IMES DISCUSSION PAPER SERIES

最近のAESを巡る動向について

宇根正志

Discussion Paper No. 98-J-21

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-21
1998 年 9 月

最近の AES を巡る動向について

宇根正志*

要旨

AES（Advanced Encryption Standard）は、現在米国政府が標準化を進めている次世代の米国政府標準暗号の名称である。米国政府は、これまで標準暗号に認定してきた DES（Data Encryption Standard）の安全性低下が深刻化していることから、1997 年 1 月に DES に代わる標準暗号として AES のアルゴリズムを公募によって策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た。公表内容によると、AES は、ブロック長として 128 bit、鍵長として 128、192、256 bit が利用可能な共通鍵ブロック暗号とされている。

AES の候補となるアルゴリズムの募集は 1998 年 6 月に締め切られ、同年 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、候補となる 15 のアルゴリズムの詳細が発表された。現在、NIST や暗号研究者によって、各アルゴリズムの安全性・処理速度等に関する分析・評価が進められており、早ければ 2000 年内にも AES の標準化が完了する見通しとなっている。NIST は、AES を米国政府標準暗号として今後 20〜30 年の間利用する方針を発表しており、標準化完了後、AES は代表的な共通鍵ブロック暗号として様々な分野において利用されるようになる可能性が高い。

本稿は、第 1 回 AES 候補コンファレンスにおいて、NIST や各候補アルゴリズムの提案者が発表した技術資料を基に、各候補アルゴリズムの概要や分析・評価結果について整理したものである。

キーワード：AES、DES、NIST、共通鍵ブロック暗号、米国政府標準暗号

JEL classification：L86、L96、Z00

* 日本銀行金融研究所研究第 2 課（E-mail: masashi.une@boj.or.jp）

本稿作成にあたっては、横浜国立大学の松本勉助教授、NTT 情報通信研究所の岡本龍明特別研究員および太田和夫特別研究員より有益なコメントを頂戴した。

目次

1. はじめに

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[1]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[2]ことから、1997 年 1 月に DES に代わる標準暗号として AES のアルゴリズムを公募によって策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た[3]。公表内容によると、AES は、ブロック長として 128 bit、鍵長として 128、192、256 bit が利用可能な共通鍵ブロック暗号とされている。

NIST は、DES に代わる次の米国政府標準暗号として Triple DES を採用する方針を決定しており、本年中にも Triple DES が FIPS として認定される見通しである。Triple DES は、2 個または 3 個の異なる鍵を用いて DES のアルゴリズムを 3 回繰り返すという暗号方式である。Triple DES は、鍵長を DES に比べて拡張させる効果があり、全数探索法に対する安全性が向上するものの、Triple DES 特有の攻撃法（Merkle-Hellman 選択平文攻撃等）が存在することや、ブロック長が 64 bit と短いことから暗号文一致攻撃[4]に対するリスクが存在すること等から、AES が利用可能になるまでの間のつなぎとなる暗号方式として位置付ける見方が多いようである[5]。

AES の候補となるアルゴリズムの募集は 1998 年 6 月に締め切られ、同年 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、候補となる 15 のアルゴリズムの詳細が発表された。現在、NIST や暗号研究者によって、各アルゴリズムの安全性・処理速度等に関する分析・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される見通しである。NIST は、AES を米国政府標準暗号として今後 20～30

---

[1] NIST（National Institute of Standards and Technology）：米国商務省の下部組織で、科学技術全般に関する標準を策定する役割を担っているほか、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）を認定する権限を有している。

[2] DES は、鍵長が 56 bit と短いことから、候補となる鍵をすべて試してみる「全数探索法」に対する安全性低下が深刻化していた。そうした中、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、DES が 56 時間で解読されている。詳細については、RSA 社のホームページ（http://www.rsa.com/rsalabs/des2）参照。

[3] NIST の発表内容については、参考文献[1][19]を参照。

[4] 暗号文一致攻撃：「n bit のデータ（総数 2^n 個）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に関する情報を得る方法である。これまで提案されてきた多くの共通鍵ブロック暗号はブロック長が 64 bit であり、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ため、今後、よりブロック長の長い暗号方式が必要となるといわれている。

[5] こうした見方については、参考文献[10]を参照。

年の間利用する方針を発表しており、標準化完了後、代表的な共通鍵ブロック暗号として様々な分野において利用されるようになる可能性が高い。

本稿は、第 1 回 AES 候補コンファレンスにおいて、NIST や各候補アルゴリズムの提案者が発表した技術資料を基に、各候補アルゴリズムの概要や分析・評価結果を整理したものである。

2. 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[6]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。

（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。

（i）暗号文のランダム性[7]
（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、復号化等の処理速度
（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[8]。技術的評価第 2 ラウンドでは、NIST

---

[6] 本節における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（参考文献[19][29]）から引用している。

[7] 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。

[8] 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

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 1 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。

表 1 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。

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。

3. 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 2 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（参考文献[20]）や各提案者が発表した資料[9]に基いて作成されている。

表 3：各アルゴリズムの設計指針
表 4：各アルゴリズムの構造上の特徴
表 5：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 6：各アルゴリズムの安全性に関する分析結果
表 7：各アルゴリズムの処理速度に関する分析結果

表 2　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）
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）

[9] 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。

（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（参考文献[4]）。

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[10]が 2^{-140} 以下であるほか、48 段の最大線形特性確率[11]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[12]や関連鍵攻撃[13]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。

---

[10]最大差分特性確率は、差分解読法に対する安全性指標の 1 つである。差分解読法は、ある特定の差分を有する平文のペアに対し、特定の差分を有する暗号文のペアが生じる確率（差分特性確率）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の数を絞り込む解読法である。最大差分特性確率は各差分に対する差分特性確率の最大値であり、この数値が小さいほど差分解読法に対する安全性が高いとされている。

[11]最大線形特性確率は、線形解読法に対する安全性指標の 1 つである。線形解読法は、平文と暗号文の bit 値の間に特定の線形関係が存在する確率（線形特性確率）が高い場合、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解読法である。最大線形特性確率は線形特性確率の最大値であり、この数値が小さいほど線形解読法に対する安全性が高いとされている。

[12]高階差分攻撃は、暗号化関数（F 関数等）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。入力データを変数として暗号化関数の出力データをブール多項式で表した場合、ブール多項式の次数が十分大きいことが、高階差分攻撃に対して安全であることの必要条件とされている。

[13]関連鍵攻撃は、ある特定の関係を有する 2 つの異なる鍵で暗号化を行うことが可能な環境において、その関係やそれらの鍵によって暗号化されたデータを利用して鍵の候補を絞り込む方法である。

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（参考文献[16]）。

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（参考文献[15]）。

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・復号化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・復号化ともに 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 (参考文献[13])。

①設計方針

DFC の主な設計方針は、(i) decorrelation theory[14]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

[14] 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 (参考文献[30])。

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（参考文献[2][3][21]）。

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、暗号化・復号化ともに約 36 Mbps、アセンブリ言語での実装においては、暗号化・復号化ともに約 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（参考文献[12])。

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（参考文献[27][28]）。

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・復号化の際に、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・復号化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・復号化ともに約 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（参考文献[8]）。

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・復号化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・復号化ともに約 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 が利用可能ある（参考文献[14]）。

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・復号化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・復号化ともに約 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（参考文献[9]）。

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・復号化速度については、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・復号化ともに約 28 Mbps、JAVA による実装において、暗号化が約 15 Mbps、復号化が約 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（参考文献[23][24]）。

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・復号化速度は、PentiumⅡ 266 MHz、32 MB RAM、Windows95、C 言語での実装において、暗号化が約 42Mbps、復号化が 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・復号化ともに約 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（参考文献[11]）。

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[15]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・復号化速度については、Pentium 200MHz、Linux、C 言語での実装において、暗号化・復号化ともに約 27 Mbps、アセンブリ言語での実装においては、暗号化・復号化ともに約 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（参考文献[18]）。

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。

[15] 鍵を固定した場合に、暗号文が平文を変数とする 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て（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、ある特定の鍵に対する暗号化関数を構成することができる。こうして導出した暗号化関数を使っ

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、暗号化・復号化ともに約 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（参考文献[5]）。

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[16]によって高速実装を可能にする、である。

て真の鍵の候補を絞っていく攻撃法が補間攻撃である。

[16] 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）。

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、暗号化・復号化ともに約 15 Mbps、JAVA による実装において、暗号化・復号化ともに約 583 Kbps と試算されている。

（15）TWOFISH

TWOFISH は米国・Counterpane Systems 社の Schneier、Kelsey、Hall、Ferguson、Hi/fu 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（参考文献[26]）。

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 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。

4. 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、FROG、HPC、LOKI97、MAGENTA、MARS の 6 つに対して、既に提案者以外の暗号研究者から分析およびコメントが寄せられている。

（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（参考文献[17]）。

・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）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（参考文献[31]）。

・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。

（3）HPC の安全性に対するコメント

IBM 社ワトソン研究所の Coppersmith は、HPC の安全性について以下のようなコメントを発表している[17]。

・HPC では、通常の鍵ブロックのほかに「第 2 の鍵」として SPICE と呼ばれるデータが利用されているが、どのようにして SPICE を安全に通信相手に送付するかについて、関連論文では全く触れられていない。そうした方法が存在しないとすれば、暗号文の受信者が自分で SPICE を入力するほかはない。この場合、攻撃者が自分の都合

[17] 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。

の良いデータを SPICE として入力することによって、鍵ブロックを推測する有効な方法が存在する可能性があり、HPC の弱点になりかねない。

（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（参考文献[22]）。

・ブロック長・鍵長 128 bit の LOKI97 は、2^{56} 個の選択平文を用いた差分解読法、もしくは同数の既知平文を用いた線形解読法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（参考文献[7]）。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、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（参考文献[25]）。

・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。

5. おわりに

AES の標準化は、現在、技術的評価第 1 ラウンドが開始されたところであり、NIST が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、共通鍵ブロック暗号の安全性や処理速度等に関する研究がより一層高度化していくものと考えられる。

以　上

参考文献

[1] 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、No. 97-J-16、1997.

[2] 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、July 1998.

[3] 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、July 1998.（http://info.isl.ntt.co.jp/~shiho/E2sbox.ps.gz）

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

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

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

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

[8] 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）

[9] 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）

[10] 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.

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

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

[13] 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）

[14] M. J. Jacobson Jr. and K. Huber, "The MAGENTA Block Cipher Algorithm," June 8, 1998.

[15] L. R. Knudsen, "DEAL – a 128-bit Block Cipher," May 15, 1998.（http://www.ii.uib.no/~larsr/aes.html）

[16] C. H. Lim, "CRYPTON: A New 128-bit Block Cipher – Specification and Analysis-," 1998.（http://crypt.future.co.kr/~chlim/pub/cryptonv05.ps）

[17] S. Lucks, "On the Security of the 128-Bit Block Cipher DEAL," August 20, 1998.（http://th.informatik.uni-mannheim.de/m/lucks/papers/deal.ps.gz）

[18] J. L. Massey, G. H.Khachatrian, 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）

[19] 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）

[20] 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, 1998.
[21] 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)
[22] V. Rijmen and L. R. Knudsen, "Weaknesses in LOKI97," June 15, 1998. (ftp://ftp.esat.kuleuven.ac.be/pub/COSIC/rijmen/loki97.ps.gz)
[23] 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)
[24] RSA Laboratories, "RC6 Statements," 1998.
[25] M. J. Saarinen, "Equivalent keys in MARS," August 18, 1998. (http://www.math.jyu.fi/~mjos/mars-eqk.ps)
[26] 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)
[27] R. Schroeppel, "The Hasty Pudding Cipher Specification," June 1998. (http://www.cs.arizona.edu/~rcs/hpc/hpc-spec)
[28] R. Schroeppel, "The Hasty Pudding Cipher: Specific NIST Requirements," June 1998. (http://www.cs.arizona.edu/~rcs/hpc/hpc-nist-doc)
[29] M. Smid and M. Dworkin, "Special Report on the First AES Conference," August 1998. (http://csrc.nist.gov/encryption/aes/round1/crypto98.pdf)
[30] S. Vaudenay, "Provable Security for Block Ciphers by Decorrelation," Lectures Notes in Computer Science, Vol. 1373, pp. 249-275, Springer-Verlag, 1998.
[31] D. Wagner, N. Ferguson, and B. Schneier, "Cryptanalysis of FROG," August 15, 1998. (http://www.counterpane.com/frog.pdf.zip)

（別紙）

表 3　AES の候補として提案されている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・復号化処理を可能にする。 ・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。
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。

（別紙）

表 3　AES の候補として提案されているアルゴリズムの設計方針（2/2）

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
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 4　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 ・段数は 12 ・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

（別紙）

表 4　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	・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）。

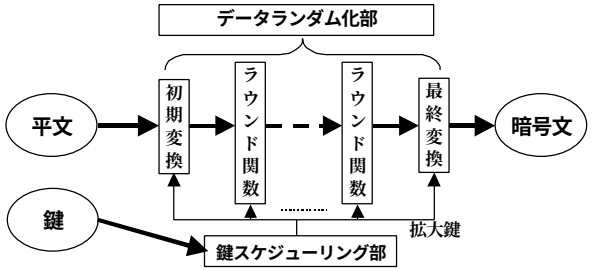
（別紙）

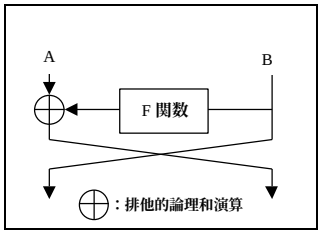
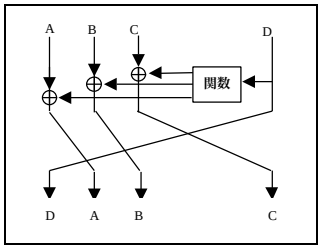
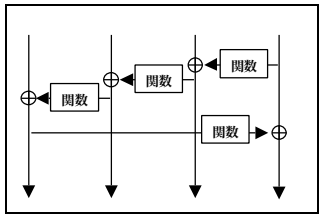
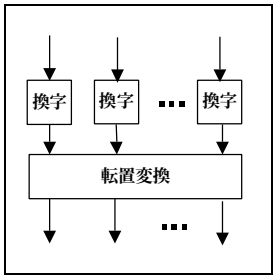
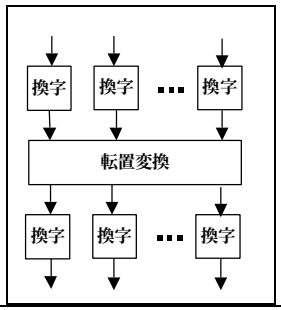
表 4　AES の候補アルゴリズムの構造上の特徴（3/3）

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・復号化する構造を採用 ・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。

表 5 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化を行う部分。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化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>type-3 Feistel <u>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</u> Feistel <u>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 6　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	―――
⑤E2	「9 段の最大差分特性確率は 2^(-140.34)であり、高い確率でみつかる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) ・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)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) ・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 以上（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)：「2^58 個の選択平文により差分解読法で解読できるほか、2^56 個の既知平文により線形解読法で解読できる」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。

（別紙）

表 6　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	―――

表7 AESの候補として提案されている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	14M			47M			Borland C++ :NIST指定
	復号化	14M			47M			**アセンブリ言語：** Pentium II 300MHz, WindowsNT4.0
	鍵セットアップ	<9090>			<4130>			
②CRYPTON	暗号化		51M		62M			MSV C 5.0 :Pentium Pro 200MHz, 32MB RAM, Windows95
	復号化		51M		62M			**アセンブリ言語： 詳細不明**
	鍵セットアップ		<325>		—			
③DEAL	暗号化	DESの1/6						
	復号化	DESの1/6						
	鍵セットアップ	DESの1/7						
④DFC	暗号化			34M				C言語その他 :Pentium Pro 200MHz
	復号化			—				
	鍵セットアップ			—				
⑤E2	暗号化	36M			61M	11M		Borland C++ :NIST指定
	復号化	36M			61M	11M		アセンブリ言語 :Pentium Pro 200MHz, 64MB RAM, Windows95
	鍵セットアップ	<2076>			—	—		JAVA JDK :Pentium Pro 200MHz, 64MB RAM, Windows95
⑥FROG	暗号化			10M				C言語その他 :Pentium 200MHz, 64MB RAM, Windows95
	復号化			13M				
	鍵セットアップ			<1960000>				
⑦HPC	暗号化			7.3M				C言語その他 :Pentium 200MHz
	復号化			7.3M				
	鍵セットアップ			<140000>				
⑧LOKI97	暗号化		6.1M					MSV C :Pentium II 233 MHz, 64MB RAM, Windows95
	復号化		6.3M					
	鍵セットアップ		<4870>					

（別紙）

表7 AESの候補として提案されている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	
⑨MAGENTA	暗号化			1.1M				C言語その他：Pentium Pro 200MHz
	復号化			1.1M				
	鍵セットアップ			<7100>				
⑩MARS	暗号化	28M		85M		15M		Borland C++ ：NIST指定
	復号化	28M		85M		17M		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	暗号化			42M	101M	1.6M		C言語その他：Pentium II 266MHz, 32MB RAM, Windows95
（20段）	復号化			45M	101M	1.6M		JAVA JDK ：Pentium Pro 180MHz, 64MB RAM, WindowsNT 4.0
	鍵セットアップ			<4710>	—	<107000>		アセンブリ言語：Pentium II 266MHz, 32MB RAM, Windows95
⑫RIJNDAEL	暗号化			27M	80M		1.1M	C言語その他：Pentium 200MHz, Linux
	復号化			27M	80M		1.1M	JAVAその他：Pentium 200MHz, Linux
	鍵セットアップ			<2100>	—		—	アセンブリ言語：Pentium 200MHz, Linux
⑬SAFER+	暗号化			12M				C言語その他：Pentium 200MHz, 64MB RAM, Windows95
	復号化			12M				
	鍵セットアップ			<15342>				
⑭SERPENT	暗号化	15M				583K		Borland C++ ：NIST指定
	復号化	15M				583K		JAVA JDK ：NIST指定
	鍵セットアップ	—				—		
⑮TWOFISH	暗号化	40M	43M		90M			Borland C++ ：NIST指定
	復号化	40M	43M		90M			MSV C ：Pentium Pro 200MHz, 64MB RAM, Windows95
	鍵セットアップ	<10300>	<8000>		<12700>			アセンブリ言語：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. 暗号化および復号化は、鍵長 128 bit、ブロック長 128 bit のケースを想定。
4. 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。