拡大鍵は段数等に依存して決定される。

③安全性と処理速度

安全性について、提案者は、「差分・線形解読法などの既存の解読法は適用困難である」との分析結果を発表している。

暗号化・復号の処理速度については、Pentium 200MHz、C 言語での実装において、暗号化・復号ともに約 7.3 Mbps と試算されている。

（8）LOKI97

LOKI97 は、オーストラリア・Australian Defense Force Academy の Brown、ウォロンゴング大学の Pieprzyk と Seberry によって提案されたアルゴリズムであり、ブロック長 128 bit、鍵長 128、192、256 bit が利用可能である（Brown and Pieprzyk[1998]）。

①設計方針

LOKI97 の主な設計方針は、(i) LOKI89 や LOKI91 のアルゴリズムをベースにする、(ii) 差分解読法、線形解読法、関連鍵攻撃等既存の攻撃法に対して十分な安全性を確保する、である。

②アルゴリズムの構造

LOKI97 のデータランダム化部は 16 段のラウンド関数によって構成されており、各ラウンド関数には Feistel 構造が採用されている。F 関数には 2-round SPN 構造が利用されており、拡大鍵依存型転置変換や 2 種類の S-box によって構成される換字変換等が利用されている。各ラウンド関数の変換には 3 個の 64 bit 拡大鍵が必要となるため、鍵スケジューリング部では、データランダム化部の F 関数を利用したアルゴリズムによって 48 個の拡大鍵が生成される。

③安全性と処理速度

LOKI97 の安全性について、提案者は「差分特性確率の算出は困難であるものの、ラウンド関数 14 段の最大線形特性確率は 2^{-141} 程度と見込まれることから、線形攻撃法に対して十分な安全性を有している」との分析結果を発表している。

暗号化・復号の処理速度については、PentiumⅡ 233 MHz、64 MB RAM、Windows95、Microsoft Visual C++による実装において、暗号化・復号ともに約 6 Mbps との測定結果が発表されている。

（9）MAGENTA

MAGENTA（Multifunctional Algorithm for General-Purpose Encryption and Network Telecommunication）は、ドイツテレコム社によって提案されたアルゴリズムであり、ブロック長 128 bit、鍵長 128、192、256 bit が利用可能である（Jacobson and Huber[1998]）。

①設計方針

MAGENTA の主な設計方針は、（i）アバランシュ性（入力データの 1 bit が変化した場合、その影響が出力データのすべての bit に及ぶ性質）等安全性の観点から望ましい性質を有する関数を利用して、簡素な構造とする、（ii）ハードウェア・ソフトウェアの両方で高速処理を実現する、である。

②アルゴリズムの構造

MAGENTA のデータランダム化部には Feistel 構造が採用されており、128 bit のデータブロックは 2 つの 64 bit サブブロックに分割された後に変換される。ラウンド関数の段数は、鍵長が 128 bit および 192 bit の場合には 6、鍵長が 256 bit の場合には 8 と設定される。F 関数（MAGENTA では E 関数と呼ばれている）には、線形変換と 256 を法とするべき乗剰余演算等の非線形変換が利用されている。ラウンド関数 1 段あたり 64 bit 拡大鍵が 1 個必要となるが、鍵スケジューリング部では、鍵を 64 bit 単位のデータに分割し、それらのデータを拡大鍵としてそのままラウンド関数に入力する仕組みとなっている。

③安全性と処理速度

安全性に関して、提案者は、「ラウンド関数 1 段の差分および線形特性確率はそれぞれ 2^{-40}、2^{-29} 以下になると見込まれることから、差分解読法や線形解読法に対して十分な安全性を有している」と分析している。

暗号化・復号の処理速度については、Pentium Pro 200MHz、C 言語による実装において、暗号化・復号ともに約 1 Mbps との測定結果が発表されている。

（10）MARS

MARS は、米国の IBM 社によって提案されたアルゴリズムであり、ブロック長 128 bit、鍵長可変（128～1,248 bit）である（Burwick et. al.[1998]）。

①設計方針

MARS の主な設計方針は、(i) 32 bit 汎用 CPU を搭載したコンピューターでのソフトウェア実装において最も高速での処理を可能とする、(ii) 複数種類のラウンド関数を利用することによって高い安全性を実現する、(iii) 安全性に関する分析が容易となるような変換手段を利用してアルゴリズムを構築する、である。

②アルゴリズムの構造

MARS のデータランダム化部は、32 段のラウンド関数によって構成されており、128 bit データブロックは 4 つの 32 bit サブブロックに分割されて変換される。Type-3 Feistel 構造と呼ばれるラウンド関数は 4 種類存在し、それぞれ 8 段ずつ組み込まれている。変換手段としては、差分特性や線形特性を考慮して選択された S-box のほか、排他的論理和、加算、乗算、データ依存型および非依存型 bit シフトが利用されている。鍵スケジューリング部には、データランダム化部のラウンド関数の構造が採用されているほか、生成した拡大鍵の bit の値をチェックすることにより、その拡大鍵が弱鍵でないことを検証する仕組みが取り入れられている。40 個の 32 bit 拡大鍵が生成される

③安全性と処理速度

MARS の安全性について、提案者は、「差分解読法を利用するためには 2^{280} 個以上の選択平文が必要であるほか、線形解読法を利用するためには 2^{128} 個以上の既知平文が必要である。また、弱鍵も見つかっていない」との分析結果を発表している。

暗号化・復号速度については、NIST 指定の C 言語によるソフトウェア実装において、暗号化・復号ともに約 28 Mbps、JAVA による実装において、暗号化が約 15 Mbps、復号が約 17Mbps との測定結果が発表されている。

（11）RC6

RC6 は、米国の RSA Laboratories 社によって開発されたアルゴリズムであり、ブロック長、鍵長（上限 2,040 bit）ともに可変である（Rivest[1998], RSA[1998]）。

①設計方針

RC６の主な設計方針は、(i) 32 bit データの乗算等 32 bit CPU によるソフトウェア実装において高速処理が可能となる演算を組み合わせる、(ii) 既存の攻撃法に対して安全性を確保する、(iii) 安全性に関する分析が容易になるようにアルゴリズムの構造をシンプル

にする、である。

②アルゴリズムの構造

RC6 のデータランダム化部では、データブロックは長さの等しい 4 つのサブブロックに分割された後、ラウンド関数に入力される。ラウンド関数の段数は可変であるが、ブロック長 128 bit の場合には 20 段が推奨されている。ラウンド関数は、データ依存型 bit シフトのほか、2^w（w はサブブロックの長さ）を法とする加算、引算、乗算といった算術演算によって構成されている。鍵スケジューリング部では、ラウンド関数を r 段とすると、ブロック長と同じ長さの拡大鍵が（2r+4）個生成される。

③安全性と処理速度

RC6 の安全性について提案者は、「18 段での最大差分特性確率が 2^{-264} 程度であり差分解読法に対して安全であるほか、線形解読法についても、少なくとも 2^{182} 個の既知平文が必要となるため安全である。高階差分攻撃等その他の解読法も適用困難であるとみられる」との分析結果を発表している。

段数 20 の場合の暗号化・復号速度は、PentiumⅡ 266 MHz、32 MB RAM、Windows95、C 言語での実装において、暗号化が約 42Mbps、復号が 45 Mbps との測定結果が発表されているほか、アセンブリ言語での実装においては、暗号化・復号ともに約 101 Mbps との測定結果が発表されている。

（12）RIJNDAEL

RIJNDAEL は、ベルギー・Banksys 社の Daemen とレーベン・カトリック大学の Rijmen によって提案されたアルゴリズムであり、ブロック長・鍵長ともに 128、192、256 bit が利用可能である（Daemen and Rijmen[1998]）。

①設計方針

RIJNDAEL の主な設計方針は、（i）既存の攻撃法に対して十分な安全性を確保する、（ii）様々なプラットフォームにおいて実装可能とする、（iii）安全性に関する分析が容易になるようにアルゴリズムの構造をシンプルにする、である。

②アルゴリズムの構造

RIJNDAEL のラウンド関数は SPN 構造を有しており、データブロックはラウンド関数内で 8 bit 単位で変換される。ラウンド関数の段数はブロック長と鍵長に依存し、10、12、14 段のいずれかとなる。ラウンド関数は 3 種類の変換部によって構成されており、線形変換層（bit シフト等）、非線形変換層（換字変換）、拡大鍵変換層（拡大鍵との排他的論理和）という順番で変換が行われる。鍵スケジューリング部では、ブロック長と同じ長さの拡大

鍵が（r+1）個（r は段数）生成される。鍵スケジューリング部の変換には、データランダム化部の bit シフトと換字変換が利用される。

③安全性と処理速度

RIJNDAEL の安全性に関しては、「8 段の最大差分・線形特性確率がそれぞれ 2^{-350}、2^{-300} 程度となることから、差分・線形解読法に対して安全である。また、補間攻撃や関連鍵攻撃等の攻撃に対しても安全であるとみられる」との分析結果が提案者によって発表されている。

暗号化・復号速度については、Pentium 200MHz、Linux、C 言語での実装において、暗号化・復号ともに約 27 Mbps、アセンブリ言語での実装においては、暗号化・復号ともに約 80 Mbps との測定結果が発表されている。

（13）SAFER+

SAFER+（Secure And Fast Encryption Routine +）は、米国の Cylink 社によって提案されたアルゴリズムであり、ブロック長は 128 bit、鍵長は 128、192、256 bit が利用可能である（Massey, Khachatrian and Kuregian[1998]）。

①設計方針

SAFER+の主な設計方針は、（i）SAFER-K や SAFER-SK をベースにして、既存の攻撃法に対して安全性を確保する、（ii）シンプルな構造を採用し、様々な実装環境において高速処理を可能にする、である。

②アルゴリズムの構造

SAFER+のラウンド関数は 4 種類の変換層によって構成されており、第一および第三層では 256 を法とする加算と排他的論理和、第二層では指数関数 45^X mod 257 と対数関数 $\log_{45}X$、第四層では法 256 の乗算が利用されている。データブロックは 16 個の 8 bit サブブロックに分割される。段数は鍵長に依存し、鍵長 128、192、256 bit に対してそれぞれ 8、12、16 段となっている。鍵スケジューリング部では、弱鍵の発生を防止するために乱数による演算が利用されており、128 bit の拡大鍵が（2r+1）個生成される（r は段数）。

③安全性と処理速度

安全性については、提案者から「5 段の差分特性確率が高々2^{-128}以下であり、6 段で差分解読法に対して十分な安全性を有しているほか、3 段で線形解読法に対して十分な安全性を有している。また、弱鍵や関連鍵もみつかっていない」との分析結果が発表されている。

暗号化・復号の処理速度については、Pentium 200 MHz、64 MB RAM、Windows95、C 言語での実装において、暗号化・復号ともに約 12 Mbps との測定結果が発表されている。

（14）SERPENT

SERPENT は、イギリス・ケンブリッジ大学の Anderson、イスラエル・テクニオンの Biham、ノルウェー・ベルゲン大学の Knudsen が提案したアルゴリズムであり、ブロック長が 128 bit、鍵長が可変（上限 256 bit）である（Anderson, Biham and Knudsen[1998a] [1998b]）。

①設計方針

SERPENT の主な設計方針は、（i）これまで十分な安全性評価が行われている構造を利用することによって、安全性評価を容易にする、（ii）Triple DES よりも高い安全性を確保する、（iii）bitslice implementation[29]によって高速実装を可能にする、である。

②アルゴリズムの構造

SERPENT は、初期・最終転置と、SPN 構造を有する 32 段のラウンド関数によって構成されている。ラウンド関数では、入力されたデータブロックは 32 個の 4 bit サブブロックに分解され、32 個の換字変換 S-box（4 bit 入出力）によって変換された後、排他的論理和や bit シフトによって構成される線形変換が行われる。なお、bitslice implementation においては、データブロックは 4 つの 32 bit サブブロックに分割された後、32 bit 入出力の S-box や線形変換によって変換される。鍵スケジューリング部では、33 個の 128 bit 拡大鍵が生成される。

③安全性と処理速度

安全性に関する提案者の分析結果では、「24 段の最大差分・線形特性確率がそれぞれ 2^{-232}、2^{-109} 以下になることから、差分・線形解読法に対して安全であるほか、5 段の出力データのブール多項式次数が 243 程度になることから、高階差分攻撃に対しても安全である」と発表されている。

暗号化・復号の処理速度については、NIST 指定の C 言語によるソフトウェア実装において、暗号化・復号ともに約 15 Mbps、JAVA による実装において、暗号化・復号ともに約 583 Kbps と試算されている。

---

[29] bitslice implementation は、通常平文ブロックあるいはサブブロック単位で暗号化を行うアルゴリズムを、1 bit 単位の論理変換を使って実装する方法。例えば、32 bit プロセッサーを利用する場合、通常のソフトウェアによる実装では一度に 1 つのデータブロックしか暗号化できないが、bitslice implementation を利用すると、32 個のデータブロックを並列的に処理することが可能となる（ただし、各データブロックの処理速度は低下することから、全体的に処理速度が向上するか否かはアルゴリズムの構造等に依存する）。

（15）TWOFISH

TWOFISH は米国・Counterpane Systems 社の Schneier、Kelsey、Hall、Ferguson、Hi/fn 社の Whiting、カリフォルニア大学バークレー校の Wagner によって提案されたアルゴリズムであり、ブロック長は 128 bit、鍵長は 128、192、256 bit が利用可能である（Schneier et. al.[1998]）。

①設計方針

TWOFISH の主な設計方針は、（i）これまでに安全性に関する分析が十分行われている変換や構造を利用する、（ii）様々なプラットフォームにおいて高速処理を可能にする、（iii）安全性の分析が容易になるようにラウンド関数の構造をシンプルにする、である。

②アルゴリズムの構造

TWOFISH のデータランダム化部は、「拡大鍵との排他的論理和→ラウンド関数 16 段→拡大鍵との排他的論理和」という構成となっており、128 bit データブロックは 4 つの 32 bit サブブロックに分割された後にラウンド関数に入力される。ラウンド関数には Feistel 構造が採用されており、F 関数は bit シフト、8 つの S-box（8 bit 入出力）による換字変換、線形変換、2^{32} を法とする拡大鍵との加算によって構成されている。拡大鍵は 32 bit であり、鍵スケジューリング部において 40 個生成される。

③安全性と処理速度

安全性に関しては、提案者は、「ラウンド関数 7 段に対して差分解読法を適用するためには少なくとも 2^{131} 個の選択平文が必要であるほか、12 段に対して線形解読法を適用するためには少なくとも 2^{121} 個の既知平文が必要となるため、16 段の TWOFISH は差分・線形解読法に対して十分な安全性を有している。また、高階差分攻撃、補間攻撃等その他の既存の攻撃法に対しても安全であるとみられる」と分析している。

暗号化・復号の処理速度については、NIST 指定の C 言語によるソフトウェア実装において約 40 Mbps、アセンブリ言語による実装において約 90 Mbps との測定結果が発表されている。

3. これまでに発表されているアルゴリズムの分析結果

第 1 回 AES 候補コンファレンスにおいて発表された 15 のアルゴリズムのうち、DEAL、DFC、FROG、LOKI97、MAGENTA、MARS に対して、既に提案者以外の暗号研究者から安全性に関する分析結果が寄せられているほか、新しい共通鍵ブロック暗号の解読法のアイデアが発表されており、候補アルゴリズムに関する分析・評価が進められている。主要な安全性に関する分析結果を紹介すると、以下の通り。

（1）DEAL の安全性に関する分析

マンハイム大学の Lucks は、DEAL の安全性に関する次のような分析結果を発表している（Lucks[1998b]）。

・鍵長 192 bit、ラウンド関数 6 段の DEAL は、2^{33} 個の選択平文を入手できれば、2^{145} 回の暗号化処理によって解読可能である。2^{33} 個の選択平文を入手するための計算量は非現実的とはいえないほか、2^{145} 回の暗号化処理に必要な計算量は、全数探索に必要な計算量 2^{192} に比べて大幅に少ない。

（2）DFC の安全性に関する分析

IBM 社ワトソン研究所の Coppersmith は、DFC の安全性に関する次のような分析結果を発表している[30]。

・第 i 段のラウンド関数の出力データが入力データに依存しなくなるような拡大鍵が、2^{-64} の確率で発生する。このような場合、DFC の段数（8 段）は実質的に 1 段減少することとなり、その安全性も低下すると考えられる。また、ラウンド関数の入力と出力ががが一致するような拡大鍵が 2^{-128} の確率で発生する。

（3）FROG の安全性に関する分析

カリフォルニア大学バークレー校の Wagner、Counterpane Systems 社の Ferguson と Schneier は、FROG の安全性に関する次のような分析結果を発表している（Wagner, Ferguson and Schneier[1998]）。

・ブロック長・鍵長ともに 128 bit の FROG は、ある一定の条件を満足する鍵（約 2^{95} 個存在する、鍵空間 2^{128} の 2^{33} 分の 1）によって暗号化された選択平文・暗号文ペアを 2^{58} 個入手できれば、差分解読法によって全数探索法よりも効率的に解読可能である。また、ある一定の条件を満足する鍵（約 2^{96} 個存在する、鍵空間 2^{128} の 2^{32} 分の 1）によって暗号化された既知平文・暗号文ペアを 2^{56} 個入手できれば、線形解読法によって全数探索法よりも効率的に解読することができる。

（4）LOKI97 の安全性に関する分析

ルーベン・カトリック大学の Rijmen とベルゲン大学の Knudsen は、LOKI97 の安全性に関する次のような分析結果を発表している（Rijmen and Knudsen[1998]）。

[30] 本コメントは、NIST の AES 関連サイト（http://www.nist.gov/aes）の電子掲示板"an electronic forum for the AES"に掲載されている。

・ブロック長・鍵長 128 bit の LOKI97 は、2^{56} 個の選択平文・暗号文ペアを用いた差分解読法によって解読可能であるほか、一部のバイアスを有する拡大鍵（拡大鍵全体の約 25%）によって暗号化された 2^{56} 個の既知平文・暗号文ペアによって解読可能である。

（5）MAGENTA の安全性に関する分析

テクニオンの Biham と Biryukov、Counterpane Systems 社の Ferguson と Schneier、ベルゲン大学の Knudsen、ワイツマン研究所の Shamir は、MAGENTA の安全性に関する次にような分析結果を発表している（Biham et. al.[1998]）。なお、この分析は、第 1 回 AES 候補コンファレンスにおける MAGENTA のプレゼンテーション後、数時間のうちに発表されている。

・MAGENTA は、中間一致攻撃により、2^{64} 個の選択平文を入手できれば 2^{64} 回の暗号化処理によって解読可能であるほか、2^{33} 個の既知平文を入手できれば 2^{97} 回の暗号化処理によって解読可能である。

（6）MARS の安全性に関する分析

University of Jyväskylä の Saarinen は、MARS の安全性に関する次のような分析結果を発表している（Saarinen[1998]）。

・MARS には等価鍵（同一の拡大鍵を生成する鍵のペア）が存在し、比較的容易に発見することができる。鍵長を 160 bit に設定した場合には鍵検索アルゴリズムを 2^{16} 回繰り返すことによって等価鍵を見つけることができるほか、1,248 bit に設定した場合には 2^{32} 回で等価鍵を見つけることができる。これは、MARS において利用可能な鍵長が 128～1,248 bit と範囲が広いため、鍵長が長くなるにつれて鍵空間が拡大し、その結果等価鍵の発生確率が高くなることが原因とみられる。

また、カナダ・Videotron 社の Savard は、MARS の安全性を向上させるための方法についてコメントしている[31]。

・MARS のデータランダム化部は 32 段のラウンド関数を有しているが、そのうち 16 段のラウンド関数は拡大鍵に依存しない構造となっている。これらのラウンド関数についても拡大鍵に依存する構造に変更することで、暗号化・復号の処理速度がやや低下する可能性もあるものの、MARS の安全性はかなり向上すると考えられる。なお、こうした構造を採用するためには、ラウンド関数や鍵スケジューリング部の変更が必要となるが、

[31] 本コメントは、NIST の AES 関連サイト（http://www.nist.gov/aes）の電子掲示板”an electronic forum for the AES”に掲載されている。

追加的に必要となる拡大鍵を他の拡大鍵と独立に生成することができれば、安全性が低下するリスクを最小限に抑えることができると考えられる。

(7) 新しい解読法の発表

テクニオンの Biham、Biryukov、ワイツマン研究所の Shamir は、第 1 回 AES 候補コンファレンス直後に開催された暗号の国際学会 CRYPTO'98 において、共通鍵ブロック暗号の新しい解読法"Impossible Differential Cryptanalysis"を考案した旨を発表し、今後これを利用して各候補アルゴリズムの解読を試みるとの考えを示している。

Impossible Differential Cryptanalysis は差分解読法の一種であり、ある特定の差分を有する平文ペアに対し、絶対生成されることがない差分を有する暗号文ペアを利用して鍵の候補を絞り込む方法である。まず、真の鍵の下で、ある特定の差分ΔP を有する平文ペアに対して、どのような暗号文ペアの差分がどの程度の確率で生成されるかを調べ、生成確率がゼロとなる暗号文ペアの差分ΔC をみつける。次に、ΔP の差分を有する平文ペアを任意の鍵 K で暗号化して、その結果生成される暗号文ペアの差分ΔD とΔC を比較する。ΔD=ΔC が成立する場合、鍵 K は真の鍵ではないと判断することにより、候補となる鍵の絞り込みが可能となる。

Biham らは、この攻撃法を利用して 31 段の Skipjack[32] を解読するために必要な選択平文・暗号文数と計算量を見積もっており、第 1 段から第 31 段の Skipjack に対しては、2^{41} 個の選択平文・暗号文ペアと 2^{78} 回の暗号化処理が必要になるとしている（Biham, Biryukov and Shamir[1998]）。

[32] Skipjack：1993 年に NSA によって開発されたブロック長 64 bit、鍵長 80 bit、段数 32 のブロック暗号方式。Skipjack のアルゴリズムは、発表当初から非公開とされていたが、1998 年 6 月に米国政府によって公開された（アルゴリズムの詳細については、NIST[1998a]を参照）。Skipjack は、1993 年 4 月に米国政府が発表したキーエスクロー政策（暗号の利用者に対して、暗号化鍵に関する情報を 2 つの信頼できる機関に寄託することを義務づけ、法律執行上の必要が生じた場合には、捜査当局がこれらの情報を基に暗号鍵を復元することを可能にする仕組み）において利用される専用装置 Clipper Chip に組み込まれる暗号アルゴリズムに指定されていた。また、Skipjack のアルゴリズムは、ISO/IEC 9979（Information technology - Security techniques － Procedures for the registration of cryptographic algorithms）に基づく ISO 暗号アルゴリズム登録制度に登録されている。

Ⅶ. おわりに

DES は、これまでブロック暗号の事実上の標準として利用されてきたが、鍵長が 56 bit と比較的短いことから、Brute Force Method に対する安全性低下の懸念が多くの暗号学者によって指摘されてきた。今年 7 月に開催された RSA 社の DES 解読コンテストの結果は、こうした指摘の正当性を実証するものである。このため、DES から Triple DES への移行が、より積極化しつつある。

米国政府は、Brute Force Method や Short Cut Method に対して十分な安全性を有する単体のブロック暗号として、AES の標準化を進めている。AES の標準化が完了し、安全性に対する高い信頼を得て AES が一般に利用可能となるまでには、2000 年入り後数年はかかるものとみられており、それまでは Triple DES が DES の後継暗号として利用されることとなろう。

現在米国政府は、AES の候補アルゴリズムの安全性・処理速度等に関する分析・評価を進めているほか、各国の暗号学者・研究者も候補アルゴリズムの安全性に関する分析を積極的に進めている。今後、こうした AES の標準化の進展とともに、共通鍵暗号に関する研究がより一層活発になることが望まれる。共通鍵暗号の研究動向について、引き続き注視していく必要があろう。

以　上

（別紙）表 7　AES の候補アルゴリズムの設計方針

表 7　AES の候補アルゴリズムの設計方針（1/2）

上段：名称 中段：代表提案者 下段：代表者の国名・所属名	各アルゴリズムの論文に明記されている設計方針
①CAST-256 Adams カナダ・Entrust Technologies 社	・CAST-128 を改良： CAST-128 のアルゴリズムをベースに、安全性および処理速度を向上させる。
②CRYPTON Lim 韓国・Future Systems 社	・安全性： 差分攻撃法や線形攻撃法等既存の攻撃法に対して、十分な安全性を確保する。 ・処理速度： 非線形変換を並列処理することが可能な構造を採用し、ハードウェア・ソフトウェア両方において高速処理を可能にする。 ・簡素な構造： 安全性の分析を容易にするために、シンプルな構造にする。
③DEAL Outerbrigde カナダ	・安全性： DES のアルゴリズムを F 関数に利用することにより、既存の解読法に対して安全性を確保する。 ・処理速度： Triple DES と同程度の処理速度で実装可能にする。 ・実装： 既に DES が実装されている環境において、DES からの移行を容易にする。
④DFC Vaudenay フランス・CNRS	・安全性： decorrelation theory を利用することで、Triple DES よりも高い安全性を確保する。 ・処理速度： DES よりも高速で実装可能にする。 ・実装： PC や IC カード等様々なプラットフォームにおいて実装可能にする。
⑤E2 神田雅透 日本・NTT 社	・安全性： 差分解読法・線形解読法のほか既存の攻撃法に対して十分な安全性を有することを客観的尺度によって示す。 ・処理速度： 簡素なラウンド関数を利用することで、高速処理を実現する。 ・実装： あらゆるプラットフォーム上で柔軟な実装が可能となる構造とする。
⑥FROG Georgoudis コスタリカ・TecApro Internacional 社	・安全性： 鍵データによって毎回異なる変換表を作成し、変換処理自体を変更することによって安全性を確保する。 ・処理速度： 簡素な変換処理を採用することによって、高速処理を可能にする。 ・鍵長とブロック長： 鍵長とブロック長の様々な組み合わせによる実装を可能にする。 ・実装： IC カード、ATM、HDTV、B-ISDN 等様々なプラットフォームでの実装を可能にする。
⑦HPC Schroeppel 米国・アリゾナ大学	・安全性：既存の攻撃法に対して十分な安全性を確保する。 ・処理速度：64 bit の汎用 CPU を利用したソフトウェア実装において最高速度での暗号化・復号処理を可能にする。 ・実装面での柔軟性：任意の鍵長やブロック長で利用可能とし、様々なアプリケーションに柔軟に対応できるようにする。 ・
⑧LOKI97 Brown オーストラリア・Australian Defense Force Academy	・LOKI89 と LOKI91 を改良：LOKI89 や LOKI91 のアルゴリズムをベースとする。 ・安全性： 差分解読法、線形解読法、関連鍵攻撃等既存の攻撃法に対して、十分な安全性を確保する。
⑨MAGENTA Huber ドイツ・Deutsche Telekom 社	・アルゴリズムの構造：アバランシュ性等安全性の観点から望ましい性質を有する関数を利用して、簡素な構造とする。 ・実装：ハードウェアとソフトウェアの両方で高速処理を可能にする。
⑩MARS Zunic 米国・IBM 社	・安全性： 複数種類のラウンド関数を利用することで高い安全性を実現する。 ・処理速度：32 bit 汎用 CPU を搭載したコンピューターにおけるソフトウェア実装において最も高速での処理を可能とする。 ・ 分析が容易な構造： 安全性に関する分析が容易となるような変換手段を利用してアルゴリズムを構築する。

表 7　AES の候補アルゴリズムの設計方針（1/2）

上段：名称 中段：代表提案者 下段：代表者の国名・所属名	各アルゴリズムの論文に明記されている設計方針
⑪RC6 Robshaw 米国・RSA Laboratories 社	・**安全性：** 既存の攻撃法に対して安全性を確保する。 ・**処理速度：**32 bit データの乗算等 32 bit CPU によるソフトウェア実装において高速処理が可能な演算を組み合わせる。 ・**簡素な構造：** 安全性に関する分析が容易となるように、アルゴリズムの構造をシンプルにする。
⑫RIJNDAEL Daemen ベルギー・Banksys 社	・**安全性：** 既存の攻撃法に対して十分な安全性を確保できる。 ・**実装：** 様々なプラットホームにおいて実装可能にする。 ・**簡素な構造：** 安全性に関する分析が容易となるように、アルゴリズムの構造をシンプルにする。
⑬SAFER+ Chen 米国・Cylink 社	・**安全性：**SAFER-K および SAFER-SK のアルゴリズムをベースに、既存の攻撃法に対して安全性を確保する。 ・**処理速度：**シンプルなアルゴリズム構造により、様々な実装環境において高速処理を実現する。
⑭SERPENT Anderson イギリス・イスラエル・ノルウェー	・**安全性：** DES の S-box 等これまで研究の蓄積のある変換処理や構造を採用することで安全性評価を容易にし、Triple DES よりも高い安全性を確保する。 ・**処理速度：** bitslice implementation によって高速処理が可能な構造とする。
⑮TWOFISH Schneier 米国・Counterpane Systems 社	・**安全性：** 安全性に対する信頼性を高めるために、これまで安全性に関する研究が十分行われている変換処理や構造を利用する。 ・**処理速度：** 様々なプラットフォームにおいて高速処理が可能となるような構造を選択する。 ・**簡素な構造：** 安全性に関する分析が容易となるように、ラウンド関数の構造をできるだけシンプルにする。

（別紙）表 8　AES の候補アルゴリズムの構造上の特徴

表 8　AES の候補アルゴリズムの構造上の特徴（1/3）

名称	全体構造	鍵長（上段）、 ブロック長（下段）	データランダム化部 （拡大鍵でデータブロックを変換）	鍵スケジューリング部 （拡大鍵を生成）
①CAST-256	・一般形 Feistel 構造を利用 ・データブロックを 4 つの 32 bit サブブロックに分割して変換	・128, 192, 256 bit ・128 bit	3 種類の F 関数を利用 ・段数は 48 ・S-box（8 bit 入力、32 bit 出力）は 4 種類	データ変換部のアルゴリズムを利用
②CRYPTON	・SPN 構造を利用 ・並列処理が可能な構造 データブロックを 4 つの 32 bit サブブロックに分割して変換	・可変（32 bit 毎に 64bit から 256 bit まで利用可能） ・128 bit	ラウンド関数（段数は 12）には、換字変換（S-box）、bit 単位の転置変換、byte 単位の転置変換、拡大鍵との排他的論理和（XOR）を連続的に利用 ・4 つの S-box（32 bit 入出力）を利用	鍵（256 bit 未満の場合には不足 bit を padding）から 52 個の 32 bit の拡大鍵を生成
③DEAL	・Feistel 構造を利用 ・データブロックを 2 つの 64 bit サブブロックに分割	・128, 192, 256 bit ・128 bit	ラウンド関数には、F 関数と XOR を利用 ・F 関数には、DES のアルゴリズムを利用 ・段数は 6 段以上	鍵を 64 bit 単位に分割後、DES 暗号のアルゴリズムを利用し、段数に等しい数の 64 bit の拡大鍵を生成
④DFC	・Feistel 構造を利用 ・並列処理が可能な構造 ・データブロックを 2 つの 64 bit サブブロックに分割して変換	・可変（上限は 256 bit） ・128 bit	ラウンド関数（2 種類）には、2^64 を法とする加算と乗算、XOR、非線形変換を利用 ・段数は 8 段 ・非線形変換は変換表（6 bit 入力、32 bit 出力）を利用	鍵を padding によって 256 bit とし、4 段の Feistel 構造を有する変換部（XOR 等を利用）によって 128 bit 拡大鍵を 8 個生成
⑤E2	・Feistel 構造を利用 ・データブロックを 2 つの 64 bit のサブブロックに分割して変換 ・データランダム化部の最初と最後に、算術演算、byte 単位の転置変換等が行われる	・128, 192, 256 bit ・128 bit	ラウンド関数には、F 関数と XOR を利用。ラウンド関数の処理を簡素にする一方、安全性の観点から十分な段数を確保 ・段数は 12 ・F 関数には、2-round SPN 構造を採用し、16 の S-box を利用（10 項目の安全性評価基準を基に S-box を選択）	拡大鍵生成過程で byte 単位での転置変換を行い、拡大鍵の一部から鍵の発見を困難にしている。128 bit 拡大鍵が 16 個生成される
⑥FROG	・鍵から 3 種類の変換表を 8 組作成。8 組の変換表は互いに異なっており、1 組ずつがラウンド関数 1 段の変換に利用される。 ・各ラウンド関数の変換はすべて異なっている。	・可変（40 bit～1,000 bit） ・可変（64 bit～1,024 bit）	ラウンド関数には3つの変換表を利用。データブロックを byte 単位で、4 段階に分けて変換 ・変換手順は「XOR→換字変換→XOR→XOR」であり、データブロックの byte 数だけ繰り返される ・段数は 8 ・3 つの変換表はそれぞれ 8 組準備される	3 つの変換表が拡大鍵の役割を担っている。変換表の大きさはブロック長に依存。鍵から XOR、乱数による変換等を利用して、変換表を生成
⑦HPC	・暗号鍵のほかに、変換に利用する変数として SPICE と呼ばれる 256 bit データを利用 ・データブロックを 64 bit 単位の word に分割して変換 ・8 段のラウンド関数の最初と最後に拡大鍵との XOR を実行	・可変 ・可変	ラウンド関数には XOR、法 2^64 を法とする加算、引算、bit シフトが多用されており、複雑な関数構造となっている ・段数は 8 段	任意の長さの鍵から、256 個の 64 bit 拡大鍵を生成。乱数による初期値を基に 256 個の 64 bit データを生成した後、各データと 64 bit ごとに分割した鍵との XOR を計算して、拡大鍵生成が完了する

表 8　AES の候補アルゴリズムの構造上の特徴（2/3）

名称	全体構造	鍵長（上段）、 ブロック長（下段）	データランダム化部 （拡大鍵でデータブロックを変換）	鍵スケジューリング部 （拡大鍵を生成）
⑧LOKI97	・Feistel 構造を利用 ・データブロックを 2 つの 64 bit サブブロックに分割して変換 ・各段のラウンド関数の変換を複雑にする一方、段数を少なくして処理速度の低下を抑制	・128, 192, 256 bit ・128 bit	ラウンド関数には、F 関数と 2^64 を法とする加算を利用 ・F 関数（入出力 64 bit）には 2-round SPN 構造を利用。F 関数では、「拡大鍵を変数とする転置変換→拡大変換（64→96 bit）→換字変換（2 種類 8 個の S-box＜13→8 bit および 11→8 bit＞）→転置変換→換字変換（2 種類 8 個の S-box）」という手順で変換 ・S-box は、非線形次数等 4 つの基準を基に生成 ・段数は 16	鍵を 64 bit 単位に分割後、F 関数を用いて 48 個の 64 bit の拡大鍵を生成
⑨MAGENTA	・Feistel 構造を利用 ・128 bit データブロックを 2 つの 64 bit サブブロックに分割して変換	・128, 192, 256 bit ・128 bit	ラウンド関数には、E 関数と XOR を利用 ・E 関数（入出力 64 bit、通常のラウンド関数の F 関数に相当）では、Fast Hadamard Transform (FHT)と呼ばれる線形変換に、法 256 によるべき乗演算（f 関数と呼ばれる）を中心とする非線形変換によって改良を加えた関数（T 関数と呼ばれる）を採用。変換は 8 bit 単位で実行 ・段数は、鍵長 128、192 bit の場合は 6、256 bit の場合は 8	特別な変換はなく、64 bit 単位に分割した鍵をそのままデータランダム化部に入力
⑩MARS	・type-3 Feistel 構造を採用 ・データブロックを 4 つの 32 bit サブブロックに分割して変換 ・データランダム化部の最初に拡大鍵との加算、最後に拡大鍵との引算を実行	・可変（128 ～ 1,248 bit） ・128 bit	4 種類のラウンド関数(各 8 段)を利用。換字表（S-box）、XOR、加算、乗算、データ依存型・非依存型 bit シフトを採用 ・最初と最後の 8 段は S-box（入力 8 bit、出力 32 bit）とデータ非依存型 bit シフトを中心に構成され、中間の 16 段は、S-box（入力 9 bit、出力 32 bit）やデータ依存型 bit シフトを含む E 関数（通常のラウンド関数の F 関数に相当）によって構成されている ・S-box は、SHA-1 のアルゴリズムをもとに生成し、差分特性や線形特性を考慮して選択。 ・段数は 32	段数 7 の Feistel 構造を利用。転置変換や XOR による変換に加え、生成した拡大鍵に弱鍵がないことを検証するアルゴリズムを採用。32 bit 拡大鍵が 40 個生成される
⑪RC6	・高速処理が可能な算術演算を中心とする構造 ・データブロックを 4 つのサブブロックに分割して変換	・可変（2,040 bit 以下） ・可変	ラウンド関数には、データ依存型 bit シフトのほか、2^w（w：サブブロック長）加算、引算、乗算、XOR を利用 ・段数は可変（20 段以上を推奨）	鍵から、2r+4 個のブロック長と同じ長さの拡大鍵を生成（r：段数）
⑫RIJNDAEL	・SPN 構造を採用 ・データは byte 単位で変換 ・データランダム化部の最初には鍵ブロックとの加算が実行され、最終段では、換字変換、bit シフト、XOR が実行される	・128,192,256 bit ・128,192,256 bit	ラウンド関数には、3 種類の変換部が存在し、線形変換層（bit シフト等）、非線形変換層（S-box による換字変換）、拡大鍵変換層（拡大鍵との XOR）を採用 ・S-box には、2^8 を法とするべき乗剰余演算を利用 ・段数はブロック長と鍵長に依存（10,12,14 段のいずれか）	鍵から、bit シフト変換と換字変換によって拡大鍵を生成。拡大鍵の長さはブロック長に等しく、(r+1) 個生成される（r は段数）

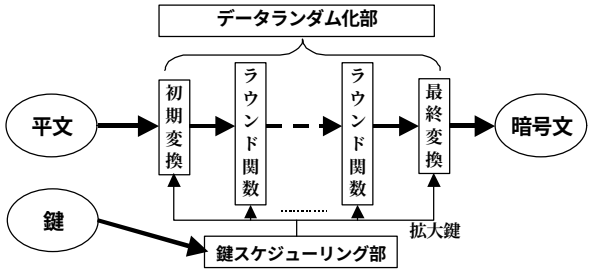
表 8　AES の候補アルゴリズムの構造上の特徴（3/3）

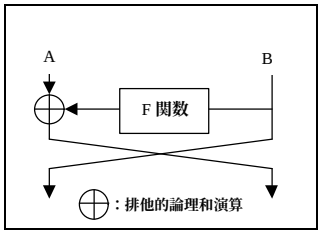
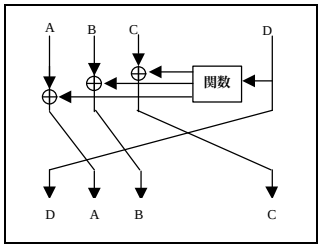
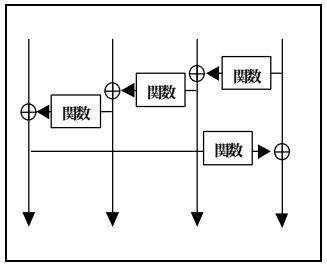
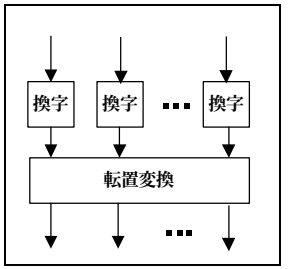
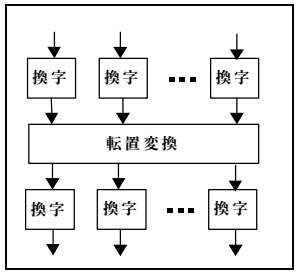
名称	全体構造	鍵長（上段）、 ブロック長（下段）	データランダム化部 （拡大鍵でデータブロックを変換）	鍵スケジューリング部 （拡大鍵を生成）
⑬SAFER+	・4 つの変換層から構成されるラウンド関数を複数繰り返して暗号化・復号する構造を採用 ・サブブロックは 8 bit ・ラウンド関数の最終段が終了後、byte 単位での法 256 の乗算と XOR による変換を配置	・128, 192, 256 bit ・128 bit	ラウンド関数は 4 種類の変換層によって構成。第 1 および第 3 層では 256 を法とする加算と XOR、第 2 層では指数関数 $45^{X} \bmod 257$ と対数関数 $\log_{45}X$、第 4 層では法 256 の乗算を採用 ・段数は鍵長に依存（鍵長と段数の対応：(128, 192, 256 bit) →（8, 12, 16 段)） ・変換処理は byte 単位で実行 ・16×16 の行列を利用している点が従来の SAFER と異なる	鍵から、2r+1 個の 128 bit の拡大鍵を生成（r：段数)。128 bit の乱数（bias と呼ばれている）を利用して変換 ・SAFER-SK で利用されている鍵スケジューリング部の構造を利用
⑭SERPENT	・SPN 構造を採用 ・Bitslice implementation が可能な構造を利用 ・データランダム化部の最初と最後には転置変換を配置	・可変（上限 256 bit） ・128 bit	ラウンド関数は、拡大鍵との XOR、32 個の S-box（入出力 4 bit)、線形変換によって構成 ・S-box は 8 種類存在し、DES の S-box を利用して生成。bitslice mode では、4 種類の 32 bit 入出力の S-box を利用。 ・段数は 32	鍵を padding して 256 bit に拡張した後、bit シフト、S-box を用いて 33 個の 128 bit 拡大鍵を生成
⑮TWOFISH	・Feistel 構造を採用 ・データブロックを 4 つの 32 bit サブブロックに分割して変換 ・データランダム化部の最初と最後に拡大鍵との XOR を配置	・128,192,256 bit ・128 bit	ラウンド関数は XOR と F 関数から構成されており、F 関数は bit シフト、g 関数（8 つの S-box による換字変換と線形変換によって構成)、2^32 を法とする拡大鍵との加算によって構成 ・S-box（8 bit 入出力）は 4 種類存在し、各々2 個ずつが利用。 ・線形変換には、32×32 bit の正方行列による積を利用 ・段数は 16	ラウンド関数で利用した変換（g 関数）を利用して、40 個の 32 bit 拡大鍵を生成

（注）「XOR」は、排他的論理和を表す。

（別紙）表 9 アルゴリズムの構造に関する概念整理

表 9 アルゴリズムの構造に関する概念整理

<table>
<tr>
<td>アルゴリズムの構造</td>
<td><u>データランダム化部</u>：
鍵スケジューリング部から入力された拡大鍵を利用して、平文ブロックの暗号化や暗号文ブロックの復号を行う部分。一般的には、初期変換、最終変換、ラウンド関数といった変換手段によって構成される。</td>
<td><u>ラウンド関数</u>：
データの暗号化や復号の基本変換となる部分。排他的論理和や加算などの複数の変換手段が含まれる。通常、データランダム化部には複数のラウンド関数が含まれる。</td>
<td><u>鍵スケジューリング部</u>：
鍵ブロックから、鍵スケジューリング部に利用するための拡大鍵を生成する部分。</td>
<td colspan="2"> 

＜一般的な共通鍵ブロック暗号の構造＞</td>
</tr>
<tr>
<td>ラウンド関数や F 関数の構造</td>
<td><u>Feistel（type-1 Feistel）構造</u>：
2 つのデータブロック A, B に対して、一方のデータブロック B を F 関数と呼ばれる非線形変換手段によって変換し、他方のデータブロック A を変換後のデータブロック B との排他的論理和によって変換する。変換後、2 つのデータブロックの位置を入れ替える。
例えば、DES や FEAL などのラウンド関数に採用されている。
 
</td>
<td><u>type-3 Feistel 構造</u>：
4 つのデータブロック（A, B, C, D）のうち、1 つのデータブロック D をある関数によって変換し、その変換結果を用いて他の 3 つのデータブロックを排他的論理和によって変換する。変換後は、各データブロックの転置変換を行い、（A, B, C, D）→（D, A, B, C）とする。
例えば、MARS のラウンド関数において利用されている。
 
</td>
<td><u>一般形 Feistel 構造</u>：
4 つのデータブロックを 4 つの関数でそれぞれ変換し、変換後の各データブロックを用いて他のデータブロックを排他的論理和で変換する。
例えば、CAST-256 のラウンド関数に利用されている。
 
</td>
<td><u>SPN 構造</u>：
複数のデータブロックを換字変換(substitution)によって変換した後、転置変換(permutation)によって変換する。例えば、CRYPTON 等のラウンド関数に利用されている。
なお、転置変換の代わりに、より一般的な線形変換が利用される場合も、SPN 構造と呼ばれることがある。このような例として、SERPENT のラウンド関数が挙げられる。
 
</td>
<td><u>2-round SPN 構造</u>：
各データブロックを「換字変換→転置変換→換字変換」の順番で変換する。例えば、LOKI97 の F 関数に利用されている。
なお、転置変換の代わりにより一般的な線形変換が利用されている場合も、2-round SPN 構造と呼ばれることがある。このような例としては、E2 の F 関数が挙げられる。
 
</td>
</tr>
</table>

（別紙）表 10 AES の候補アルゴリズムの安全性に関する分析結果

表 10　AES の候補アルゴリズムの安全性に関する分析結果（1/2）

名称	提案者による分析結果			提案者以外による解読法に関する研究成果
	差分解読法	線形解読法	高階差分攻撃等その他	
①CAST-256	「40 段の最大差分特性確率は 2^(-140)以下であり十分な安全性を有している」と評価 ・1 段の最大差分特性確率は 2^(-14)以下	「48 段の最大線形特性確率は 2^(-122)であり、十分な安全性を有している」との評価	「高階差分攻撃や関連鍵攻撃等既存の攻撃法は適用困難」との評価	―――
②CRYPTON	「8 段の最大差分特性確率は 2^(-160)であり、差分解読法に対して十分な安全性を有している」との評価 ・S-box の最大差分特性確率は、2^(-5)	「8 段の最大線形特性確率は 2^(-128)であり、線形解読法に対して十分な安全性を有している」との評価 ・S-box の最大線形特性確率は、2^(-4)	「高階差分攻撃に対しては、S-box の非線形次数が 5 であり、4 段の非線形次数が 5^4（> 128）となることから、十分な安全性を有している」との評価	―――
③DEAL	「6 段の場合の選択平文攻撃には、2^70 個以上の選択平文と 2^121 回の DES の暗号化処理が必要となることから、適用困難。8 段の場合、全数探索法よりも効率的な解読法は存在しない」との評価	―――	「弱鍵が発見されているものの、極めて少数であり、実際の脅威とはならない」との評価	Lucks (1998)：「6 段で鍵長 192 bit の DEAL は、2^33 個の選択平文と 2^145 の計算量によって解読可能」との評価
④DFC	「2^110 個以上の選択平文が必要であり、適用困難」との評価	「2^92 個以上の既知平文が必要であり、適用困難」との評価	「同一の鍵を 2^48 回以上の暗号化に利用しないことが安全性を確保する上で必要」との評価	Coppersmith：「ラウンド関数の出力データが入力データに依存しない拡大鍵が 2^(-64)の確率で発生する」との指摘
⑤E2	「9 段の最大差分特性確率は 2^(-140.34)以下であり、高い確率の差分特性は存在せず、適用困難」との評価 ・F 関数の最大差分特性確率は 2^(-23.39)以下 ・S-box の最大差分特性確率は 2^(-4.67)	「9 段の最大線形特性確率は 2^(-131.52)以下であり、効率的な線形特性は存在せず、適用困難」との評価 ・F 関数の最大線形特性確率は 2^(-21.92)以下 ・S-box の最大線形特性確率は 2^(-4.38)	「3 段以上で高階差分攻撃は適用困難なほか、補間攻撃も適用困難」との評価 ・S-box のブール多項式の最大次数は 7（F 関数では 40 以上） ・S-box の多項式表示における項数：254 以上 ・S-box の差分線形確率：2^(-2.59)	―――
⑥FROG	「全数探索法よりも効率的な解読法はない」との評価	―――	―――	Wagner 他 (1998)：「弱鍵（2^95 個存在）によって暗号化された 2^58 個の選択平文・暗号文ペアによって解読が可能」との評価
⑦HPC	「通常の差分解読法は適用困難」と評価する一方、「攻撃者にとって都合のよい SPICE データを利用することによって解読を試みる『選択 SPICE 攻撃』に対する安全性については未知数」としている	「線形解読法は適用困難」と評価	「暗号化の際に利用される中間データや拡大鍵の量を基に推定すると、解読に成功するためには少なくとも 2^400 の暗号化処理が必要」と評価。	Coppersmith から「SPICE を安全に配送する方法について何らの提案もなく、SPICE を利用した攻撃法が成立する可能性がある」と指摘

表 10　AES の候補アルゴリズムの安全性に関する分析結果（2/2）

名称	提案者による分析結果			提案者以外による解読法に関する研究成果
	差分解読法	線形解読法	高階差分攻撃等その他	
⑧LOKI97	「ラウンド関数に加算が用いられているため、通常の差分特性確率を数学的に算出することは困難」との評価	「最大線形特性確率は 14 段で 2^(-141)であり、適用困難」との評価 ・1 段の最大線形特性確率は 2^(-11)	―――	Rijmen and Knudsen(1998)：「2^56 個の選択平文あるいは既知平文によって解読可能」
⑨MAGENTA	「E 関数（通常のラウンド関数の F 関数に相当）の差分特性確率は、少なくとも 2^(-40)であり、ラウンド関数の差分特性確率は DES よりも小さくなっているとみられることから適用困難」との評価 ・f 関数の差分特性確率は 2^(-5)以下 ・T 関数の差分特性確率は 2^(-20)以下	「E 関数の線形特性確率は約 2^(-29)以下であることから、適用困難」との評価 ・f 関数の線形特性確率は約 0.6 以下	―――	Biham 他（1998）：「2^64 個の選択平文を入手すれば 2^64 回の暗号化処理で解読できるほか、2^33 の既知平文があれば 2^97 回の暗号化処理で解読可能」
⑩MARS	「2^280 個以上の選択平文が必要であり、適用困難」との評価 ・E 関数（通常のラウンド関数の F 関数に相当）の差分特性確率は概ね 2^(-9) ・16 段の差分特性確率は概ね 2^(-240)	「2^128 個以上の既知平文が必要であり、適用困難」との評価 ・4 段の線形特性確率は概ね 2^(-69)	「弱鍵は見つかっていないほか、光学的暗号攻撃や故障利用暗号攻撃の適用は困難」との評価	Saarinen (1998)：「鍵長 160 bit では 2^16 の計算量で等価鍵がみつかるほか、鍵長 1248 bit では 2^32 の計算量でみつかる」
⑪RC6	「18 段の最大差分特性確率は概ね 2^(-264)であり、適用不可能」との評価	「18 段の RC6 に線形解読法を適用するためには、少なくとも 2^182 個の既知平文が必要であり、適用不可能」との評価	「高階差分攻撃や弱鍵の発見に必要なデータを入手することは困難」と評価	―――
⑫RIJNDAEL	「8 段の最大差分特性確率は 2^(-350)であり、適用不可能」との評価 ・S-box の最大差分特性確率は 2^(-7)	「8 段の最大線形特性確率は 2^(-300)であり、適用不可能」との評価 ・S-box の最大線形特性確率は 2^(-6)	「補間攻撃や関連鍵攻撃に対して安全であるとみられる」との評価	―――
⑬SAFER+	「6 段以上の SAFER+に対して適用困難」との評価 ・5 段の差分特性をすべて調べた結果、差分特性確率は 2^(-128)以下	「3 段以上の SAFER+に対して適用困難」との評価 ・2.5 段において、線形バイアスを有する入出力ペアが見つかっていない	「拡大鍵生成に乱数を利用していることから、弱鍵や関連鍵が発生する可能性は低い」との評価	―――
⑭SERPENT	「6 段の最大差分特性確率が 2^(-58)以下であり、24 段の場合には 2^(-232)以下となることから、適用困難」との評価	「24 段の最大線形特性確率は 2^(-109)以下であることから、適用困難」との評価	「高階差分攻撃は適用困難」との評価 ・S-box のブール多項式次数が 3 であることから r 段後の出力データの次数は 3^r（5 段で 243）	―――
⑮TWOFISH	「7 段の TWOFISH に対して、2^131 個の選択平文が必要であることから、適用困難。全数探索法よりも効率的な解読法は知られていない」との評価	「12 段の TWOFISH に対して、2^121 個の既知平文が必要であることから、適用困難」との評価	「S-box は高次の非線形性を有していること等から、高階差分攻撃、補間攻撃、関連鍵攻撃に対して高い安全性を有している」との評価	―――

（別紙）表 11　AES の候補アルゴリズムの処理速度に関する分析結果

表 11　AES の候補アルゴリズムの処理速度に関する分析結果（1/2）

		C 言語			アセンブリ言語	JAVA		実装環境
		Borland C++ 5.0	MSV C 5.0	その他		JDK	その他	
①CAST-256	暗号化	14 M			47 M			・Borland: NIST 指定 ・アセンブリ：PentiumII 300MHz, Windows NT 4.0
	復号	14 M			47 M			
	鍵セットアップ	<9090>			<4130>			
②CRYPTON	暗号化		51 M		62 M			・MSV 5.0： Pentium Pro 200MHz, 32 MB RAM, Windows95 ・アセンブリ：詳細不明
	復号		51 M		62 M			
	鍵セットアップ		<325>		—			
③DEAL	暗号化	DES 暗号化の 1/6						
	復号	DES 復号の 1/6						
	鍵セットアップ	DES 鍵セットアップの 1/7						
④DFC	暗号化			34 M				・C 言語その他：Pentium Pro 200MHz
	復号			—				
	鍵セットアップ			—				
⑤E2	暗号化	36 M			62 M	11 M*		・Borland：NIST 指定 ・アセンブリおよび JAVA JDK： Pentium Pro 200MHz, 64MB RAM, Windows95
	復号	36 M			62 M	11 M*		
	鍵セットアップ	<2076>			—	—		
⑥FROG	暗号化			10 M				・C 言語その他： Pentium 200MHz, 64MB RAM, Windows95
	復号			13 M				
	鍵セットアップ			<1960000>				
⑦HPC	暗号化			7.3 M				・C 言語その他：Pentium 200MHz
	復号			7.3 M				
	鍵セットアップ			<140000>				
⑧LOKI97	暗号化		6.1 M					・MSV C： PentiumII 233MHz, 64 MB RAM, Windows95
	復号		6.3 M					
	鍵セットアップ		<4870>					

表 11　AES の候補アルゴリズムの処理速度に関する分析結果（2/2）

		C 言語			アセンブリ言語	JAVA		実装環境
		Borland C++ 5.0	MSV C 5.0	その他		JDK	その他	
⑨MAGENTA	暗号化			1.1 M				・C 言語その他：Pentium Pro 200MHz
	復号			1.1 M				
	鍵セットアップ			<7100>				
⑩MARS	暗号化	28 M		85 M		15 M		・Borland：NIST 指定
	復号	28 M		85 M		17 M		・C 言語その他：PowerPC 604e, C Set ++3.1.1
	鍵セットアップ	<9200>		<2050>		<1760>		・JAVA JDK：NIST 指定
⑪RC6（20 段）	暗号化			42 M	101 M	1.6 M		・C 言語その他およびアセンブリ言語：
	復号			45 M	101 M	1.6 M		PentiumⅡ266MHz, 32MB RAM, Windows95
	鍵セットアップ			<4710>	—	<107000>		・JAVA JDK：Pentium Pro 180MHz, 64MB RAM, WindowsNT 4.0
⑫RIJNDAEL	暗号化			27 M	80 M		1.1 M	・C 言語その他：Pentium 200MHz, Linux
	復号			27 M	80 M		1.1 M	・アセンブリ言語：Pentium 200MHz, Linux
	鍵セットアップ			<2100>	—		—	・JAVA その他：Pentium 200MHz, Linux
⑬SAFER+	暗号化			12 M				・C 言語その他：Pentium 200MHz, 64MB RAM, Windows95
	復号			12 M				
	鍵セットアップ			<15342>				
⑭SERPENT	暗号化	15 M				583 K		・Borland：NIST 指定
	復号	15 M				583 K		・JAVA JDK：NIST 指定
	鍵セットアップ	—				—		
⑮TWOFISH	暗号化	40 M	43 M		90 M			・Borland：NIST 指定
	復号	40 M	43 M		90 M			・MSV C およびアセンブリ：
	鍵セットアップ	<10300>	<8000>		<12700>			Pentium Pro 200MHz, 64MB RAM, Windows95

（注）

1. NIST が指定した実装環境は、CPU：Pentium Pro 200MHz、メモリー：64MB RAM、OS：Windows95 を搭載した IBM PC 互換機。プログラム言語は ANSI C（Borland C++ 5.0）と JAVA（JDK 1.1）。
2. 表中の数値の単位は bps。鍵セットアップの数値（< >内の数値）の単位は clock cycle。
3. 暗号化および復号は、鍵長 128 bit、ブロック長 128 bit のケースを想定。
4. 「*」は、JIT compiler を利用した場合の計数。
5. 各計数は、第 1 回 AES 候補コンファレンス（1998 年 8 月 20～22 日）に発表された各資料に掲載されているもの。CAST-256、DEAL、HPC、SERPENT の計数は試算値であり、これら以外のアルゴリズムに関する計数は実測値である。

参考文献

岩下直行・谷田部充子、「金融分野における情報セキュリティ技術の国際標準化動向」、日本銀行金融研究所情報セキュリティ・シンポジウム提出論文、1998 年 11 月.

宇根正志、「AES（Advanced Encryption Standard）について」、日本銀行金融研究所ディスカッションペーパーシリーズ、No. 97-J-16、1997 年.

宇根正志、「最近の AES を巡る動向について」、日本銀行金融研究所ディスカッションペーパーシリーズ、No. 98-J-21、1998 年.

宇根正志・岡本龍明、「公開鍵暗号の理論研究における最近の動向」、日本銀行金融研究所情報セキュリティ・シンポジウム提出論文、1998 年 11 月.

神田雅透・盛合志帆・青木和麻呂・植田広樹・大久保美也子・高嶋洋一・太田和夫・松本勉、「128 ビットブロック暗号 E２の提案」、ISEC 98-12、1998 年 7 月.

下山武司・盛合志帆・金子敏信、「高階差分攻撃の改良と KN 暗号の解読」、ISEC97-29, 1997.

反町亨・松井充、「RC5 の強度評価に関する一考察（その 3)」、SCIS'97-18A, 1997 年.

田中秀磨・久松和之・金子敏信、「FL 関数のない場合の MISTY1 に対する高階差分攻撃」、ISEC98-5,1998 年.

角尾幸保・山田真紀、「選択差分を用いた証明可能安全な暗号に対する攻撃に関する一考察」、SCIS'98-7.2.E, 1998 年.

中山靖司・太田和夫・松本勉、「電子マネーを構成する情報セキュリティ技術と安全性評価」、日本銀行金融研究所情報セキュリティ・シンポジウム提出論文、1998 年 11 月.

浜出猛・横山尚史・島田徹・金子敏信、「DES 暗号に対する partitioning 解析に関する一考察」、SCIS'98-2.2.A、1998 年.

久松和之・金子敏信、「FL 関数のない場合の MISTY1 に対する高階差分攻撃による強度評価の一考察」、SCIS'98-1.2.C, 1998 年.

松井充、「ブロック暗号アルゴリズム MISTY」、ISEC96-11,1996 年.

三上正・金子敏信、「二段消去攻撃における IDEA 暗号の弱鍵」、ISEC97-110、1997 年.

盛合志帆・青木和麻呂・神田雅透・高嶋洋一・太田和夫、「既知のブロック暗号攻撃に対する安全性を考慮した S-box の構成法」、ISEC 98-13、1998 年 7 月.（http://info.isl.ntt.co.jp/~shiho/E2sbox.ps.gz）

盛合志帆・下山武司・金子敏信、「SNAKE 暗号の補間攻撃」、SCIS'98-7.2.C、1998 年.

松本勉・岩下直行、「金融分野における情報セキュリティ技術の現状と課題」、日本銀行金融研究所情報セキュリティ・シンポジウム提出論文、1998 年 11 月.

C. Adams, "The CAST-256 Encryption Algorithm," 1998.（ http://www.entrust.com/resources/pdf/cast-256.pdf）

R. Anderson, E. Biham and L. R. Knudsen, "Serpent: A Proposal for the Advanced Encryption Standard," 1998a.（ http://www.cl.cam.ac.uk/ftp/users/rja14/serpent.pdf）

R. Anderson, E. Biham and L. R. Knudsen, "Serpent: A Flexible Block Cipher With Maximun Assurance," 1998b.（http://www.cl.cam.ac.uk/ftp/users/rja14/ ventura.ps.gz）

American National Standards Institute, "X3.92 — 1981, Data Encryption Algorithm," 1981.

American National Standards Institute, "X9.52 — 1998, Triple Data Encryption Algorithm Mode of Operation," 1998.

E. Biham, "New types of cryptanalytic attacks using related key," Advances in Cryptology — Proceedings of EUROCRYPT '93, Lecture Notes in Computer Science, Vol. 765, pp. 398-409, Springer-Verlag, 1994.

E. Biham, "On Matsui's Linear Cryptanalysis," Advances in Cryptology — Proceedings of EUROCRYPT '94, Lecture Notes in Computer Science, Vol. 950, pp. 341-355, Springer-Verlag, 1995.

E. Biham, "Cryptanalysis of Multiple Modes of Operation," Journal of Cryptology, Vol. 11, No. 1, pp. 45- 58, 1998.

E. Biham, A. Biryukov, N. Ferguson, L. R. Knudsen, B. Schneier, and A. Shamir, "Cryptanalysis of Magenta," August 20, 1998.

E. Biham, A. Biryukov, and A. Shamir, "Cryptanalysis of Skipjack Reduced to 31 Rounds using Impossible Differentials," Technion — Computer Science Department — Technical Report CS0947, October 1998.

E. Biham and L. R. Knudsen, "Cryptanalysis of the ANSI X9.52 CBCM Mode," Advances in Cryptology — Proceedings of EUROCRYPT '98, Lecture Notes in Computer Science, Vol. 1403, pp. 100-111, Springer-Verlag, 1998.

E. Biham and A. Shamir, "Differential cryptanalysis of DES-like Cryptosystems," Jornal of Cryptology, Vol. 4, No. 1, pp. 3-72, 1991.

E. Biham and A. Shamir, "Differential cryptanalysis of Snefru, Khafre, REDOC-II, LOKI and Lucifer," Advances in Cryptology — Proceedings of CRYPTO '91, Lecture Notes in Computer Science, Vol. 576, pp. 156-171, Springer-Verlag, 1992.

E. Biham and A. Shamir, "Differential cryptanalysis of the Full 16-Round DES," Advances in Cryptology — Proceedings of CRYPTO '92, Lecture Notes in Computer Science, Vol. 740, pp. 487-496, Springer-Verlag, 1993.

A. Biryukov and E. Kushilevitz, "Improved Cryptanalysis of RC5," Advanced in Cryptology — Proceedings of EUROCRYPT '98, Lecture Notes in Computer Science, Vol. 1403, pp. 275-286, Springer-Verlag, 1998.

M. Blaze, W. Diffie, R. L. Rivest, B. Schneier, T. Shimomura, E. Thompson, and M. Wiener, "Minimum Key Lengths For Symmetric Ciphers To Provide Adequate Commercial Security," A Report By An Ad Hoc Group Of Cryptographers And Computer Scientists, January 1996. (http://www.bsa.org/policy/encryption/cryptographers_c.html)

J. Borst, L. R. Knudsen, and V. Rijmen, "Two Attacks on Reduced IDEA," Advances in Cryptology — Proceedings of EUROCRYPT '97, Lecture Notes in Computer Science, Vol. 1233, pp. 1-13, Springer-Verlag, 1997.

L. Brown, M. Kwan and J. Pieprzyk, "Improving resistance to differential cryptanalysis and the redesign of LOKI," Advances in Cryptology — Proceedings of ASIACRYPT '91, Lecture Notes in Computer Science, Vol. 739, pp. 36-50, Springer-Verlag, 1993.

L. Brown and J. Pieprzyk, "Introducing the new LOKI97 Block Cipher," June 12, 1998. (http://www.adfz.oz.au/~lpb/research/loki97/loki97spec.ps.gz)

L. Brown, J. Pieprzyk, and J. Seberry, "LOKI — A cryptographic primitive for authentication and secrecy applications," Advances in Cryptology — Proceedings of AUSCRYPT '90, Lecture Notes in Computer Science, Vol. 453, pp. 229-236, Springer-Verlag, 1990.

C. Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, M. Matyas Jr., L. O'Connor, M. Peyravian, D. Safford, and N. Zunic, "MARS – a candidate cipher for AES," July 10, 1998. (http://www.research.ibm.com/security/mars.html)

D. Coppersmith, C. Holloway, S. M. Matyas, and N. Zunic, "The Data Encryption Standard," Information Security Technical Report, Vol. 2, No.2, pp. 22-24, ZERGO, 1997.

J. Daemen and V. Rijmen, "AES Proposal: Rijndael," June 11, 1998. (http://www.esat.kuleuven.ac.be/rijmen/rijndael/Rijndaeldoc.pdf)

W. Diffie and M. E. Hellman, "Exhaustive cryptanalysis of the NBS data encryption standard," Computer, Vol. 10, No. 6, pp. 74-84, 1977.

Electronic Frontier Foundation, *Cracking DES*, O'Reilly & Associates, 1998.

D. Georgoudis, D. Leroux, and B. S. Chaves, "The 'FROG' Encryption Algorithm," June 15, 1998. (http://www.tecapro.com/aesfrog.htm)

H. Gilbert, M. Girault, P. Hoogvorst, F. Noilhan, T. Pornin, G. Poupard, J. Stern, and S. Vaudenay, "Decorrelated Fast Cipher: an AES Candidate," July 12, 1998. (ftp://ftp.ens.fr/pub/dmi/users/vaudenay/GG+98b.ps)

C. Harpes, G. G. Cramer, and J. L. Massey, "A generalization of linear cryptanalysis and the applicability of Matsui's piling-up lemma," Advances in Cryptology —

Proceedings of EUROCRYPT '95, Lecture Notes in Computer Science, Vol. 921, pp. 24-38, Springer-Verlag, 1995.
C. Harpes and J. L. Massey, "Partitioning Cryptanalysis," Fast Software Encryption '97, Lecture Notes in Computer Science, Vol. 1267, pp. 13-27, Springer-Verlag, 1997.
International Organization for Standardization, "ISO 8731-1 Banking — Approved algorithms for message authentication — Part 1: DEA," 1987.
International Organization for Standardization, "ISO 8732 Banking — Key management (wholesale)," 1988.
International Organization for Standardization, "ISO 9564-2 Banking — Personal Identification Number management and security — Part 2: Approved algorithm(s) for PIN encipherment," 1991.
International Organization for Standardization and International Electrotechnical Commission, "ISO/IEC 9979 Information technology — Security techniques — Procedures for the registration of cryptographic algorithms," 1991.
International Organization for Standardization and International Electrotechnical Commission, "ISO/IEC 10118-2 Information technology — Security techniques — Hash-functions Part 2: Hash functions using an n-bit block cipher algorithm," 1994.
International Organization for Standardization and International Electrotechnical Commission, "ISO/IEC 10116 Information technology — Security techniques — Modes of operation for an n-bit block cipher," 1997.
M. J. Jacobson Jr. and K. Huber, "The MAGENTA Block Cipher Algorithm," June 8, 1998.
T. Jakobsen and L. R. Knudsen, "The Interpolation Attack on Block Cipher," Fast Software Encryption '97, Lecture Notes in Computer Science, Vol. 1267, pp. 28-40, Springer-Verlag, 1997.
J. Kelsey, B. Schneier, and D. Wagner, "Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple DES," Advances in Cryptology — CRYPTO '96, Lecture Notes in Computer Science, Vol. 1109, pp. 237-251, Springer-Verlag, 1996.
L. R. Knudsen, "A Key-schedule Weakness in SAFER K-64," Advances in Cryptology — Proceedings of CRYPTO '95, Lecture Notes in Computer Science, Vol. 963, pp. 274-786, Springer-Verlag, 1996.
L. R. Knudsen, "DEAL – a 128-bit Block Cipher," May 15, 1998. (http://www.ii.uib.no/~larsr/aes.html)
L. R. Knudsen and T. Berson, "Truncated differential of SAFER," Proceedings of Fast Software Encryption, Third International Workshop, Lecture Notes in Computer Science, Vol. 1039, pp. 15-26, Springer-Verlag, 1996.
K. Kusuda and T. Matsumoto, "Optimization of Time-Memory Trade-Off Cryptanalysis and Its Application to DES, FEAL-32, and Skipjack," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E79-A, No. 1, January 1996.
K. Kusuda and T. Matsumoto, "A Strengh Evaluation of the Data Encryption Standard," Institute for Monetary and Economic Studies, Bank of Japan, DPS No. 97-E-5, 1997.
X. Lai, "Higher Order Derivatives and Differential Cryptanalysis," Communications and Cryptography, pp. 227-233, Kluwer Academic Publishers, 1994.
X. Lai, J. L. Massey, and S. Murphy, "Markov ciphers and differential cryptanalysis," Advances in Cryptology — Proceedings of EUROCRYPT '91, Lecture Notes in Computer Science, Vol. 547, pp. 17-38, Springer-Verlag, 1991.
C. Lee and Y. Cha, "The Block Cipher: SNAKE with Provable Resistance against DC and LC attacks," Proceedings of JW-ISC'97, pp.3-17, 1997.
C. H. Lim, "CRYPTON: A New 128-bit Block Cipher – Specification and Analysis-," 1998. (http://crypt.future.co.kr/~chlim/pub/cryptonv05.ps)
S. Lucks, "Attacking Triple Encryption," Proceedings of Fast Software Encryption '98,

Lecture Notes in Computer Science, Vol. 1372, pp.239-253, 1998a.
S. Lucks, “On the Security of the 128-Bit Block Cipher DEAL,” August 20, 1998b. (http://th.informatik.uni-mannheim.de/m/lucks/papers/deal.ps.gz)
S. M. Matyas, C. H. Meyer, and J. Oseas, “Generating strong one-way function with cryptographic algorithm,” IBM Technical Disclosure Bulletin, Vol. 27, pp. 5658-5659, 1985.
J. L. Massey, “SAFER K-64: A byte-oriented block-ciphering algorithm,” Proceedings of Fast Software Encrypiton, Cambridge Security Workshop, Lecture Notes in Computer Science, pp. 1-17, Springer-Verlag, 1994.
J. L. Massey, “SAFER K-64: One year later,” Proceedings of Fast Software Encrypiton, Cambridge Security Workshop, Lecture Notes in Computer Science, Vol. 1008, pp. 212-241, Springer-Verlag, 1995.
J. L. Massey, G. H.Khachatrian, and M. K. Kuregian, “Nomination of SAFER+ as Candidate Algorithm for the Advanced Encryption Standard (AES),” June 12, 1998. (http://www.cylink.com/internet/objects.nsf/reference/safer/$file/safer. pdf)
M. Matsui, “Linear Cryptanalysis Method for DES Cipher,” Advances in Cryptology — Proceedings of EUROCRYPT ’93, Lecture Notes in Computer Science, Vol. 765, pp. 386-397, Springer-Verlag, 1994.
M. Matsui, “The first experimental cryptanalysis of the Data Encryption Standard,” Advances in Cryptology — Proceedings of CRYPTO ’94, Lecture Notes in Computer Science, Vol. 839, pp. 1-11, Springer-Verlag, 1994.
R. C. Merkle and M. Hellman, “On the Security of Multiple Encryption,” Communications of the ACM, Vol. 24, No. 7, pp. 465-467, 1981.
S. Miyaguchi, A. Shiraishi, and A. Shimizu, “Fast data encipherment algorithm FEAL-8,” Review of Electrical Communication Laboratories, Vol. 36, No. 4, pp.321-327, NTT, 1988.
National Institute of Standards and Technology, “Data Encryption Standard (DES),” Federal Information Processing Standards Publication (FIPS PUB) 46-2, December 13, 1993.
National Institute of Standards and Technology, “Announcing Request for Candidate Algorithm Nominations for the Advanced Encryption Standard (AES),” September 12, 1997. (http://csrc.nist.gov/ encryption/aes/aes_ 9709.htm)
National Institute of Standards and Technology, “SKIPJACK and KEA Algorithm Specifications,” May 29, 1998a. (http://csrc.nist.gov/encryption/skipjack-1.pdf)
National Institute of Standards and Technology, “AES — The First Advanced Encryption Standard Candidate Conference,” The Proceedings of The First Advanced Encryption Standard Candidate Conference, August 20, 1998b.
Nippon Telegraph and Telephone Corporation, “Specification of E2 – a 128-bit Block Cipher,” June 14, 1998. (http://info.isl.ntt.co.jp/e2/E2spec.pdf)
K. Nyberg, “Linear Approximation of Block Ciphers,” Advances in Cryptology — Proceedings of EUROCRYPT ’94, Lecture Notes in Computer Science, Vol. 950, pp. 439-444, Springer-Verlag, 1995.
K. Ohta and K. Aoki, “Linear cryptanalysis of the Fast Data Encipherment Algorithm,” Advances in Cryptology — Proceedings of CRYPTO ’94, Lecture Notes in Computer Science, Vol. 839, pp. 12-16, Springer-Verlag, 1994.
P. C. van Oorschot and M. J. Wiener, “A known plaintext attack on two-key triple encryption,” Advances in Cryptology — Proceedings of EUROCRYPT ’90, Lecture Notes in Computer Science, Vol. 473, pp.318-325, Springer-Verlag, 1990.
V. Rijmen and L. R. Knudsen, “Weaknesses in LOKI97,” June 15, 1998. (ftp://ftp.esat.kuleuven.ac.be/pub/COSIC/rijmen/loki97.ps.gz)
R. L. Rivest, “The RC5 encryption algorithm,” Fast Software Encryption, Second Internationl Workshop, Lecture Notes in Computer Science, Vol. 1008, pp. 86-96, , Springer-Verlag, 1995.

R. L. Rivest, M. J. B. Robshaw, R. Sidney, and Y. L. Yin, “The RC6 Block Cipher,” 1998. (http://www.theory.lcs.mit.edu/~rivest/rc6.ps)

RSA Laboratories, “RC6 Statements,” 1998.

M. J. Saarinen, “Equivalent keys in MARS,” August 18, 1998. (http://www.math.jyu.fi/~mjos/mars-eqk.ps)

B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Fergusen, “Twofish: A 128-Bit Block Cipher,” June 15, 1998. (http://www.counterpane.com/twofish.ps.zip)

R. Schroeppel, “The Hasty Pudding Cipher Specification,” June 1998a. (http://www.cs.arizona.edu/~rcs/hpc/hpc-spec)

R. Schroeppel, “The Hasty Pudding Cipher: Specific NIST Requirements,” June 1998b. (http://www.cs.arizona.edu/~rcs/hpc/hpc-nist-doc)

M. Smid and M. Dworkin, “Special Report on the First AES Conference,” August 1998. (http://csrc.nist.gov/encryption/aes/round1/crypto98.pdf)

T. Tokita, T. Sorimachi, and M. Matsui, “Linear cryptanalysis of LOKI and s^2DES,” Advances in Cryptology — Proceedings of ASIACRYPT '94, Lecture Notes in Computer Science, Vol. 917, pp. 293-303, Springer-Verlag, 1995.

W. Tuchman, “Hellman presents no shortcut method to the DES,” IEEE Spectrum, Vol. 6, No. 7, pp.40-41, Springer-Verlag, July 1979.

S. Vaudenay, “Provable Security for Block Ciphers by Decorrelation,” Lectures Notes in Computer Science, Vol. 1373, pp. 249-275, Springer-Verlag, 1998.

D. Wagner, N. Ferguson, and B. Schneier, “Cryptanalysis of FROG,” August 15, 1998. (http://www.counterpane.com/frog.pdf.zip)

M. Wiener, “Efficient DES Key Search,” A Rump Session Talk at CRYPTO'93, 1993.