H９－WG０５

暗号利用技術ハンドブック

平成 10 年２月

電子商取引実証推進協議会

共通セキュリティ関連技術検討ＷＧ

暗号利用技術ハンドブック

目次の見方

本ハンドブックは、暗号技術を取り入れたシステム構築を行っていく、システム事業者、システムインテグレータ、システム管理者などのシステムエンジニアの方々を主な読者とし、暗号技術を利用したセキュリティを実現するために必要な暗号知識を、分かりやすく解説することを目的としている。

そのため、目次の解説項目毎にシステムエンジニアから見た「重要度」と「難易度」について、A～Cのランク付けを行い、本ハンドブックの読み易さを目指した。

「重要度」　A：重要（必読）、B：重要だが担当者以外は任意、C：任意
「難易度」　A：理論を読み取る必要がある、B：十分理解可能、C：簡単

目　　次

図表番号掲載一覧

1 概　要

1.1 はじめに

電子商取引の分野においては、決済に限らず不正防止が信頼性のかなめとなる。不正を防止するためには各種セキュリティ機能を駆使したインフラ作りとシステムの構築が不可欠な物であり、暗号技術は、このようなセキュリティ機能を実現するための、重要な基礎技術である。

本書は、これからの電子商取引分野の発展とともに暗号技術を取り入れたシステム構築を行っていく、システム事業者、システムインテグレータなどのシステムエンジニアの方々、システムの運用に携われるシステム管理者を主な読者とし、暗号技術を利用したセキュリティを実現するために必要な、暗号知識（暗号方式の種類、安全な鍵長、鍵管理方法など）を、分かりやすく解説することを目的とする。

残念なことに、暗号技術は従来軍事などの限られた部分でしか用いられず、大学などの教育機関でも、特殊な応用研究と位置付けられ、多くのエンジニアの方々は電子回路や情報理論、符号理論、会計学や管理工学は知っていても、暗号についての体系だった知識は皆無である。そもそも、「暗号」という文字から受けるイメージも、どちらかというと「暗い」「防御的」「スパイが使う」など、あまり建設的とはいえない。そのため研究者達が内向的になりがちで一部では「明るい暗号」など、イメージの払拭を図ったが、商用ニーズが少なく研究者達の人気を得るにはいたらなかった。近年、情報技術の発展に伴う電子商取引への期待が高まるにつれ、暗号技術も裏方から一気に表舞台に躍り出ることとなり、情報セキュリティを担う主役となった。しかし、暗号技術を含めて、セキュリティ技術はシステム全体を視野に入れながら適用する必要があり、システムの他の部分にある欠陥を見逃すと、どんなに優れた技術も役に立たなくなるという特殊性がある。たとえば、冷蔵庫は、食品を低温で保存することで安全を保つので温度が下がらなくなれば安全性が落ちたことが簡単に分かるが、金庫の鍵が番号通りに操作しなければ開かないからといっても、その番号が（誰にでも）よく知られていては役に立たない。また、金庫は仮に鍵を忘れたりメカニズムの誤動作で開かなくなった場合でも、その金庫のメーカは開けることが出来るが、暗号は鍵を失うと力ずくの解読作業を行わない限り誰にも（その暗号製品メーカにも！）元の内容を知ることが出来ない。

セキュリティを保持するためには、システムの企画設計段階からどんな場面で、誰が、どのように使うかを想定することが必要であり、更にどんな事をすると破られるのか、また現在の技術では不可能でもいつ頃には破られるかを、システムのライフサイクルとあわせて想定する必要もある。このような配慮は、多くは経験に基づく物であり一朝一夕には得ることは難しい。

本書では、従来であれば企業秘密に属するようなノウハウも部分的に紹介し、種々の暗号の概要説明やそれらの比較、また安全性の基準などを交え、システム構築に必要となる情報（鍵の管理や輸出入の法令）を、できるだけ簡単にまとめた。技術の解説では数式に

よる表現もできるだけ抑え、特に重要と思われる項目については目次にある通り重要度や技術的難易度を考慮して明記し、最低限の内容については理解を頂けるように工夫をした。なお、本書の読者は暗号についての初歩的な知識はお持ちであるという前提のため、そのような点は文中の参考文献を参照頂きたい。また、個別の技術の詳細については、本書の意図から外れるため、必要な方は参考文献を参照して頂きたい。

これらの情報を少しでも有効に活用して頂き、一般の利用者の方が本当に「安心」して利用できる「安全」なシステムを構築して頂ければ幸いである。そして、暗号技術が、広く受け入れられ利用されることで、「安号」「安心できる安全な暗号」となることを期待したい。

1.2 用語の定義

暗号関連の用語はＪＩＳなどでも定義されているが最近の電子商取引やインターネットのブームにより新たな用語や従来とは異なる訳語が多くある。混乱を避けるため、本書ではＪＩＳや従来の暗号関連用語を尊重し、以下の用語を用いることとする。

- 暗号（cryptography）
  意味が簡単には読み取れない符号化のこと。
- 暗号文（ciphertext）
  暗号化されたデータ／文字列。
- 平文（ひらぶん）（plaintext）
  暗号化する前の元のデータ／文字列。
- 暗号化（encipherment/encryption）
  平文を暗号文にする操作。
- 復号（decipherment/decryption）
  暗号文を平文に戻す操作（復号化とは言わない）。
- 鍵（key）
  暗号化／復号を行う際にその変換操作を決定する極めて意味の長さの文字（またはビット）列。
- 暗号化鍵（encipherment/encryption key）
  暗号化を行う際の鍵。共通鍵、公開鍵、秘密鍵が用いられる。
- 復号鍵（decipherment/decryption key）
  復号を行う際の鍵。共通鍵では暗号化鍵と同一、公開鍵で暗号化した場合は秘密鍵が、秘密鍵で暗号化した際は公開鍵が用いられる。
- 鍵長（key length）
  暗号化／復号鍵の長さ。通常ビット単位である。
- 鍵管理（key management）
  暗号化／復号鍵の管理。4章参照。
- 共通鍵暗号方式（common-key cryptographic scheme）
  暗号化と復号に同一の鍵を用いる暗号方式。2.2節参照。
- 共通鍵（common-key）

共通鍵暗号方式で用いられる鍵。2.2 参照。

- 公開鍵暗号方式 (public-key cryptographic scheme)
  暗号化と復号に異なる鍵を用いる暗号方式。2.3 参照。
- 公開鍵 (public key)
  公開鍵暗号方式で用いられる鍵のうち、公開するものを言う。2.3 参照。
- 秘密鍵 (private key)
  公開鍵暗号方式で用いられる鍵のうち、秘密に保持するものを言う。2.3 参照。
  ＪＩＳでは「私用かぎ」が正しい用語であるが、一般的ではない。
  なお、海外でもsecret key と呼ばれる場合がある。この場合共通鍵と紛らわしいので注意が必要。
- 解読 (cryptoanalysis)
  暗号文を、正しい鍵を持たないものが、暗号文の解析や鍵の推測により、平文を求めること。
- 一方向性関数 (one-way function)
  ある入力を変換し出力するのは簡単だが、逆方向の出力から入力を求めることが困難もしくは不可能な関数。ハッシュ関数などもこの一部。
  なお、公開鍵暗号で使用するものは、特殊な価値を持って知っていると逆方向に求めることが可能になるもので、落とし戸付き一方向性関数と呼ばれる。純粋な一方向性関数だとおもって利用していたものが、落とし戸が見つかると非常に危険である。
- ハッシュ関数 (hush function)
  広義には、大きな定義域から小さな領域に値を写像する関数であるが、暗号などで利用されるハッシュ関数は、①一方向性を持ち ②非衝突一致性がある（元の値が異なるのに結果が同一になる確率が低い）ものを言う。2.4 参照。
- 耐タンパー性 (tamper-resistance)
  不正な手段によっては読み書きが不可能なこと。
  ＣＰＵ付きＩＣカード(smart card)などが備えている。
- 否認防止 (non-repudiation)
  データや文書に署名された場合、当事者が後でその署名を行ったことを否定できないこと。
  否認防止というと法令関係用語と紛らわしいため、否認不可、否認拒否とも言う。
- 電子署名 (digital signature)
  公開鍵暗号方式などを利用し、デジタルデータに署名すること。3.4 参照。
- 認証 (certification/authentication)
  相手が真に正しい相手であることを確認するために証明を求めること。3.3 参照。
  識別(identification)と証明(certification)を行う必要がある。
- 暗証番号 (Personal Identification Number ＝PIN)
  個人を識別するために、各個人が覚えて提示する番号。
- パスワード (password)
  暗証番号と同様に、各個人が覚えて提示する文字列。コンピュータのログイン時に

使用される。

- バイオメトリクス（biometrics）
  生化学的特徴。指紋や網膜、虹彩などの個体差がはっきりしているものが用いられる。
- 認証局（certificate authority）
  証明書を発行し、個人や組織、機器やソフトウェアの正当性を証明する機関。
  ＪＩＳでは証明機関。
- 証明書（certificate）
  認証局により発行され、個人や組織、機器やソフトウェアの正当性を証明するデジタルデータ。
  デジタル証明書ともいう。認証書と呼ぶ場合もある
- 失効（revocation）
  証明書や公開鍵が何らかの理由で利用できなくなること。
- 失効リスト（revocation list）
  証明書や公開鍵が何らかの理由で利用できなくなった場合に、その旨を周知するための廃棄鍵のリスト。4.7 参照。
- メッセージ認証子（Message Authentication Code＝ＭＡＣ）
  メッセージデータの要約を暗号やハッシュ関数を用いて作成し、第３者によるものでないという証明をするデータ。

1.3 暗号技術の利用動向と問題点

インターネットが盛んに利用されるにつれて、電子メールの添付を使用し、コンピュータウイルスを送りつけたり、不幸メール、メール爆弾、ウイルス情報を偽ったデマメール、ねずみ講まがいメールなど、ＥＣ関係では他人のクレジットを悪用したり、Ｗｅｂサイトに不正アクセスし他人のパスワードを盗んだり、ホームページデータを不正に入替えたり、悪質なものがでてきている。またこれらは情報を扱う上で、今まではある程度クローズドな情報であったものが、ネットワークの急速な広がりにより個人、組織、国家、社会、国際的にまで広がっており、それら情報が簡単な操作で居ながらにして扱えるたえめ不正に対する意識も薄い。

したがってそれを適切に扱う考え方、すなわち情報倫理の確立も重要であり且つそれら情報を守るための暗号化技術が必要である、今後ＥＣや電子マネーなどがますます使用されるにつれ、アクセス者の認証やデータ暗号化などの技術が必要不可欠なものになると考えられる。例えば、ワンタイムパスワードなどの認証を使用し、不正アクセスに対する防御を行ったり、電子メールの暗号化や電子署名を行うことで、送受信者の特定に応用し、またファイルの暗号化により、データの秘匿や暗号データ作者の認証を行ったりすることが可能となる。

また、暗号技術は使用するシステム全体の仕様や運用、安全性を考慮し最適な技術を導入することが望ましく、暗号アルゴリズムや暗号化鍵の管理方法なども様々なものがあり、その特徴も異なる。暗号アルゴリズムは、暗号化する鍵の長さにより力まかせ攻撃（鍵を１つずつ入力し意味の通る文書やデータに戻るかを確認しながら解読する攻撃方法、総

当り方ともいう）に耐えられる強度がことなる。もし　重 要 なデータを鍵長の短い暗号アルゴリズムで暗号化した場合や、強度がある程度の確 認 をされていない暗号アルゴリズムを使用した場合、簡単に解読されてしまう恐れがある。また暗号鍵の管理も鍵が暗号データの送信者、受信者、その他第３者などの誰が待 て いてよいのか、その管理方法はどのようなものがあるかなど様々な留意点、問題点がある。

2 要素技術と留意点

2.1 本章の目的

情報セキュリティ技術は、非常に多くの分野にわたる基本的な技術の積み重ねによりはじめて実用となる。その中で、暗号技術とその応用が特に重要な位置を占めている。

本書では、これらの技術を大きく３つに分けて取り扱う。まず第一は、全ての基礎となる暗号や乱数などの要素技術を本章で扱う。第二は、暗号の各種の利用法をまとめて応用技術として、３章で述べる。第三に、暗号を扱う上で避けて通れない鍵管理に関する技術を、基本から実装面まで含めて４章としてまとめてある。

なお、５章では技術面ではなく、暗号技術を取り巻く法令をまとめる。これは、他の技術と異なり、標準や安全などの法的制約以外に、暗号技術が政策的な側面を持っているためである。

2.1.1 本章の内容と範囲

本章では、コンピュータ上で主に用いられている各種暗号技術と、暗号技術を安全に利用するための乱数や識別といった技術要素について述べる。これらの技術は必ず必要になる物であり、基礎的な内容については理解が必要であるが、個々のアルゴリズム毎の詳細な仕様や優位点などは、実際に利用を検討する際の参考にして頂けることができるものである。

暗号技術では共通鍵暗号（ブロック暗号とストリーム暗号）と公開鍵暗号、およびハッシュ関数について述べる。これらを理解するには数理の理解が必要となるので、数学的理論なども一部交えて紹介せざるを得ないが、むしろ重要なのはこれらの技術で何ができるのか、個々の技術は何が異なりどのような弱点があるのか、そして想定される攻撃方法はどのようなもので、それに耐えるためには何が必要なのかである。さらに、今後どのような技術進歩や標準化がなされるのか、また今後どうあるべきかを理解していただきたいと考える。

また、乱数は暗号の利用において非常に重要な役割を担っているが、これまでは十分な検討がなされずに、危険な乱数が使われるケースが非常に多かった。そのため、ここでは様々な乱数についての特性や問題点を検討し、暗号技術での利用にあたって注意していただくべき点を述べる。

識別は、本来暗号とはあまり関連が無いが、電子商取引などの分野では利用者本人を認証する以外に、システム自体に改変が加えられたりしたことを察知し、不正利用を防止できることが要求されると思われるため、特にそのような場面においてどのような識別手法があるかを述べる。

最後に、暗号技術の公開をめぐる議論として、暗号アルゴリズムの公開に関する考え方を提示する。

暗号技術は軍事用途など政策的に特別な位置付けがある上、万一破られた場合の影響の大きさから考えて、公開されているアルゴリズムを選ぶべきかそれともまったく誰も知らない方法を選ぶべきか、それぞれの得失を検討する

2.1.2 留意点

本章で述べる技術はあくまで基礎部分だけであり、たとえばあるアルゴリズムを実装するとそのアルゴリズムの持つ安全性がシステムに付加されるということでは決してない。理論的安全性はそれぞれの理論があるが、実装に関する問題点は３章に詳しく述べるので、本章の内容は、基本概念の安全性を検討する際の指標と考えて頂きたい。

暗号はあくまで「なまもの」なので、陳腐化しないように予め考えておくことが重要であるし、解読技術との追いかけっこである点には十分注意を払って頂きたい。また、ここで述べる各暗号技術も、いつ新しい攻撃方法が見かるかもしれず、常に最新の情報を入手しながら、利用する暗号技術の評価を行う必要がある。本書もあくまで今現在の状況を述べるに過ぎないため、可能な物についてはインターネットのＵＲＬを記述してあるので、最新の情報も是非参照して頂きたい。

2.2 共通鍵暗号（対称暗号・慣用暗号）

2.2.1 概要

(1) 共通鍵暗号方式（Common-Key Cryptoqraphic schewe）の原理

共通鍵暗号方式は、古来からの暗号と同じく、暗号化と復号に同一の鍵を用いる方式であり、公開鍵（非対称）暗号方式が開発されるまで、可逆変換可能な暗号方式は、この共通鍵暗号方式しかなかったため、慣用暗号とも呼ばれる。また、暗号化／復号アルゴリズムの内部処理、および鍵の作用から、対称暗号方式と呼ばれることもある。

共通鍵暗号方式では、暗号化と復号に同一の鍵を用いるため、この鍵を秘密にしておかなければ、通信や保管データの安全性が保てない。そのため、秘密鍵（Secret Key）暗号と呼ばれる事もあるが、公開鍵暗号の秘密鍵（Private Key：私用鍵ともいう）と紛らわしいため、その呼称は用いない方がよい。原理的には、平文ブロックの存在空間（ビット列の取りうる値）と暗号文ブロックの存在空間の写像が１対１の可逆変換であり、両方向の変換を行うためのマジックナンバー（鍵）に同じ物を用いるものである。

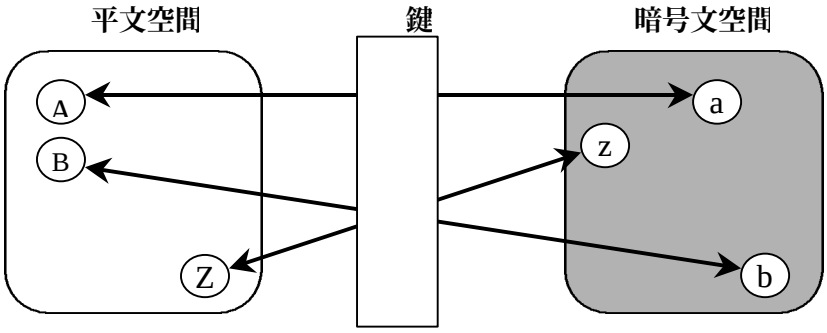


図2.2-1　平文と暗号文の写像

通常は、復号処理は暗号化処理の逆の手順で行えばよい。

また、暗号の分類としては、平文データに対する操作の仕方により

a)　ブロック暗号

b)　ストリーム暗号（逐次暗号）

の2種類が存在する。

(2) 共通鍵暗号の長所と短所

a) 鍵はランダムに生成でき、その長さは比較的短くてよい。

b) 同一アルゴリズムであれば、解読の困難さは鍵の長さに比例する。

c) アルゴリズムにより、線形解読法、差分解読法などの手法を工夫すればすべての鍵を試行しなくとも解読が可能になる。

d) 鍵が共有されるので、暗号を生成したのが誰かは特定できない。

e) 一度鍵が解読されると鍵を共有する多くの利用者が脅威にさらされるため、鍵の生成、配送、保管に十分注意する必要がある。

f) 暗号強度は鍵の長さとランダム性を利用するため計算量が少ない（高速である）。

g) 同様に専用ハードウェアの開発が比較的簡単であり低コストである。

h) すべての鍵を試すことにより正しい暗号化／復号鍵が決定できる（総当り法）、この場合、無条件安全性は保証されず、計算量的安全性の保証のみとなる。

i) 鍵は1つの利用主体もしくは通信経路毎に必要であるため、複数の相手と暗号通信を行う場合は C_2 種類という膨大な数の鍵その管理が必要になる。

(3) 共通鍵暗号アルゴリズムの適用分野

共通鍵暗号アルゴリズムは上記のような特長があり、電子署名など原理的に公開鍵暗号でなければならない場合を除き、ほとんどすべての暗号の応用領域で利用される。

特に、大量のデータを暗号化する際は、必ず用いられている。

不特定多数との暗号通信のように、共通鍵暗号でなく公開鍵暗号を使った方が良い場合も、通常は公開鍵を用いて共通鍵（セッション鍵）の交換を行い、データの暗号化そのものは共通鍵暗号アルゴリズムが用いられる。この場合、必要になる度に鍵を交換しても良いため、鍵の保管における安全性配慮や、鍵の管理についての負担が軽減されるというメリットも生じる。

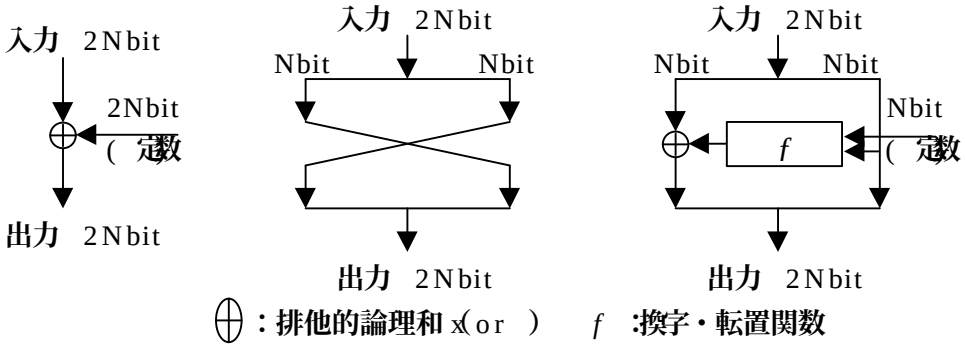
ただし、同じ鍵を用いて同一のデータを暗号化しても同一の結果が出力されるため、攻撃者が正当な通信相手になりすます可能性がある。このような攻撃を防ぐためには、データが毎回異なるよう、暗号を利用するシステム側で配慮されなければならない。

2.2.2 共通鍵ブロック暗号方式

(1) 共通鍵ブロック暗号の定義と原理

共通鍵ブロック暗号とは、平文を一定のブロック長毎に区切って、共通鍵を用いて暗号化処理を行い、復号の際も同じブロック単位は共通鍵を用いて復号を行うものをいう。平文がブロック長に満たない場合はパディングが必要である。パディング方法についてはアルゴリズム毎に定められた方法を用いることが、暗号の強度を保つためにも必要であるが、特に定められていない場合は鍵に応じ決定する必要がある。

共通鍵ブロック暗号では暗号化／復号の際には、解読を困難にするために様々な手法が用いられるが、現在多くのアルゴリズムにおいて、Feistel の開発した暗号で用いられたインボリューションと呼ばれる１：１変換のデータランダム化処理テクニックが用いられている。これらは２回繰り返したり、逆操作を行うことで逆変換が可能であるが、その操作内容を決定するのに暗号化鍵または鍵から導出される複数のサブ鍵）を用いる。



⊕：排他的論理和（xor） f：換字・転置関数

図2.2-2 インボリューションの例

ここで、関数 f では ビットを攪拌するための操作として、ビットの置き換えや並び替え（位置の変更）が行われる。これらを繰り返し行うことで解読しにくい暗号文を生成する。

一般的な Feistel 型暗号の構成を図 2.2-3 に示す。

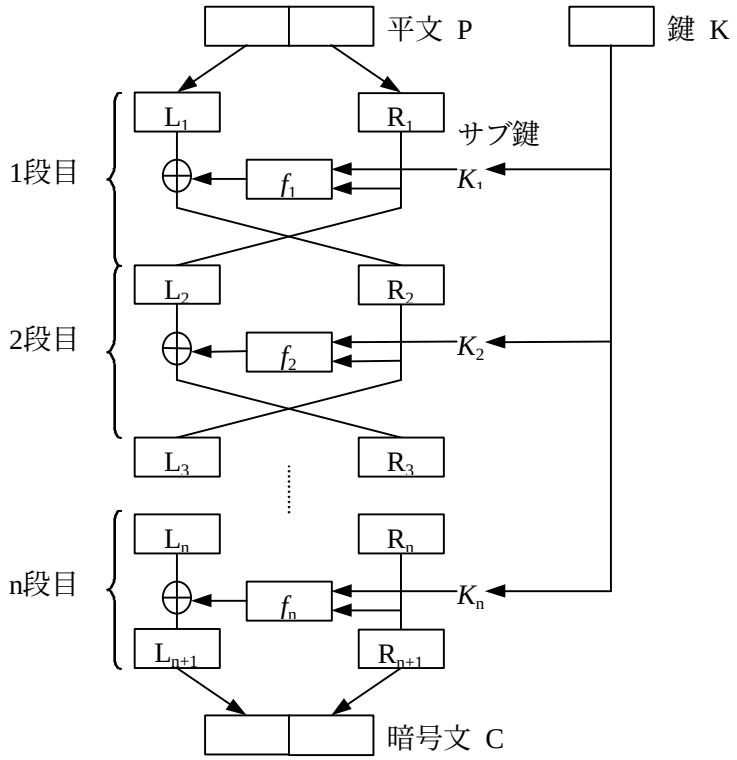


図2.2-3 Feistel 型暗号の 基本構造

ここで暗号アルゴリズムの強度を決定する要因としては 鍵長のほかに、段数、各段の 関数、サブ鍵がある事が分かる。

ＤＥＳなどでは S-box と呼ばれるテーブルを参照し その内容にしたがってビットの並び替えを行う。この方法はハードウェアで実装するには都合が良いが、ソフトウェアで処理するには非常に時間がかかる。そのため 最近の主要な暗号方式ではＣＰＵ処理の軽いビットシフト操作（Rotation ）を用いるものも多い。

一般に、出力暗号文の各ビットは平文と鍵のより多くのビットの影響を受けるよう設計する。そのために、図 2.2-3 のように複数段繰り返して行うことが必要であるが、段数（ラウンド数とも呼ばれる）が増加すると処理時間が増大するため、少ない段数で攪拌効果を高めるために各アルゴリズム毎に工夫されている。処理ブロック長はアルゴリズムにより固定のものと可変のものがあるが 可変の場合も、処理効率からインプリメント毎に固定とすることが多い。また、最近のアルゴリズムでは段数も可変とし、暗号化対象の重要度にあわせて暗号強度を変えられるにできるようにしたものがあるが、用意する S-box やサブ鍵、処理テーブルの数が影響を受けるため、固定としたものも多い。一般に、ブロック長が長くなると攪拌効果が低下するため、段数も増加しなければならないと考えられている。ブロック長決定においては安全性を考慮

し、各アルゴリズムで推奨される段数以上を用いた方がよい。

暗号化鍵（マスター鍵とも呼ぶ）から各段で用いるサブ鍵を生成する処理は、鍵スケジュール処理（Key Scheduling）と呼ばれる。鍵スケジュール処理では、元の暗号化鍵からサブ鍵を求めるのに以下のようないくつかの手法がある。

- マスター鍵をｎ分割しｎ個のサブ鍵とする（サブ鍵の総ビット数はマスター鍵と同一）。
- マスター鍵（またはその分割したブロック）を種としてビット拡大のための一方向性関数（2.4　ハッシュ関数）を用いる（サブ鍵の総ビット数はマスター鍵より多い）。

近年、この鍵スケジュール処理アルゴリズムにより類似した暗号化鍵を使用する場合の安全性についての研究が行われるようになった。その結果鍵の差分攻撃ともいえる方法で鍵を推定できることが明らかになった[1]、[2]。

この中で、安全な鍵スケジュール処理の条件として以下のものが挙げられている。

- サブ鍵を求める処理が線型変換でないこと（逆演算不可能なこと）
- マスター鍵の各ビットがサブ鍵に対して同じ影響力を持つこと
- 生成されるサブ鍵間に依存関係が無いこと

(3) **共通鍵ブロック暗号方式の利点と欠点**

共通鍵自体の得失については2.2.1(2)に述べたとおりで、ここではブロック暗号である点の得失について述べる。

a) **利点**

共通鍵ブロック暗号は、もともとハードウェア化する事を前提に開発された経緯があり、そのため固定のブロック単位で処理を行うようになっている。すなわち、ハードウェア化が容易で、チップも比較的入手しやすい。ソフトウェアで実装する場合も入出力のブロックが一定であり、開発・検証が容易であり、コードも比較的単純である。

また、同じブロック長と鍵長のアルゴリズム同士の場合、簡単に別のアルゴリズムに切り替えることができるメリットがある。さらに、最新の暗号方式では、他の方法に比べて使用するメモリなどの容量が小さくてすむように工夫されているという特長がある。

（ストリーム暗号では鍵の展開や乱数生成のために多くのエリアが必要となるし、公開鍵暗号方式は計算量が多いためコードも大きくなるだけでなく、鍵やデータのバッファサイズが極端に大きくなってしまう。）

b) **欠点**

先に述べたとおり共通鍵ブロック暗号方式では同一の鍵と平文の組み合わせでは必ず同一の結果が得られる。したがって、このような攻撃を想定したシステム構築が必要である。

特に、長いデータ項目を暗号化する場合、冗長データ（Null、スペース、0など）が多いと同じ暗号文ブロックが発生し、平文内容が推定される恐れがあり、解読が容易になる可能性がある。また、ブロック単位で処理を行うため、暗号文中ではブロック境界で偽の情報を埋め込んだり情報の一部を消されて改ざんされ

ても、受信者に判別できないという欠点もある。

ただし、以上の欠点を克服するために、ブロック暗号は2.2.7に述べるような利用モードを設けることができる。とくにこの中のＣＢＣモードは比較的ポピュラーであり、多くの暗号アルゴリズムで使用することができる。また、ＲＣ５など一部の暗号方式ではＣＢＣ同様の連鎖（チェーニング）処理を組み込んだものもある。また、ストリーム暗号と異なり、ブロック長に合わせるためのパディングが必要となるが、その処理が不適切だと、平文の内容に偏りが生ずる恐れがある（上記冗長データと同じ）。

共通鍵ブロック暗号アルゴリズム全体の弱点としては、鍵の総当たり法以外に次のような既知の攻撃法があり、各アルゴリズムに対してはこれらの攻撃法による強度低下の研究や、対抗するためにビット攪拌方式を工夫している。

a) **線形解読法**

1993年に三菱電機の松井によって報告された解読法で、既知平文攻撃（暗号文と平文のペアから推定する方法）の一種。平文と暗号文の幾つかのビットの排他的論理和と鍵の特定ビットとの相関関係より転置・換字のパターンを推測する。

b) **差分解読法**

1990年にワイツマン研究所のＢｉｈａｍとＳｈａｍｉｒが報告した選択平文攻撃（任意の平文に対する暗号文を入手できる場合の方法）の一種。都合の良い複数の入力の差分（異なるビット）を選び出し、それに対する暗号文の差分の統計的性質より転置・換字のパターンを推測する。のちに、ＩＢＭ Ｓの開発時には考慮されていたことが判明した。

上記解読法はいずれも数百億個の平文と暗号文のペアを入手しなければならず、それだけのデータ量を収集し保管・検索することは、現実的な時間内では困難である。上記報告内容も論理的な考察であり、実際にＤＥＳが（公の場で）解読されたことは、つい最近までなかった。

また、通常はこれだけ多くのデータの暗号化を一つの鍵で行うことは、万一鍵が解読された場合に非常に多くのデータが解読されることを意味するため大変危険であることは良く知られており、通常は一定期間（または回数）で鍵の変更が行われる。

一方、総当たり法での解読には膨大な回数の試行が必要であるが、たった一つの暗号文と平文の組だけがあれば可能である（実際には平文は正確に内容が分かっている必要はなく、平文の推測される範囲、たとえば全てアルファベットである、全て数字である、あるいは記号を含む英数字であるといった機械的に判定できるものに限定できればよい）。

この事からも、平文の内容が推定されるようなデータ（連続文字またはビットパターン列）にブロック暗号暗号を施す場合には注意が必要である。後述する実際のＤＥＳの解読でも、暗号文のみが与えられた状態で解読作業を行い、意味のある英文を取り出すことに成功したため、解読できたと判断することができた。逆に言えば、平文の内容がまったくランダムなビット列であった場合は、それが正しく解読できたかどうかを他の手段で確認できなければ、攻撃者は解読に成功したこともわからないといえる。

2.2.3 各種ブロック暗号の比較

ここでは、現在主に用いられている、あるいは今後有力になると見られる、各種ブロック暗号について簡単に内容をまとめる。

表 .2-1　主な共有鍵ブロック暗号方式の比較

ISO9979 登録番号	名　　称	キー長	ブロック長	段数	開発元（パテント）[*1]	発表年	速　　度 [*2]
0004	DES	56bit	64bit	16	IBM（特許放棄 [*3]）	1977	7.7Mbps(P5-90) [*4]
	TripleDES（２キー）	56bit×2 (57bit 相当)	64bit	16 ×3	同上（特許なし [*5]）	1977	DES の 1/3
	TripleDES（３キー）	56bit×3 (112bit 相当)	64bit	16 ×3	同上	1977	DES の 1/3 2.6Mbps(P5-90) [*4]
0002	IDEA	128bit	64bit	8	Ascom Systech	1991	DES の 2 倍 177Mbps(専用 LSI)
0010	FEAL-*N*	64bit	64bit	N	ＮＴＴ	1987	7Mbps(P5-90) [*6] 55Mbps(専用 LSI)
0009	MULTI2	256bit	64bit	可変	日立製作所	1989	
0013	MISTY1 MISTY2	128bit	64bit	8 [*7]	三菱電機	1992	20Mbps(P5-100)
0012	SXAL8	64bit	64bit	8	ローレルインテリジェントシステムズ	1991	
0012	MBAL	64bit	16-1024 Byte	3	ローレルインテリジェントシステムズ	1993	24Mbps(P5-133)
0008	RC2	1-128Byte	64bit	18	ＲＳＡ	1997	8Mbps(P5-90)
	RC5	1-256Byte	32,64,128 ….bit	可変	ＲＳＡ（申請中）	1994	24Mbps(P5-90)
0014	ENCRiP	64bit	64bit	可変	ＮＥＣ	非公開	
	SAFER	64bit	64bit	6~	Jim Massey（フリー）	1994	
0006	Skipjack	80bit	64bit	32	NSA	非公開	
	Blowfish	32-448bit	64bit	可変	Bruce Schneier（フリー）	1994	25Mbps(P5-150)
	Khufu / Khafre	64bit	64bit	８ ×ｎ	Xerox	非公開	
	CAST	40-128bit	64bit	可変	Northern Telecom（フリー）	1997	26Mbps(P5-150) CAST-128
	GOST 28147-89	256bit +512bit	64bit	32	Zabotin,Glazkov, Isaeva	1989	
0017	SPEAM1	128bit	64bit	４ × n ８～	松下電器産業	非公開	

*1 開発元は主に知的所有権保有者を記す。
*2 速度は注記以外開発元による公表速度
*3 ＤＥＳ自体の特許については放棄しているが関連特許があるので注意
*4 ＲＳＡ社製 BSAFE ツールキットによる値
*5 月刊インタフェース 1997 年 9 月号 pp.118
*6 ＦＥＡＬ 32 の値
*7 実際には入れ子構造のため８(32bit)×３(16bit)×３(7,9bit)＝72 段の S-box を用いる

2.2.4 各共通鍵ブロック暗号の概要

(1) ＤＥＳ

ＤＥＳ（Data Encryption Standard）は、現在世界中で最も広く使われている暗号方式で、一説では暗号製品の８０％にも及ぶといわれている。後述のように、１９９７年６月にＲＳＡ社の暗号解読コンテストにおいてついに解読されるということになったが、解読作業そのものは総当たり法による解読であり２.２.５で述べる計算量的安全性評価の範囲内であったため、鍵長の問題は別にしてＤＥＳの暗号化方式に対する脅威とは見なせないであろう。

元々はＩＢＭ社が１９７０年代前半に開発したものを９７７年に行われたＮＢＳ（米国商務省標準化局National Bureau of Standards、現在のＮＩＳＴ）の暗号アルゴリズム公募に提案されたものを修正したものである。当初は専用ハードウェアを必要としその内容も完全には公開されていなかったため、政府による解読のために、秘密の抜け道があるのではないかと疑われていた。その後、仕様が完全に公開され、プロセッサの能力向上と合わせてソフトウェアによる実装も多く見られるようになってきた。ただし、仕様が公開された後も、ＤＥＳの中で用いられる換字処理のテーブル設計に疑問点があることが指摘され、この点も政府による解読のためではないかという議論も行われた[3]。現在では、ＤＥＳの設計方針が差分解析などの解析技法を研究の上決定されたこと、米国政府が頑なに５６bitのＤＥＳ製品の輸出自由化に応じないことからそのような疑いはなくなったようである。

ＤＥＳ自体の特許は標準規格採用時にＩＢＭ社が放棄し、また関連特許もほとんどが期限切れであるため現在は自由に使用できる安全性の保証された暗号方式という位置付けになっている。

ＤＥＳの仕様そのものは先に述べたFeistelの開発した暗号の応用である。暗号化／復号処理は１６段の換字f（関数）と３２bitの転置処理からなる。ただし、ＤＥＳの特長としては各段に用いるサブ鍵を、５６bitの鍵→パディィを付加した６４bit鍵→分割した３２bitのサブ鍵→４８bitのサブ鍵へ拡大転置という手法で作成し、この４８bitサブ鍵でビットの置き換え処理を行っていることである。このように、複雑な処理を行うためソフトウェアによる実装では十分な速度が得られないことが問題であったが、多くの企業・学者の努力で各処理の高速化とテーブル処理の追加により現在では多くの高速化ルーチンが入手可能である。前記のデータもＲＳＡ社が自社のＢＳＡＦＥツールキットでアセンブラによる最適化処理を行った結果であり、高級言語で仕様書どおりに組んだ場合に比べて５倍程度高速といわれている。

なお、ＤＥＳは途中で一部仕様が改訂されており、現在でも文献によりS-boxの設計が異なるものがあるため注意が必要である。これらは、それ自身暗号化と復号が正常に行われたように見えるが暗号文が異なるため注意が必要である。

実装および検証にあたっては、以下の公開標準ドキュメントの内容に照らし合わせて確認することをお勧めする。

- FIPS PUB 46-2 DATA ENCRYPTION STANDARD (DES) [4]
- FIPS PUB 74 - GUIDELINES FOR IMPLEMENTING AND USING THE NBS DATA ENCRYPTION STANDARD [5]

- DES MODES OF OPERATION [6]

(2) ＩＤＥＡ

ＩＤＥＡ（International Data Encryption Argorithm）は、インターネット上でチューリッヒにあるスイス連邦技術研究所のＸ. Lai とＪ. Massey がＡscom 社と共同開発した暗号アルゴリズムで1991 年に発表された。現在、権利はＡscom Systec 社が保有している。ＩＤＥＡはもっとも普及している暗号メール ＰＧＰの標準アルゴリズムとして採用され知名度が高くなった。ＩＤＥＡでは128bit 鍵を用いるため総当たり攻撃でも今後数十年安全を保てる可能性があることが特長である。また、内部処理では64bit 平文を４つの16bit に分割し、加算・乗算ＸＯＲ・剰余処理を行っており、マイクロプロセッサとの親和性を重視していることがわかる。専用チップでも177Mbps という高速性を発揮しＤＥＳに対する優位性が強調されている[7]。また、128bit の鍵から52 個の16bit サブ鍵を生成し、各段で６個のサブ鍵を用いて８段の処理を行い、最後に４つのサブ鍵で最終変換を行うという構成になっている。

ＩＤＥＡは日米欧で特許が成立しており商用利用の場合はライセンスが必要であるが、ＰＧＰのようなフリーソフトで使用する場合は申請のみで使用料が不要である。

(3) ＦＥＡＬ-*N*/*NX*

ＦＥＡＬ（Fast Encryption Algorithm ）は1987 年にＮＴＴで開発された暗号方式で、8bit マイクロプロセッサ上で高速に処理できることが特長である。ＦＥＡＬには64bit 鍵を用いるＦＥＡＬ-*N* と128bit 鍵を用いるＦＥＡＬ-*NX* がある（ブロック長はいずれも64bit ）が、ＦＥＡＬ-*N* はＦＥＡＬ-*NX* の鍵の下位64bit を0としたものである。*N* は段数で４以上の偶数であるが8、16、32 が一般に知られている。

内部処理では128bit 鍵から*N* + 8個の16bit サブ鍵を求め、初期処理として４個、各段１個、最終変換で４個のサブ鍵が使用される。各段では*f* 関数内で鍵および入力データをそれぞれ２個と４個の8bit ブロックに割り、ＸＯＲ、加算、剰余、2bit 左巡回シフト操作を行い、結果の8bit×4ブロックを 2 bit 出力としている。

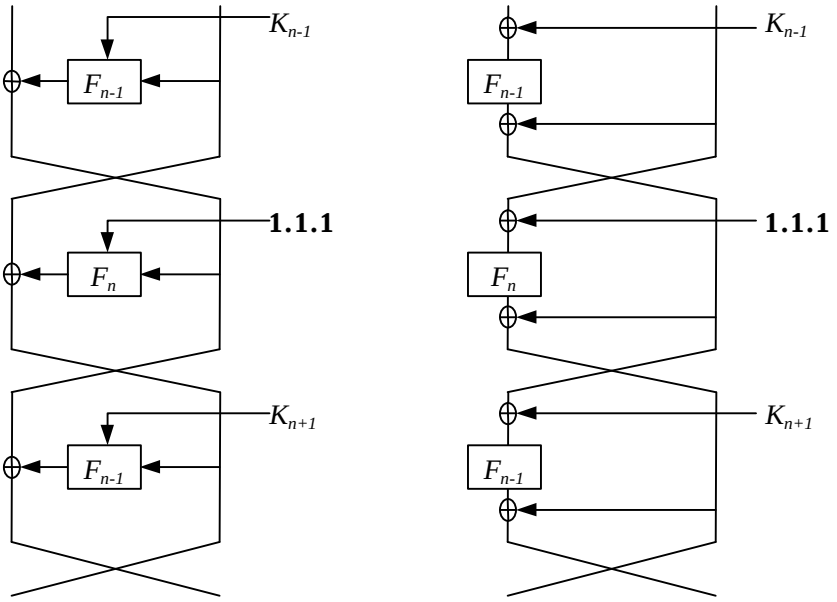
ＦＥＡＬ-8および16 に関しては差分解読法および線形解読法での解法が示されているが、ＦＥＡＬ-32 に関してはいずれも不可能であることが証明されている。ＦＥＡＬはＮＴＴが開発したこともあり、国内で安全に使用できる強力な暗号としての選択肢に入るものである。また、専用チップも国内で販売されており応用製品も販売されているが、8bit 処理を多用しているため上記ＩＤＥＡチップに比べて１／３の能力にとどまっている。ただし、これまでに多くの解読実績があることから、ＦＥＡＬの安全性を疑問視する向きも多く、今後の信頼性回復が最重要課題である。

(4) ＭＵＬＴＩ２

ＭＵＬＴＩ２（Multimedia Encription 2）は日製作所が1989 年に発表した暗号方式である。256bit という非常に長い鍵を使用することが特長である。

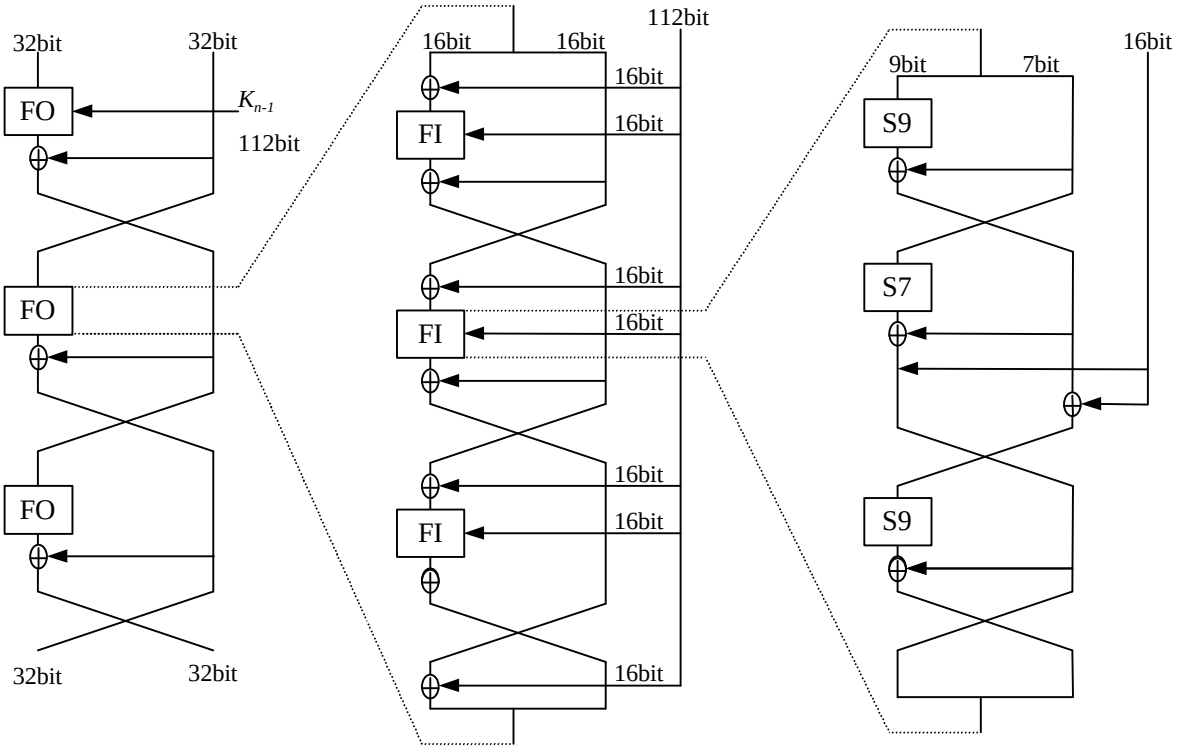
基本構造として32bit プロセッサ上での高速処理を目指したもので、同様のアルゴリズムがデジタル衛星放送などで使用されている。各段では、32bit の加減算・ＸＯＲ・ＯＲ・右巡回シフト（１、２、８、16bit ）を行って、攪拌効果を高めるよう工夫されている。また、利用者が安全性と処理速度のバランスで繰返し回数を選択できるようになっている。

なお、日立製作所ではＭＵＬＴＩ３以降の暗号については、受託開発方式であるとしておりアルゴリズムも公開されていない。



MISTY1の基本構成　　MISTY2の基本構成

図 .2-4　ＭＩＳＴＹの基本構成



MISTY2 Level1 (8段)　　MISTY2 Level2 (3段)　　MISTY2 Level3 (3段)

図 .2-5　ＭＩＳＴＹの入れ子構造

(5) **ＭＩＳＴＹ**

ＭＩＳＴＹは三菱電機が 1992 **年に発表した暗号アルゴリズムで、ＭＩＳＴＹの名前は開発者の頭文字をつなげたものである。ＭＩＳＴＹでは** 128bit **鍵をから** 256bit **の拡大鍵を生成して使用し、ガロア体上の演算を基本に演算要素としてＡＮＤ、ＯＲ、ＸＯＲとテーブル参照のみで** F **関数を設計している。差分解析法および線形解析法（開発者の一人・松井が考案）に対する証明可能安全性を持つことが特長である。**

これは、ＭＩＳＴＹが３重の入れ子構造で、サイズの小さい、平均差分／平均線形確率の小さなテーブルを組み合わせて構成しているために可能となった。ＭＩＳＴＹにはＭＩＳＴＹ１とＭＩＳＴＹ２の２種類あり、ＭＩＳＴＹ１は通常の Feistel **型であるが、ＭＩＳＴＹ２では処理の並列化が可能になるよう、各段次のように並べ替えて実装している点も特長である。**

ここで各段のＦ関数（32bit **入出力：ＭＩＳＴＹではＦＯ関数と呼ぶ）は、さらに３段の** 16bit **入出力のＦＩ関数からなり、ＦＩ関数はまた３段のＳ関数（Ｓ７およびＳ９）からなるという構成をとっている[8]、[10]、[11]。**

(6) **ＳＸＡＬ／ＭＢＡＬ**

ＳＸＡＬ（Substitution Xor Algorithm**）とＭＢＡＬ（**Multi Block algorithm**）はともにローレルインテリジェントシステムズ社の平田氏の考案になるもので、内部処理は８**bit **単位の換字・ＸＯＲ・転置処理で構成されるため、マイクロプロセッサ上のソフトウェアで高速に実現できる暗号方式である[12]。**

ＳＸＡＬ（段数を表しＳＸＡＬ８とも呼ばれる）では、サブ鍵として初期処理と最終変換に２個、各段に１個使い合計 10 **個のサブ鍵が用いられる。各段は図** 2.2-6 **のように** 32bit×２**入力２出力の基本アルゴリズムｆで、入力の左右両方を換字・ＸＯＲする処理である。一方ＭＢＡＬではこの基本アルゴリズムを拡張して可変長バイトの入力に対して処理を行う拡張基本アルゴリズム** $F_m(K)$**が用いられ、更に段数を３回に減らしても安全性を確保できるよう、段間で両端データをＳＸＡＬで暗号化処理を行う[9]。**

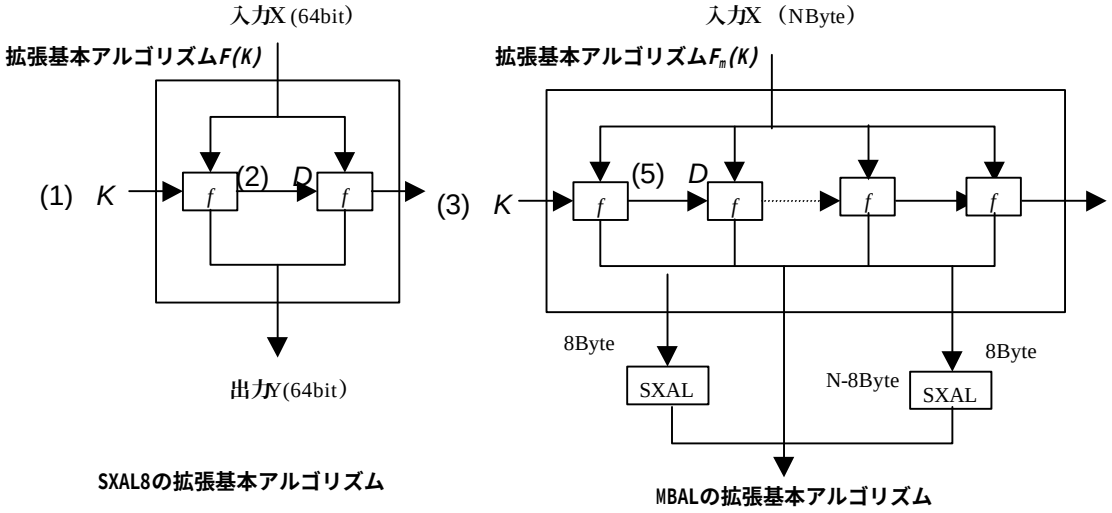


図2.2-6　ＳＸＡＬ／ＭＢＡＬの 基本アルゴリズム

(7) **ＲＣ２**

ＲＣｘという暗号方式はＲＳＡ開発者の一人である Rivest がＤＥＳの欠点を克服するために開発した共通鍵暗号方式である。現在ＲＣ２、ＲＣ５（ともにブロック暗号）、ＲＣ４（ストリーム暗号）の３つが広く使われている。また、ＲＣ１は実現できず、ＲＣ３は開発中に破られたため欠番となっている。

ＲＣ２は長くそのアルゴリズムが非公開とされ、ＲＳＡ社のツールキットを使用しなければならなかった。ただし、鍵長が可変であるため、米国からの輸出規制（40bit）に早くから適合し、製品に採用されている場合も多い。国内でもＤＥＳが解禁される前からポピュラーな存在であった。

最近になってＩＥＴＦでの標準化の提案にあわせて、ＲＣ２のアルゴリズムが Rivest 自身により公開された[13]。これによると、ＲＣ２は１～128Byte の可変長の鍵を使用可能で、64bit ブロック単位に処理を行う暗号方式である。また、内部の処理単位として 16bit 語のマイクロプロセッサ上で実装が簡単になるように考慮されている。鍵の展開処理においては 16bit 長、８bit 長どちらでも処理できるようになっているが、ランダムなバイト列を作成するために、π（円周率）に基づく 256Byte の数列を用いている点がユニークである。

ＲＣ２の攪拌処理は単純な繰返しではなく、５回の Mix 処理－１回の Mash 処理－６回の Mix 処理－１回の Mash 処理－５回の Mix 処理という上下対称の処理ロジックとなっている。Mix 処理では、加算・ＡＮＤ・左巡回シフト操作（１、２、３、５bit）を行い、Mash 処理では加算とＡＮＤ処理を行う。

(8) **ＲＣ５**

ＲＣ５は Rivest の開発した最新の暗号方式で、処理速度、スケーラビリティに配慮した処理を基本とし、ブロック長、鍵長、段数がいずれも可変という特長を持つ。これらは、処理ブロックプロセッサのワード長の２倍で処理することで高速性を持たせ、鍵長と段数は処理速度と安全性のバランスで利用者が決められるようにしたものであり、ブロック長により十分な攪拌効果を得るための段数が変わる。ＲＣ５の基本アルゴリズムは驚くほど簡単で、ＸＯＲ操作、データ依存左巡回シフト操作、サブ鍵の加算、左右交換処理だけからなっている。このため、コードサイズ、メモリサイズいずれも少なくすることができ、ＩＣカードなどでも利用することができる。ＲＣ５の基本的な安全性はデータ依存のシフト操作によるものであるが、未だ発表から年月が経っていないこともあり安全性についての研究はまだ十分であるとは言い切れない。また、実装例ではＣＢＣ処理を基本処理に組み込んでおり、データ依存シフト処理の危険性回避と見ることもできる。

また、ＲＣ５は当初よりアルゴリズムが公開されているが、ＲＳＡ社は商標および特許を登録している。ただし、ＲＳＡ社の特許に関する権利の放棄をＩＥＴＦが標準化採用の見返りとして要求していることもあり、今度の動向が注目される。なお、日本国内においてはＲＣシリーズ暗号は出願されておらず、ＲＣｘという商標も登録申請されていないようである。ＲＣ５の完全な仕様書とプログラム例がＩＥＴＦにより[14]に公開されている。

(9) **ＥＮＣＲｉＰ**

ＥＮＣＲｉＰはＮＥＣが開発し 1997 **年にＩＳＯ登録を行った** 64bit **鍵、** 64bit **ブロック長の暗号方式である（段数については登録内容は８段であるが詳細は不明）。**

詳細が公開されていないため、サブ鍵の生成などの処理内容は不明であるが、基本となる攪拌処理内容について関連特許（出願 Ｈ７-206372）**が公開されている。それによれば、内部の基本処理としては** 32bit **単位でのＸＯＲ、ＡＮＤ、左巡回シフト操作（１、７、**11、22bit）**で構成されており、特許出願意匠もこの組み合わせによる高度なビット撹乱効果である。シフト操作が１、７（=８－１）、**11（=32+１÷３）、22（=11*２）**と工夫することで、１段あたり非常に多くの入力と鍵のビットが出力ビットに影響を及ぼす設計となっているようである。**

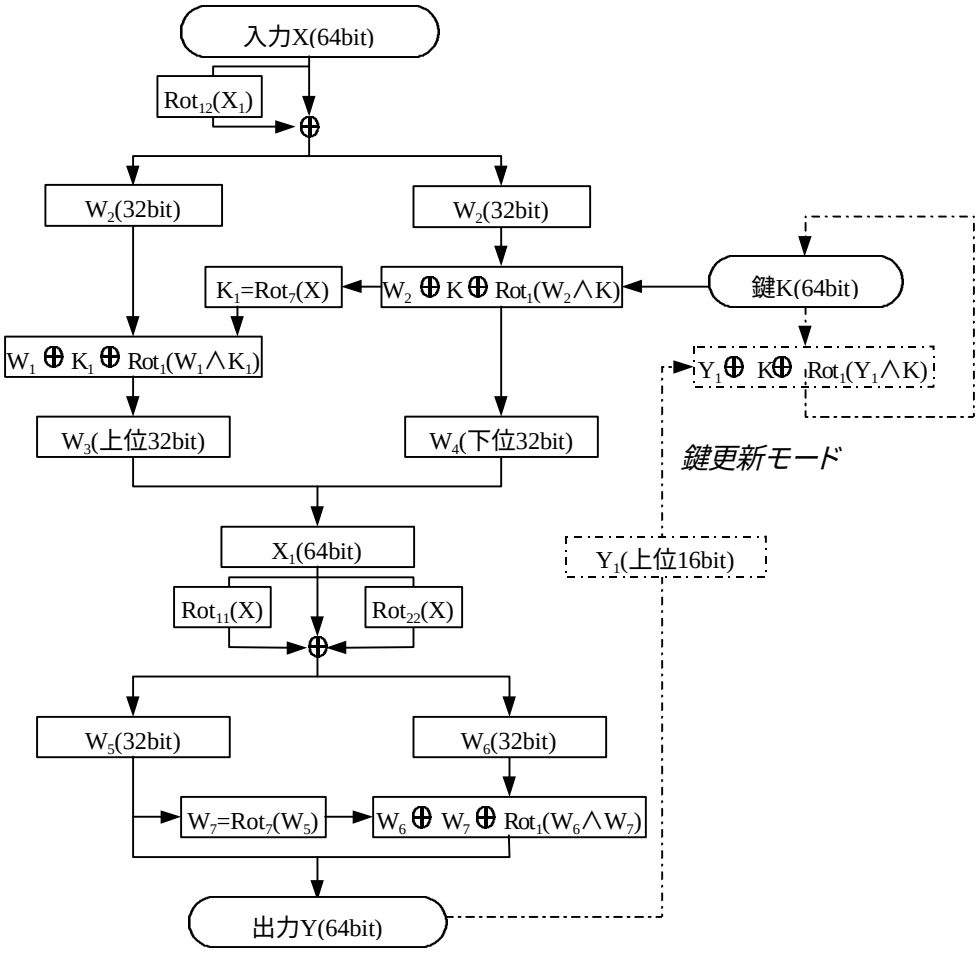


図2.2-7　ＥＮＣＲｉＰで 使用さ れるアルゴリズム

(10) **ＳＡＦＥＲ**

ＳＡＦＥＲ（Secure And Fast Encryption Routine**）は、ＩＤＥＡの開発者の一人** Massey **が** Cylink **社のためのオープンな暗号方式として開発したものであり、現在はフリーで使用することができる。また、鍵長を表しＳＡＦＥＲ** K-64 **と呼ばれることもある[15]。**

ＳＡＦＥＲの特長は、全て８bit **単位で処理していることと、擬似** Hadamard **変換**

（ＰＨＴ：Pseudo-Hadamard Transform）と呼ばれるあまり知られていない線形変換処理をブロック全体にたすきがけしている点、そして 45 を底とする指数・対数演算と 257 を基数とする剰余処理を採用している点である。実際にはこれらの処理はテーブル参照により行われるため処理そのものは高速である。

段数は可変となっており６段以上の任意の値にすることができる。各段では 64bit サブ鍵を２個、最終変換で１個使用し、１つの 64bit 鍵から $2r+1$（r は段数）このサブ鍵を生成して使用する。サブ鍵の生成では鍵バイアスというテーブル参照で弱い鍵をなくすようにしている。

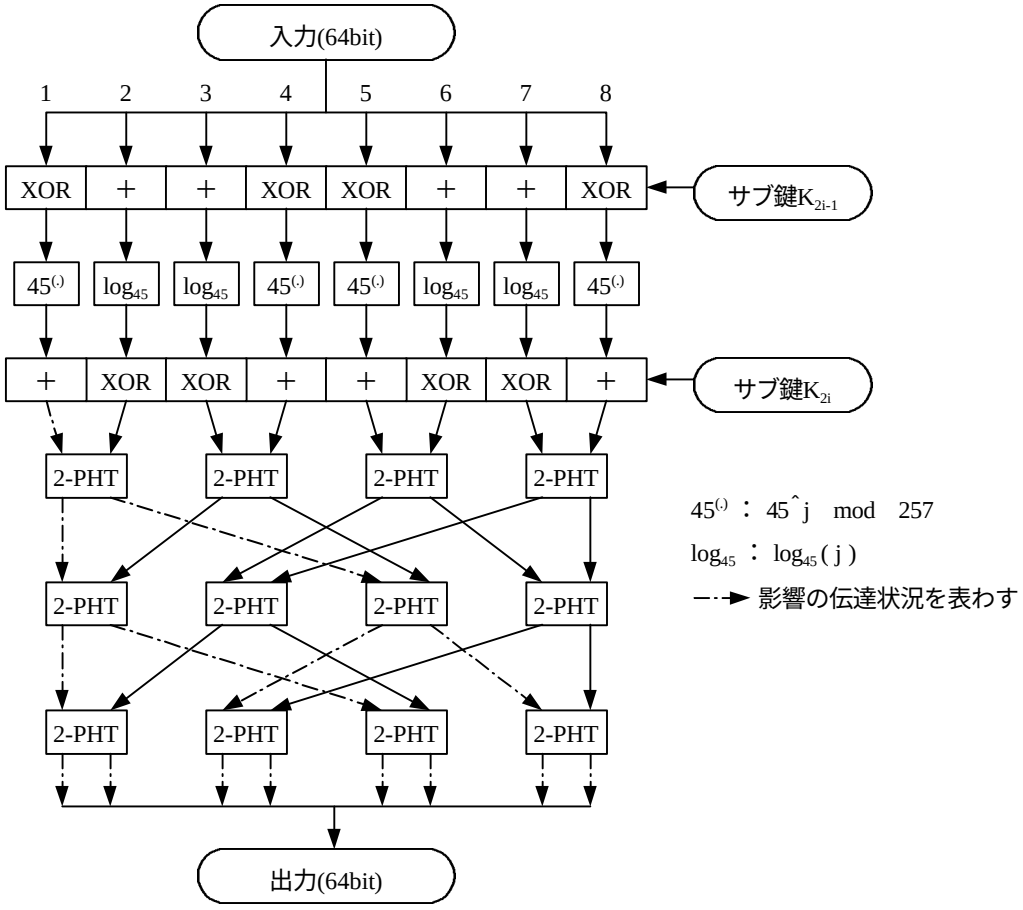


図2.2-8　ＳＡＦＥＲの 各段の 処理

(11)　Blowfish

Blowfish は民間の暗号学者で”Applied Cryptography”の著者として知られている Bruce Schneier が開発しフリーソフトとして公開している 64bit ブロック暗号方式である[16]。

鍵長は可変となっており 32bit から 448bit まで可能である。また、段数は 16 段固定で、各段で１個、最終変換で２個の 32bit サブ鍵を使用する。S-box は 32bit のものを４×256 個使用する。特長は高速性であり、１Byte あたり処理クロック数はＲＣ５や Khufu よりも少ないといわれている。f 関数として８bit ブロックでの加算、ＸＯＲ、232 を基数とする剰余を利用している。

最近ＩＥＴＦにも標準化採用のための提案がなされており、ブロック長を伸ばす改良が加えられているとも伝えられているが、安全性に対する評価は定まってはいない。

(12) Skipjack

Skipjack は、ＮＳＡが Capstone プロジェクトで使用する Clipper チップで使用するために開発した暗号化アルゴリズムである。単純に安全性を比較すると 80bit 長のキーを使用して 32 回データを攪拌するためＤＥＳよりも安全性が高いと考えられる。

一方 Skipjack のアルゴリズムについては公表しない方針が採られ、ソフトウェアでの実現は不可能で、認可されたチップメーカの製造するハードウェアでのみ実現可能なため、採用には非常に大きな問題点がある。特に、アルゴリズムが公開されない点は、政府のみが知る解読法を隠すためではないかという批判があり、専門家の信頼性が得られていない。このため政府は、民間の暗号学者に Skipjack のアルゴリズムの研究を依頼し、レポートが公開された[17]。これによれば「開示された範囲で調査した結果、ショートカット法などの攻撃に対して特別なリスクは見当たらない」ということであった。

(13) Khufu／Khafre

Khufu／Khafre は Xerox 社が開発した暗号で、マイクロプロセッサ上で非常に高速に処理が行えることが特長である。

内部のデータ処理アルゴリズムは同一であるが、256 個の S-box が Khafre では固定であり、Khufu は鍵から導出するという違いがある。Khufu／Khafre の f 関数は驚くほど単純であり、S-box によるＸＯＲ処理と８bit の左旋回シフト処理のみである。

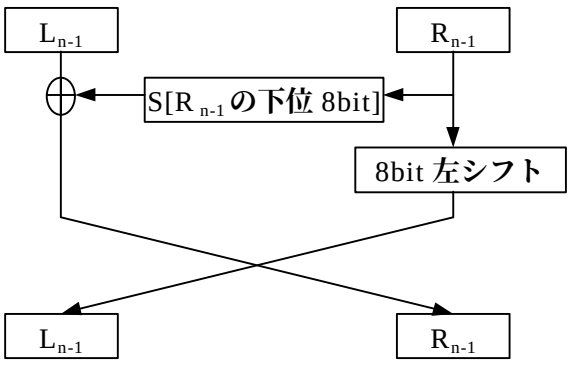


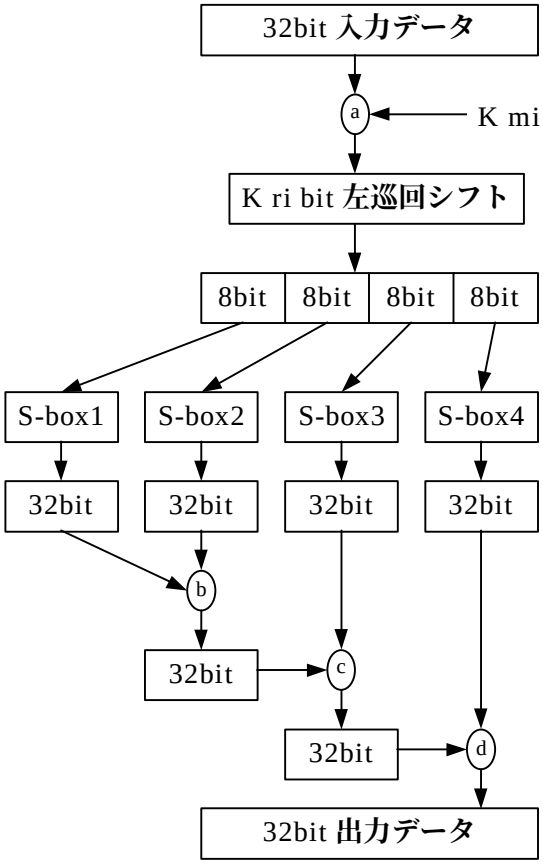
図2.2-9 Khufu／Khafre の処理アルゴリズム

正式の仕様については未確認であるが、Khufu のソースコードといわれるものがインターネット上で入手可能である[18]。

(14) ＣＡＳＴ

ＣＡＳＴは Northern Telecom が開発した Feistel 型暗号で、各段の f 関数および鍵スケジュール処理を工夫することでＤＥＳよりも少ない段数で、より良い特性が得られるよう設計された。また、各パラメータは 32bit プロセッサでのソフトウェア処理に向くよう決定されている。

ＣＡＳＴの各段の処理は図 2.2-10 のようになっており、８bit 入力－32bit 出力の S-box 変換と加算、減算、ＸＯＲのみからなっている。



CAST-128 の場合の各処理

TYPE	a	b	c	d
1	＋	XOR	－	＋
2	XOR	－	＋	XOR
3	－	＋	XOR	－

TYPE1 は 1,4,7,10,13,16 段目、
TYPE2 は 2,5,8,11,14 段目、
TYPE3 は 3,6,9,12,15 段目に使用する

図2.2-10　CAST の 基本処理

ＣＡＳＴは元々設計手法を指す語として使われ、ブロック長、鍵長、段数などが可変であるが、一般的に使用されているＣＡＳＴ暗号は、64bit ブロック長、128bit 鍵長、16 段処理のＣＡＳＴ-128 と呼ばれるものである。この場合、サブ鍵として各段で 32bit のＫｍと５bit のＫｒを用いる。

ＣＡＳＴに関する資料は、インターネットの[19]、[20]より入手可能である。

(15)　**ＧＯＳＴ** 28147-89

ＧＯＳＴ 28147-89 は旧ソビエト連邦（現ロシア共和国）の政府標準暗号として、Zabotin、Glazkov、Isaeva らによって開発された暗号で、1989 年に制定された。

かつては、詳細なアルゴリズムなどまったく知られていなかったが、ソビエト連邦崩壊後は一転して商用利用に向けて積極的なＰＲ活動が行われているようである。もともとはロシア語の仕様書しかなく正確な翻訳が無かったが、Sun Microsystems の Diffie により英訳された仕様書[21]が現在入手可能である。よくＧＯＳＴとだけ呼ばれるが、ＧＯＳＴはソビエト連邦における政府標準規格（GOsudorstvennyi STandard）のことであり、ＪＩＳやＡＮＳＩなどと同じように多くの規格が存在する。この中の標準暗号がＧＯＳＴ 28147-89 である。

ＧＯＳＴ 28147-89 の特長は、次の通り。

- 32 段と段数が多い
- *f* 関数では mod32 の加算、11bit 左巡回シフト、および８つの４bit 入出力 S-box 変換のみからなる

- 鍵として 256bit のプライマリ鍵と 512bit のセカンダリ鍵が使用できる
- プライマリ鍵は８個の 32bit サブ鍵に分割され、昇順に３回、降順に１回使用される
- セカンダリ鍵は標準仕様には含まれないが順列テーブルとして８つの S-box を構成する
- ＥＣＢモードのほかにＣＦＢ、ＯＦＢモードが定義されている

なお、これまでにＧＯＳＴ 28147-89 の欠点として変異テーブルのセカンダリ鍵の定義が十分でない点、サブ鍵を合わせるとマスター鍵が求められるため鍵差分攻撃に弱い点が指摘されている。また、プライマリ鍵とセカンダリ鍵を合計すると 768bit となるが、B.Schneier の分析[22]によると、実効鍵長としては 610bit 相当である。

2.2.5 共通鍵ブロック 暗号の 強度検討

1997 年６月 17 日火曜日に、同年１月 29 日よりＲＳＡ社が行っていた暗号解読コンテストにおいて、ついにＤＥＳが解読された[23]。解読者はユタ州ソルトレイクシティの Michael K. Sanders で、彼はＤＥＳＣＨＡＬＬという解読プロジェクトチームの作成した解読プログラムを会社（iNetZ 社）にある FreeBSD を搭載した Pentium 90MHz（16Mbyte ＲＡＭ）のマシンで実行し、正しい鍵 ”85 58 89 1a b0 c8 51 b6” を見つけ出すことができた。

同プロジェクトチームでは、インターネットを通じて解読作業のボランティアを募集し、応募者に解読プログラムと探索すべき鍵の範囲を配付する方法で、２月の 18 日より解読作業を開始し、累計で約７万８千台のマシンを動員した。解読内容の報告書によると、ピーク時で１日あたり約１万４千台のマシンが 600 兆個の鍵をテストしたそうである[24]。

ＤＥＳの鍵は 56bit 長であり、これは数字で表わすと 72,057,594,037,927,936（７京２千兆）にのぼる。今回ＤＥＳＣＨＡＬＬチームが探索した鍵の総数は 17,731,502,968,143,872（１京８千兆弱）であり、これは全鍵空間の 24.6%に相当するわけで、今回は幸運にも恵まれたといえる。しかしながら、ＤＥＳＣＨＡＬＬチーム以外でもスウェーデンの SolNet チーム（彼らはＤＥＳＣＨＡＬＬチームと異なり北米以外のボランティアが利用できる解読プログラムを配付した）もおおよそ１京個の鍵を探索し終えており、また、個々の研究機関においてもイリノイ大学やペンシルバニア州立大学、カーネギーメロン大学などで、それぞれ千兆個を超える鍵をチェックしている。

また、これに引き続き、56bit ＲＣ５も 10 月 19 日 13:25（ＧＭＴ）に、Project Bovine と呼ばれるグループにより解読された[25]、[26]。このように、現在においてはわざわざ超高速のスーパーコンピュータを用意したり、専用の解読装置を何万個も製造することなく、机の上にあるパソコンのアイドルタイムを活用することでも、十分な脅威となりうることが明らかになった。今日では、政府の機関や大企業など数十万台規模のＰＣをネットワークで接続し、そのほとんどが夜間や土日などには利用されないといった状況のなかで、組織的に（あるいは内部のハッカーに）より同様の解読作業が行われないとは誰もいえない状況にある。

これらのことを考慮すると、現在利用されている 56bit から 64bit の鍵長の暗号方式で

は、総当たり法に対して十分な強度であるとはいえなくなっている。1996 年１月に暗号の専門家を集めたグループが、同様の研究を行っており、やはり 56bit の鍵長の危険性が指摘され、今後 20 年間の期間内で少なくとも 90bit 以上の鍵を利用することを推奨している[27]。

ここでは、上記報告を元に、以下の条件の下での総当たり法での解読時間の推定を行い、暗号の安全性を評価するための基準値を求める。

＜前提条件＞

a) アルゴリズム間の速度の差はビット数換算で多く見積もって７bit 程度である。

b) ＤＥＳでは線形解読法で既知の平文に対して計算量が 12bit 分の 1 程度に少なくなることが知られている
逆にいえば、この程度の強度の冗長性があれば線形解読法は脅威にならないことを示している

c) 各方式は効率の良い解読方法が極力無いように設計されている必要がある

d) 計算機性能は“同一コストで 18 ヶ月で２倍になる”Moore の法則“がこれまでも、今後も成り立つものと仮定できる

表2.2-2 総当り法による秘密鍵暗号の解読時間（ＤＥＳ56bit 相当）

攻撃者	個　人	企　業	国　家
予　　算	100 万円	10 億円	１兆円
解読装置	1,000 台の PC 又は FPGA	FPGA／ASIC	ASIC
１９９５年	1.5 年	６分	0.4 秒
１９９８年	135 日	90 秒	0.1 秒
２００１年	34 日	23 秒	23 ミリ秒
２００４年	8.4 日	６秒	６ミリ秒
２００７年	（☆）2.1 日	1.5 秒	1.5 ミリ秒
２０１０年	12.6 時間	0.4 秒	0.4 ミリ秒
２０１３年	3.1 時間	0.1 秒	0.1 ミリ秒
２０１６年	47 分	25 ミリ秒	25μ秒
２０１９年	12 分	6 ミリ秒	６μ秒
２０２２年	3 分	1.5 ミリ秒	1.5μ秒

基準としてＤＥＳを選択しているので、ＤＥＳに比べて速度が５倍、10 倍の性能の暗号の場合、上記表の５分の１、10 分の１の時間で解読できる事に注意が必要である。これにより、現在のＤＥＳ 56bit 相当の暗号では今後 10 年程度で（☆）、誰でも数日で解読できるレベルとなることがわかる[28]。

表2.2-3　今後20年間情報を守るのに必要とされる90bit鍵での予測値

攻撃者	個　人	企　業	国　家
予　　算	100万円	10億円	1兆円
解読装置	1,000台のPC又はFPGA	FPGA／ASIC	ASIC
１９９５年	257億年	20万年	196年
１９９８年	64億年	5万年	49年
２００１年	16億年	1.2万年	12.2年
２００４年	4億年	3,064年	3年
２００７年	1億年	766年	280日
２０１０年	2,560万年	191年	70日
２０１３年	640万年	48年	17.5日
２０１６年	160万年	12年	4.4日
２０１９年	40万年	3年	1日
２０２２年	10万年	273日	6.5時間

上記の結果より、仮にＤＥＳよりも100倍（7bit）以上高速な暗号方式であっても、112bit以上の鍵を使用すると今後20年程度なら安全に使用できる可能性があると考えられる。

2.2.6 今後の展望　～ＡＥＳとその方向性～

最新の共通鍵暗号が採用している64bitの鍵長／ブロック長は、32bit汎用プロセッサ上での処理効率が非常に高くなることを利用している。これは、多くのアルゴリズムにおいて用いられているインボリューション（2.2.2参照）では、入力を２（もしくは2^n）個のビット列に分割し処理を行っているためで、分割後のデータがが32bitや16bitとなりＣＰＵやＯＳのワード長と一致するため、処理が容易になり速度が向上することによる。今後、ＣＰＵ、ＯＳともに64bit化が予定されており、その時には128bitの鍵長／ブロック長の暗号も速度向上が見込まれる。

一方、標準としての地位を築き上げたＤＥＳ（一説には暗号製品の80%を占めるといわれる）も、開発から20年経過し、仕様が最新のコンピュータにマッチしない、鍵長／ブロック長が短く安全面で不安であるなど、必ずしも万全とはいえなくなった。特に、近年登場した多くのアルゴリズムはより長い鍵長や、可変の鍵長を持ち、特に米国の輸出規制を容易にクリアできるような物が登場し始めた。ＤＥＳ自体も、３つの鍵を用いるTriple-ＤＥＳ方式で、さまざまな分野への適用が図られている。

このような状況で、1997年1月、ＮＩＳＴ（米国商務省 標準化局）は、ＤＥＳに替わる新たな暗号の標準を模索し、Advanced Encryption Standard（ＡＥＳ）として、一般の意見を取り入れながら公募プログラムを開始、1997年9月12日に正式の応募要綱を発表した[29]。

a)　ＡＥＳの概要

ＤＥＳの後継となるブロック暗号方式の確立

一般からの意見を参考に、公募形式で進める

2000 年をめどに標準化完了し 2020 年から 30 年程度の利用を前提とする

b) ＡＥＳ候補アルゴリズムの必要条件

共通鍵暗号であること

ブロック暗号であること

少なくとも鍵長－ブロック長が以下の組み合わせが可能なこと

128-128bit、192-128bit、256-128bit

c) アルゴリズム評価基準

① 安全性

i. 一同一鍵長・ブロック長での他のアルゴリズムとの比較

ii. 平文のランダム化したものとと暗号文の類似性比較

iii. 数学的安全性の根拠

iv. 評価プロセス中に指摘される安全性に関する問題

② コスト

i. ライセンス要件

ii. 計算効率性

iii. 実装時必要なメモリ量（ハードウェアはゲート数）

③ その他アルゴリズムの特長

i. 各種アプリケーション対応の柔軟性

ii. ハードウェア、ソフトウェアとの親和性

iii. シンプルな構造

d) ＡＥＳ標準化スケジュール

① 1997 年 月 日上記要件の草案と意見募集

② 1997 年4月1 5 日概要説明と収集意見に対するデ ス カッションのためのワークショップ開催

③ 1997 年9月 2 日募集要項発表

④ 1998 年6月 5 日応募締切

1(998 年4月 5 日までの応募に対しては５月 5 日までに必要な修正内容が提示される）

⑤ レビューの上、全ての応募内容を公開し意見収集

⑥ 1998 年夏第１回ＡＥＳ候補カンファレンス開催

応募者による説明と質疑応答、意見収集

⑦ 技術評価第１ラウンド

カンファレンスの結果レビュー、候補の絞り込み

⑧ 第１回カンファレンスの６ヶ月後、第２回ＡＥＳ候補 カンファレンス開催

ＮＩＳＴによる技術評価内容の議論、より一層の絞り込み

⑨ 技術評価第２ラウンド

安全性、計算効率や知的所有権などを考慮し最大５つまで候補の絞り込み

⑩ 技術評価第２ラウンドの結果発表後６～９ヶ月後､第 ３回ＡＥＳ候補カンファレンス開催

⑪　ＮＩＳＴによる最終選定

現在のスケジュールでは、ＤＥＳのＦＩＰＳ見直しの行われる 1998 年末までに結論は出ない。そのためＦＲＢ（米国連邦準備基金）は、ＡＥＳ案制定までの間 Triple-DES を使用できるよう求めている。一方、ＤＥＳからＡＥＳへのスムーズな移行を求める声も多く、今後暗号応用システムを開発する立場からすると、両方に対応が容易なシステム設計が必要となると思われる。なお、現時点では強力な暗号アルゴリズムについては、米国内でも海外からのアクセスのある Web サイトなどにコードをおくことが困難であるが、この点については現時点ではＮＩＳＴとしては特に配慮はしていない（情勢が流動的であるともいえる）。

もともとＤＥＳに代わるということは、「機密扱い（軍用など）でない重要な情報の暗号化」に用いるべきものであるため、公開できる程度の安全性である可能性も否定はできない。ただし、商用化の上では標準化されることによる爆発的な普及がＤＥＳのように起きるかどうかも疑問視されている。これは、ＤＥＳ開発当時に比べ様々な暗号方式の選択肢が非常に広くなってきたことも関係する。今後は、コストと安全性を十分検討した上でいろいろな選択肢の中から暗号アルゴリズムを選ぶということが一般化するかもしれない。

2.2.7 共通鍵ブロック 暗号方式の 利用モード

共通鍵ブロック暗号方式は、同一の鍵と平文の組み合わせでは常に同じ暗号文が得られる。したがって頻度の高いコードや金額など、推定されやすいデータを繰返し含む平文の暗号化には向かない。

ＩＳＯおよびＮＩＳＴではＤＥＳの利用モードとして（図 2.2-11）、これらの欠点を補う利用モードを標準化している[6]。

①　ＥＣＢモード（Electronic Code Book）
オリジナルのＤＥＳ利用モード。

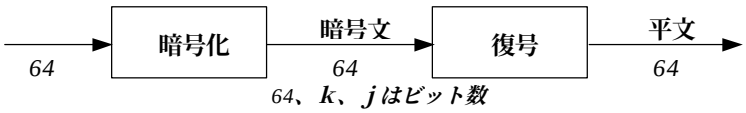
②　ＣＢＣモード（Cipher Block Chaining）
平文の暗号化前に、前のブロックの暗号文をＸＯＲしてから暗号化を行う。
同一の平文であっても、ランダムな暗号文をＸＯＲするため異なる暗号結果が得られる。
なお、最初のブロックの暗号化には初期値（または初期化ベクタ）（*IV*）と呼ばれる値をＸＯＲする。解読には暗号化鍵と *IV* が必要である。

③　ＣＦＢモード（Cipher Feedback Mode）

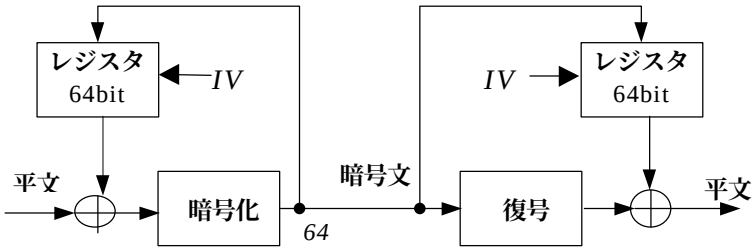
④　ＯＦＢモード（Output Feedback Mode）
上記③、④の２方式は、ＤＥＳ暗号化を平文に対してではなく、レジスタの内容に対して行う。平文の暗号化は、選られたＤＥＳ暗号結果のランダムな出力ビットストリームとＸＯＲすることによりなされるため、一種のストリーム暗号方式である。すなわち、ＤＥＳのビット撹乱・ランダム化による非線型変換を、乱数発生器として使用するものである。レジスタには、ＣＢＣモードと同様の初期値（IV）をセットする。復号には暗号化鍵と IV が必要である。これら２方式は、誤り訂正効果がある、暗号化・復号いずれもＤＥＳの暗号化ロジックですむとい

う長所を持つものの、処理速度が極端に低下するという欠点もあわせもつことが特徴である。したがって、比較的低速な回線の暗号化や、専用チップを用いる暗号化に向いている。

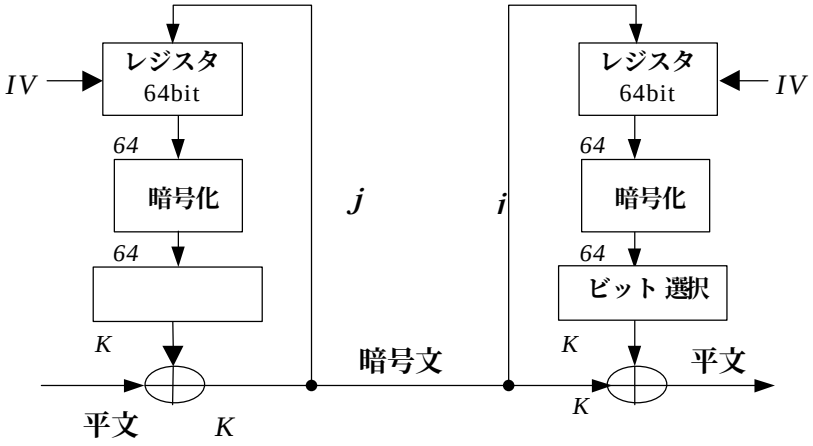
(a)　ＥＣＢモード



(b)　ＣＢＣモード



(c)　ＣＦＢモード



(d)　ＯＦＢコード

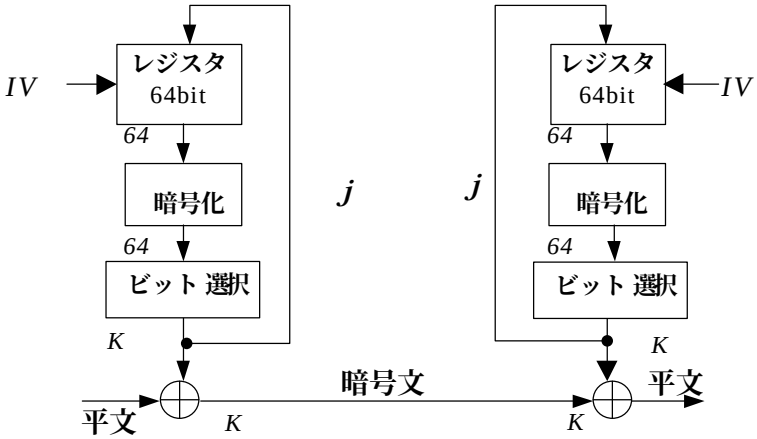
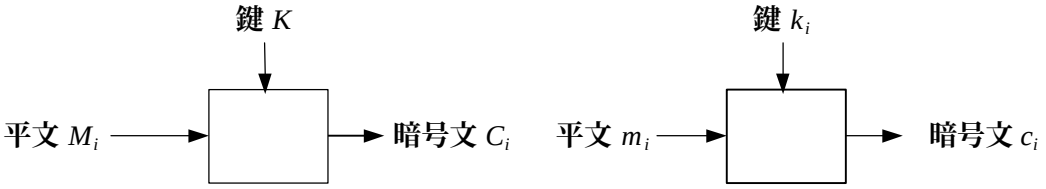


図2.2-11　利用モード

2.2.8 共通鍵ストリーム暗号の定義と原理

ストリーム暗号とは、平文を１bit ないし１Byte 単位に、ある特定の鍵系列により逐次的に暗号化する手法であり、逐次暗号とも呼ばれる。



ブロック暗号
鍵 K は **固定**
$M = M_1\ M_2\ M_3\ \cdots$
M_i は数十 bit

ストリーム暗号
鍵 k_i は **逐次変化**
$M = m_1\ m_2\ m_3\ \cdots$
m_i は１～数 bit

図2.2-12　ブロック暗号と ストリーム暗号

最も単純なストリーム暗号は、十分に長い乱数系列を用意し、暗号化と復号に際して平文のｎBit 目と乱数列のｎBit 目をＸＯＲするもので、一般にバーナム暗号と呼ばれている。現在のところ、暗号化と復号には同一の鍵系列を用いる物しかなく、あえて共通鍵をつけて呼ばれることはないので、ここでは以下ストリーム暗号と呼ぶ。

ストリーム暗号では、平文と暗号文の対応が分かれば、用いている鍵系列が直ちに求まる。そのため、安全性は主としてこの鍵系列として用いる乱数の性質に依存する。一方で、乱数列の生成は一般にブロック暗号の攪拌処理に比べて高速であり、ソフトウェアのみ用いた場合でも高速な通信経路に追随する処理速度が得られる。完全に使い捨ての乱数列を用いたバーナム暗号は、鍵系列となる乱数列が秘匿される限り、理論的に解読不可能であるが、鍵系列の長さが平文と同じ物になるためどのようにこれを生成、配送、保管するかが問題となる。

一般的に使用されるストリーム暗号は、特定の長さの鍵を入力とする乱数発生器の出力を鍵系列として用いる。乱数発生の種々の方式については 2.5 で詳細に述べるので、ここでは乱数の望ましい性質として以下の点が重要であることを述べるにとどめる。

① ０、１のなど頻度性（０と１の出現確率がそれぞれ 0.5 に近いこと）
② 長周期性（繰り返しの発生周期が非常に長いこと）
③ 非線型性（乱数発生に非線型操作を含んでいること）
④ 無相関性（乱数発生器の入出力間および出力間に相関が無いこと）
⑤ 線形複雑度が大（線形フィードバックレジスタ表現の場合のビット数が長いこと）

また、歴史的にも性能的にもストリーム暗号の適用分野は音声通話やデータ通信時の暗号方式として用いられることが多く、そのため音声欠落やデータの再送を極力押さえるため、自己同期方式を用いることがある。2.2.7 で述べたＣＦＢモードやＯＦＢモードも自己同期ストリーム暗号方式である。

2.2.9 各種ストリーム暗号の比較

ストリーム暗号は、共通鍵ブロック暗号方式のＤＥＳのように普及した方式が無く、一般的に知られている種類も少ない。ここでは比較的良く使われている方式について若干の説明を行う。

(1) **ＲＣ４**

ＲＣ４は他のＲＣシリーズと同様に、Rivest により 1987 年に開発されたストリーム暗号方式である。鍵長は 2,048bit までの可変長である。

ＲＣ４のアルゴリズムは秘密とされてきたが、現在では同等のソースコードの入手が可能である[29]。ただし、北米においてはＲＣ４の商標権をＲＳＡ社が保持しているため、インターネット上などではＡＲＣＦＯＵＲ（A(not)-ＲＣ４）という名称で用いられることもある。ＲＣ２と同様に、可変長の鍵を用いることが出来るため、米国の輸出規制を逃れることができ、Lotus Notes など多くの製品にインプリメントされている。ＲＣ４では標準で８×８の８bit S-box を乱数発生器として用い、ＯＦＢモードで動作する[22]。

乱数発生ロジックは以下のように非常に簡単な物となっている。

$i = (i + 1) \bmod 256$

$j = (j + S_i) \bmod 256$

swap S_i S_j

$t = (S_i + S_j) \bmod 256$

乱数 $K = S_t$

この K を平文（または暗号文）とＸＯＲすることで暗号化（復号）を行う。S-box の初期化は、鍵を用いて以下のロジックを実行することで行われる。鍵はあらかじめ $K_0, K_1 \cdots K_{255}$ の配列に必要回数繰り返し格納される。

for $i = 0$ to 255

$j = (j + S_i + K_i) \bmod 256$

swap S_i S_j

上記の処理は８bit のＲＣ４のものであるが、16×16 の 16bit S-box を用いるものも定義が可能である。この場合、65,536 個の S-box の初期化のために $K_0, \cdots K_{65535}$ の鍵を用いることが可能で、鍵長は最大１Mbit にも及ぶが、初期化処理に必要な時間は増大する。一方、処理の単位が２Byte 単位となるため処理速度は高速化できると考えられる。

(2) **ＳＥＡＬ**

ＳＥＡＬはＩＢＭの Rogaway と（ＤＥＳの開発者の一人）Coppersmith により開発された方式で、米国で特許が成立している。

ＳＥＡＬは 32bit プロセッサに最適化したアルゴリズムを持ち、きわめて高速に動作する（486-50MHz でおよそ 58Mbps）[22]。ＳＥＡＬの特長は、ＳＨＡアルゴリズムを用いて、160bit の鍵 k と 32bit の n を用いて 64kByte 以下の擬似乱数列を生成することである。そのために、ＳＥＡＬでは９×32 個の 64bit S-box を用いるが、そのためのセットアップ時間が 200 個のＳＨＡハッシュ値の計算量に匹敵するため、少量のデータの暗号化には向かないといわれる。

(3) カオス暗号

カオス（Chaos）とは本来「混沌」を意味するギリシャ語であるが、常微分方程式や差分方程式で記述される物理的な統計確率現象の、離散時間における実数値解の不規則な振る舞いを表わすのものとして用いられる。カオスによる乱数生成の方法や性質については 2.5.3 に述べる。

アナログカオスには、特定の条件下で初期条件が異なっていても、十分時間が経過した後の定常状態ではほぼ同一の振る舞いをするという同期現象がある。これを利用したのが一般的な「カオス暗号」であるが、カオスのパラメータそのものが構造的安定性を有しているため推定されやすいという欠点がある。また、アナログ値を近似して用いるため微少な誤差による同期外れなども懸念される。

デジタルカオスは、アナログ回路の常微分方程式や差分方程式を、デジタル計算で解く手法を利用するため、アナログカオスに比べて再現性が高い。近年、デジタルカオスを用いたと称する暗号がいくつか提唱されているが、パラメータの取り方や安全性解析など、明らかにされていないケースが多い。

現時点では、このような暗号では様々な攻撃に対しての安全性が証明されておらず、提唱者、実装者による仕様の公開と、安全性に対する評価が必要であろう。カオス自体は、エルゴード性（任意の初期値から出発した軌道が任意の点の近傍を何度も通過する）を持つため、軌道上の特定の点から初期値を求めることは極めて困難であり、初期値を鍵とする暗号では各種解読法に対する強さが期待できる。

今後、カオスの持つ様々な性質がより深く解析され、実証的にも原理的にも安全な暗号方式が提唱されることを期待したい。

参考文献 [1] J. Kelsey, B. Schneier, and D. Wagner, "Related-Key Cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA," ICICS '97 Proceedings

[2] J. Kelsey, B. Schneier, and D. Wagner, "Key-Schedule Crypta-nalysis of 3-WAY, IDEA, G-DES, RC4, SAFER, and Triple-DES," Advances in Cryptology--CRYPTO '96 Proceedings

[3] 池野信一・小山謙二：「現代暗号理論」第３章 3.4 電子情報通信学会、1986

[4] NIST：FIPS PUB 46-２ DATA ENCRYPTION STANDARD (DES)
URL: http://www.itl.nist.gov/div897/pubs/fip46-2.htm

[5] NIST：FIPS PUB 74 - GUIDELINES FOR IMPLEMENTING AND USING THE NBS DATA ENCRYPTION STANDARD
URL: http://www.itl.nist.gov/div897/pubs/fip74.htm

[6] NIST：FIPS PUB 81 DES MODES OF OPERATION
URL: http://www.itl.nist.gov/div897/pubs/fip81.htm

[7] Ascom Systec 社ホームページ：
URL: http://www.ascom.ch/Web/systec/security/idea.htm

URL: http://bsa.org/policy/encryption/cryptographers.html
[28] NIST AES　ホームページ：
URL: http://csrc.nist.gov/encryption/aes/aes_home.htm
[29] RC4　URL: http://www.cs.hut.fi/crypto/rc4.txt

2.3 公開鍵暗号（非対称暗号）

2.3.1 概　　要

(1) **公開鍵暗号（非対称暗号）とは**

公開鍵暗号（非対称暗号）とは、暗号化と復号に異なる鍵を用いる暗号である。いずれか一方の鍵から他方の鍵が容易に計算できないため、一方の鍵を公開することができるという特徴を有する。他方は秘密に保持する。

(2) **公開鍵暗号の原理**

公開鍵暗号では、利用者Ａが利用者Ｂに暗号通信する場合以下の手順で行う。

[鍵生成]各利用者は、あらかじめ一対の鍵を生成し、一方（公開鍵（Public key)）を公開し、他方（秘密鍵（Private Key)）を秘密に保持する。

[暗号化]利用者Ａは、利用者Ｂの公開鍵を用いて平文を暗号化して利用者Ｂに送信する。

[復号]利用者Ｂは、自分の秘密鍵を用いて受信した暗号文を復号する。

このとき公開鍵暗号は、以下の２条件を満たすように構成される。

１）　暗号化と復号の計算は比較的容易である。

２）　暗号解読（Ｂの秘密鍵を知らない者が、Ｂの公開鍵、暗号文、および暗号アルゴリズムから元の平文を計算すること）は非常に困難である。

公開鍵暗号の重要な応用に電子署名（デジタル署名）がある。電子署名については 3.4 で説明する。

(3) **公開鍵暗号の特徴（長所と短所）**

公開鍵暗号は、共通鍵暗号と比較して以下の長所・短所を有する（表 2.3-1 参照）。

a)　**鍵の秘密配送が不要**

共通鍵暗号では、暗号通信に先だって秘密に共通鍵を配送（共有）する必要があるが、公開鍵暗号では、公開されている相手の公開鍵を入手するだけでよい。

b)　**秘密に保持する鍵が少ない**

ｎ人の利用者と暗号通信する場合、共通鍵暗号では、ｎ個の異なる共通鍵を秘密に保持する必要があるが、公開鍵暗号では、自分の秘密鍵を保持するだけでよい。

c)　**電子署名（デジタル署名）が実現できる**

共通鍵暗号では、ある共通鍵で暗号化された文書がその共通鍵を共有する当事者のいずれによって暗号化されたかが特定できないが、公開鍵暗号では、ある秘密鍵で暗号化された文書はその秘密鍵を保持する当人しか暗号化できないため誰が作成したかが特定できる。

d)　**共通鍵暗号に比べて処理が低速**

公開鍵暗号では、共通鍵暗号に比較して、暗号処理が低速である。このため、多量のデータの暗号通信には共通鍵暗号を利用し、共通鍵暗号に用いられる共通鍵の秘密配送と電子署名に公開鍵暗号を利用するのが一般的である。

e)　**公開鍵の正当性確保が必要**

公開鍵暗号では、公開鍵を公開できる反面、それが偽の公開鍵にすり変えられる

と大きな問題となる。これを防ぐため、公開鍵が誰の公開鍵であるかを証明する機関（認証局(Certification Authority)）を利用するのが一般的である。

f) **暗号通信による相手認証は不可能（匿名暗号通信）**

共通鍵暗号では、当事者しか知らない共通鍵を用いて暗号通信するため、暗号文が正しく復号できるかどうかによって通信相手を認証できる。しかし公開鍵暗号では公開鍵を用いて誰でも暗号通信できるため、受け手は誰から暗号文が送られてきたかが分からない。言い換えれば匿名の暗号通信が可能（なお公開鍵暗号の電子署名機能を用いれば、相手認証が可能である）。

表2.3-1　公開鍵暗号と 共通鍵暗号の 比較

	公開鍵暗号	共通鍵暗号
a)鍵の秘密配送	○不要	×必要
b)秘密に保持すべき鍵	○自分の秘密鍵のみ	×通信相手毎に必要
c)電子署名	○可能	×困難
d)処理速度	×低速	○高速
e)公開鍵の管理	×必要	○不要
f)暗号通信による認証	×困難	○可能

2.3.2 公開鍵暗号の 機能と 用途

公開鍵暗号が提供する機能と用途をまとめると次の通りである。

表2.3-2　公開鍵暗号の 機能と 用途

機能・用途	内容
1)暗号化	正当な受信者以外には分からないようにデータを送る機能
2)電子署名	データ発信者の認証とデータの完全性（改ざんされていないこと）を第3者が検証できるようにする機能
3)相手認証	通信相手を認証する機能（公開鍵暗号の電子署名機能を利用する場合）
4)否認検出	あるデータを送信した事実や受信した事実を後で否認すること検出防止する機能
5)公開鍵証明書発行	公開鍵が誰の公開鍵であるかを証明する公開鍵証明書を発行する機能

2.3.3 公開鍵暗号の 種類

(1) 公開鍵暗号の種類

1976 年に Diffie-Hellman が公開鍵暗号の概念を発表して以来、様々な公開鍵暗号方式が発表された。以下では公開鍵暗号を次の２つの観点から分類する。

①　実現機能による分類

公開鍵暗号は、暗号通信のみ可能な方式、電子署名のみ可能な方式、暗号通信と電子署名の双方が可能な方式に分類できる。

表2.3-3　実現機能による分類

実現機能	方式
暗号通信及び署名通信	ＲＳＡ暗号、Rabin 暗号など
暗号通信のみ	ElGamal 暗号、楕円曲線暗号（楕円曲線上の ElGamal 暗号）など
署名通信のみ	ElGamal 署名、Schnorr 署名、ＤＳＡ署名、楕円ＤＳＡ署名（楕円曲線上のＤＳＡ署名）など

② 安全性の根拠となる数論による分類

公開鍵暗号は、公開鍵から秘密鍵を計算すること（暗号解読）の難しさが、古くから研究されている素因数分解問題や離散対数問題といった数学問題を解くことの難しさに対応している。主な公開鍵暗号をそれらが基づく数論よって分類すると次のようになる。

表2.3-4　安全性の根拠となる数論による分類

安全性の根拠とする数論		方式
素因数分解問題	n(=pq)から素因数 p,q を求める問題	RSA 暗号、Rabin 暗号、Williams 暗号、逆数暗号、楕円ＲＳＡ暗号など
有限体上の離散対数問題	$Y(=g^x \bmod p),g,p$（ここで p は素数または素数のべき、g は有限体 Fp の原始根）から x を求める問題	ElGamal 暗号、ElGamal 署名、ＤＳＡ署名、Schnorr 署名など
楕円曲線上の離散対数問題	Y(=x＊G),G（ここで Y,G は、有限体 Fp 上の楕円曲線 E/Fp 上の点）から、x を求める問題	楕円暗号（楕円曲線上の ElGamal 暗号）、楕円ＤＳＡ署名（楕円曲線上のＤＳＡ署名）など

2.3.4 代表的な公開鍵暗号方式

以下では、代表的な公開鍵暗号方式として、ＲＳＡ暗号、ElGamal 暗号、楕円曲線暗号の構成について具体的に示す。なお、詳細については、それぞれ参考文献を参照されたい。

(1) ＲＳＡ暗号の構成[3]

[鍵生成]

各利用者 は、それぞれ、$n_i=p_iq_I$（ここで p_i,q_i は素数）、$e_id_i=1 \pmod{LCM(p_i-1,q_i-1)}$を満たす e_i,d_i,n_i,p_i,q_i を求め、(e_i,n_i) を公開鍵として公開し、(d_i) を秘密鍵として秘密に保持する[1]。ここでＬＣＭ(a,b) はａとｂの最小公倍数を意味する。

[暗号化]

利用者Ａは利用者Ｂの公開鍵 (e_B,n_B) を用いて、平文Ｍ（$<n_B$）に対する暗号文Ｃを以下の式で求める。$C=M^{eB} \pmod{n_B}$

---

[1]秘密鍵として(d1,d2,p,q)（ここで d1=d mod(p-1)、d2=d mod(q-1)）を用いて、復号処理を高速に行う方法が Qusiquater らによって提案されている[ ]。

[復号]

利用者Bは自分の秘密鍵（d_B）と n_B を用いて、暗号文Cに対する復号文Mを以下の式で求める。$M=C^{dB}(\mathrm{mod}\ n_B)$

(2) ElGamal 暗号の構成［3］

[鍵生成]

- システムパラメータ生成

  システム管理者は、素数 p、有限体 F_p の原始根gを生成し、(p、g)を全利用者共通のシステムパラメータとして公開する（より一般には p、g を各利用者毎に異ならせる構成も可能である）。

- 各利用者の鍵生成

  各利用者 $_i$ は、システムパラメータ p,g を用いて、p-1 未満の乱数 x_i と、$y_i = g^{x_i}(\mathrm{mod}p)$を生成し、$y_i$ を公開鍵として公開し、x_i を秘密鍵として秘密に保持する。

[暗号化]

利用者Aはシステムパラメータ p、g、利用者Bの公開鍵 Y_B を用いて、平文 M(<p) に対する暗号文（C１、C２）を以下の式で求める。$C1=g^k(\mathrm{mod}p)$、$C2=M*y_B{}^k(\mathrm{mod}\ p)$（ここで k は乱数）。

[復号]

利用者Bは自分の秘密鍵（x_B）とシステムパラメータ g,p を用いて、暗号文（C１、C２）に対する復号文Mを以下の式で求める。$M=C2/C1^{\times B}(\mathrm{mod}p)$

(3) 楕円曲線暗号の構成［4］［9］［10］

[鍵生成]

- システムパラメータ生成

  システム管理者は、有限体 Fp 上の楕円曲線E／Fp、楕円曲線E／Fp 上の位数 q の元（ベース点）Gを生成し、全利用者共通のシステムパラメータとして公開する（より一般には、楕円曲線E／Fp、Gを各利用者毎に異ならせる構成も可能である）。

- 各利用者の鍵生成

  各利用者 は、それぞれ、q-1 未満の乱数 x_i、及び楕円曲線E／Fp 上で、$Y_i = x_i \cdot G$（Y、G∈ E／Fp）を求め、Y を公開鍵として公開し、xを 秘密鍵として秘密に保持する。

[暗号化]

利用者Aは、システムパラメータG、及び利用者Bの公開鍵 Y_B を利用して平文mに対する暗号文（C１、C２）を以下の式により求める。

$C1=k\cdot G$、$C2=(m*\{k\cdot Y_B\}x)\ (\mathrm{mod}p)$、ここで k は乱数、{A}x は楕円曲線上の点Aの x 座標。

[復号]

利用者Bは、秘密鍵 x_B を用いて、暗号文（C１、C２）に対する復号文 m を以下の式により求める。$m=(C2/\{x_B\cdot C1\}x)\ (\mathrm{mod}p)$

2.3.5 公開鍵暗号の 攻撃方法［8］

(1) 公開鍵暗号の攻撃の種類と解読のレベル

暗号を解読するための計算量は、一般に、攻撃者が利用できる攻撃の種類や、解読のレベルによって変化する。以下に、公開鍵暗号の攻撃の種類と解読のレベルについて示す。

表2.3-5　攻撃の 種類

攻撃の種類	内容
1)直接攻撃	公開鍵から直接秘密鍵を求める攻撃
2)選択平文攻撃	攻撃者の選んだ平文に対する暗号文を用いる攻撃
3)選択暗号文攻撃	攻撃者の選んだ暗号文に対する平文を用いる攻撃
4)適応的選択暗号文攻撃	選択暗号文攻撃において、この攻撃で前に用いた平文から、この攻撃で次に用いる暗号文を選択できる場合

表2.3-6　解読の レベル

ｱ)全面的解読	ある公開鍵に対する秘密鍵を求めることができる場合
ｲ)一般的解読	任意の暗号文に対する平文を求めることができる場合
ｳ)部分的解読	平文の一部を求めることができる場合

このうち、1)及び 2)は、誰でも行える攻撃であるので、これらに対して十分な安全性を有していなければならない。すなわちこれらの攻撃に対して上記解読ｱ)、ｲ)、ｳ)が成立しないようにしなければならない。3)、4)は一般的には成立し難いため、仮にこれらの攻撃に対して安全でないとしても、実用上問題ないケースがある。このように、公開鍵暗号を使用する場合は、想定する攻撃の種類と解読レベルに応じて適切な公開鍵暗号を利用する必要がある。

(2) 直接攻撃に対する安全性

公開鍵暗号では、直接攻撃（公開鍵から直接秘密鍵を計算する攻撃）の難しさが、素因数分解問題や離散対数問題といった数学問題を解く難しさに対応している。以下では、公開鍵暗号に利用される代表的な数学問題に対してこれまでに知られている解法についてまとめる。以下では計算量の評価に $Ln[a,b]=\exp((b+o(1))(\log p)^{a}(\log\log p)^{1-a})$を用いる。 $o(1)$は極限値が０の定数。

① 素因数分解問題の解法

素因数分解問題の主な解法を表 2.3-7 に示す。表において１～４は解読計算量が素因数の性質に依存する解法であり、５～６は解読計算量が合成数 n のサイズのみによって決まる解法である。表からｎを２素数 p、q の積とするとき、$(p\pm1)$、$(q\pm1)$がそれぞれ大きな素因数をもちかつ $|p-q|$ が大きい場合、現時点で最強の解法は、数体ふるい法でありその場合の計算量は以下の通りである。

$$Ln[1/3,1.922]=\exp((1.922+o(1))(\log p)^{1/3}(\log\log p)^{2/3})$$

表2.3-7　素因数分解問題の 主な 解法

解法	内容	計算量
1)試行割算法	合成数ｎを小さな素数から順に割っていく方法	√n
2)p-1 法、p+1 法	合成数 n の素因数 p に対して、それぞれ(p-1)、(p+1)が小さい素因数の積に分解されるとき有効な方法	p-1 が B-smooth の場合 Blnｎ/lnB(ln は自然対数)
3)Fermat 法	合成数 n の素因数 p、q に対して|p-q|が小さい場合に有効な方法	
4)楕円曲線法	Zp 上の楕円曲線 E/Zp の位数(楕円曲線の点(元)の個数)が小さな素数の積に分解できるとき有効な方法	Ln[1/2,1.02]
5)２次ふるい法	２次合同式 $k^2=l^2$(modn)を満たす k と l を求め、GCD(k±l,n)より n の素因数を求める方法	Ln[1/2,1.02]
6)数体ふるい法	２次ふるい法を一般化した方法	Ln[1/3,1.922]

② 有限体上の離散対数問題（ＤＬＰ）に対する解法

有限体上の離散対数問題に対する主な解法を表 2.3-8 に示す。

表から(p-1)が大きな素因数をもつ場合、離散対数問題に対する現時点で最強の解法は数体ふるい法でありその場合の計算量は以下の通りである。

$Ln[1/3,1.922]= \exp((1.922+o(1))(\log p)^{1/3}(\log\log p)^{2/3})$

表2.3-8　有限体上の 離散対数問題の 主な 解法

解法	内容	計算量
1)Pohig-Hellman 法	$y=g^x$(modp)において g の位数（ｇが原始根の場合(p-1)）が小さな素数の積に分解されるときに有効な方法	√p
2)Index Caliculus 法 (指数計算法)	ある素数の集合に対する対数をまず求め、それを利用して所望の離散対数を求める方法	Ln[1/2,1.4]
3)数体ふるい法	Index Caliculus(指数計算)法の一種で、素因数分解問題に対する最強の解読方法である数体ふるい法を離散対数問題に応用した方法	Ln[1/3,1.922]

③ 楕円曲線上の離散対数問題（ＥＤＬＰ）に対する解法

主な楕円曲線上の離散対数問題に対する解法を表 2.3-9 に示す。

MOV-Reduction 法、および、Smart および佐藤－荒木の攻撃法の適用を受けない楕円曲線を利用する場合、楕円曲線上の離散対数問題に対する現時点での最強の解読法は、Pohig-Hellman 法であり、その場合の計算量は√p となる。このことから、楕円曲線に基づく暗号・署名方式では、従来の素因数分解問題や有限体上の離散対数問題に基づく暗号・署名方式に比べて鍵サイズを小さくでき、処理を高速に行えると見られている。

表2.3-9　楕円曲線上の 離散対数問題の 主な 解法

解法	内容	計算量
1)Pohig-Hellman 法	楕円曲線上のベース点 G の位数が小さな素数の積に分解されるときに有効な方法。	√p
2)MOV-reduction 法	Trace=0 の楕円曲線(supersingular)に対して有効な方法。楕円曲線 E/Fp 上の離散対数問題を、有限体上の離散対数問題に帰着させる解法であり、楕円曲線上の離散対数問題が、Fp の小さな拡大体上の離散対数問題に帰着できる。	離散対数問題に対する解法が適用できる
3)Smart および佐藤-荒木の攻撃	Trace=1 の楕円曲線(anomalous)に対して有効な方法。素数 p を法とする有限体 Fp 上で構成される楕円曲線 E/Fp においてその楕円曲線の点の個数がちょうど p である場合に有効である。	o(logp)

2.3.6 主な 公開鍵暗号に 対する 解法

ここでは主な公開鍵暗号について、上記以外の攻撃方法について説明する。

(1) ＲＳＡ暗号の安全性

①　直接攻撃

ＲＳＡ暗号において公開鍵ｎから秘密鍵ｄを求める攻撃法として、ｎを素因数分解して p、q を求めた後に秘密鍵ｄを求める方法より効率的な方法は今のところ知られていない。したがって、ＲＳＡ暗号の直接攻撃に対する安全性は、素因数分解問題に帰着されると考えてよい。

②　既知平文攻撃

平文Ｍとそれに対応する暗号文Ｃを入手し、$M=C^d(\mathrm{mod}n)$から秘密鍵ｄを求める攻撃。合成数ｎを法とする離散対数問題に帰着される（この問題は素因数分解問題より困難と考えられている）

③　短周期攻撃

暗号文Ｃに対して、f（Ｃ）＝$C^e(\mathrm{mod}n)$とするとき、f^t（Ｃ）＝Ｃが成り立つとき$M= f^{t-1}$（Ｃ）が平文となる。この攻撃に対しては、(p±1)、(q±1)の最大素因数をそれぞれ、p_+、p_-、q_+、q_-とするとき(p_+±1)、(p_-±1)、(q_+±1)(q_-±1)がそれぞれ大きな素因数を持つとき安全である。

④　共通　攻撃

各利用者が共通のｎを利用するとき、一般的解読（秘密鍵を知ることなく任意の暗号文に対する平文を解読ずること）が可能となる。この攻撃を避けるため、各利用者はそれぞれ異なるｎを用いる必要がある。

⑤　小共通　攻撃

各利用者が共通の e を利用するとき一般的解読が可能となる。この攻撃を避けるために e は、32 以上とする。

(2) ElGamal　暗号に対する攻撃方法

①　直接攻撃

ElGamal 暗号において、公開鍵から秘密鍵を求める問題はそれぞれ有限体上の

離散対数問題に帰着される。

② 共通乱数ｋ攻撃

同一の乱数ｋで暗号化した場合は、一対の平文と暗号文から同一の乱数で暗号化された平文は解読できる。この攻撃を避けるため乱数 k は毎回変える必要がある。

(3) 楕円曲線暗号に対する攻撃方法

① 直接攻撃

楕円曲線暗号において、公開鍵から秘密鍵を求める問題はそれぞれ楕円曲線上の離散対数問題に帰着される。

② 共通乱数ｋ攻撃

同一の乱数ｋで暗号化した場合は、一対の平文と暗号文から同一の乱数で暗号化された平文は解読できる。この攻撃を避けるため乱数 k は毎回変える必要がある。

2.3.7 公開鍵暗号の 安全性と 鍵サイズについて

(1) 鍵サイズと解読計算量について

ＲＳＡ暗号、ElGamal 暗号および楕円暗号における鍵サイズと解読計算量の関係を表 2.3-10 に示す[2]。

ここでは、ＲＳＡ暗号、ElGamal 暗号、および楕円暗号に対して、現在知られている最良の解法を適用（2.3.5(2)参照）した場合を仮定している。

表2.3-10 鍵サイズと 解読計算量の 関係

ＲＳＡ暗号 (ビット)	ElGamal 暗号 (ビット)	楕円暗号 (ビット)	解読計算量 (MIPS*YEAR)
５１２	５１２	１００	$3*10^4$
７６８	７６８	１２８	$2*10^8$
１０２４	１０２４	１６０	$3*10^{11}$
２０４８	２０４８	２２４	10^{21}

ここでＭＩＰＳ＊ＹＥＡＲは、１ＭＩＰＳマシン（１秒に 10^6 回演算を行う能力を有するマシン）を１年間動かし続けて得られる計算量であり $3.15*10^{13}$ 回の演算に相当する。

(2) 安全な鍵サイズについて

ＲＳＡ暗号、ElGamal 暗号および楕円暗号について、現在知られている最良の解法を適用した場合に、現在から 2022 年までの各時点において、鍵サイズと、暗号解読者の費やす解読コストと、解読時間の関係について検討した結果を表 2.3-11～2.3-14 に示す。

なおここでは、次の仮定をおいている。

- 現在(1997 年)100ＭＩＰＳのパソコンが約 20 万円で購入できるものとする。

[2] ＲＳＡ暗号の 512、768、1,024bit における計算量の 見積もり は 文献 [12] に 基づく。 RSA暗号の 2,024bit、および、ElGamal 暗号、楕円暗号については 独自に 見積った結果である。

● 1.5 年でＣＰＵパワーが２倍（５年で 10 倍）になるものとする。

このとき、次のようにして解読時間を見積ることができる。例として、解読にかける予算を 100 万円とする。このとき 100ＭＩＰＳパソコン５台分の計算パワー（500ＭＩＰＳ）を手にすることができる。このほか 512bitＲＳＡの解読時間は

$$3 \times K10^{4}\text{ＭＩＰＳ年} / 500\text{ＭＩＰＳ} = 60\text{年}$$

となる。また 5 年後の 2002 年には計算機パワーが 10 倍になるから 512bitＲＳＡの解読時間は 6 年となる（表 2.3-11）。

表より、ＲＳＡ 512bit 相当の場合、企業レベルの攻撃者を仮定すると、現時点で既に数週間で解読できることがわかる（☆）。また、今後 15 年程度で、個人レベルでも、数週間で解読できるようになることがわかる（☆☆）。

また、ＲＳＡ 1,024bit 相当の場合、企業レベルの攻撃者を仮定しても、今後 20 年間は、解読が困難であることがわかる（☆☆☆）。

また、ＲＳＡ 2,048bit 相当の場合、国家レベルの攻撃者を仮定しても、今後 20 年以上、解読が困難であることがわかる。

表2.3-11　512bit　RSA相当の 場合の 解読時間

攻撃者	個　人	企　業	国　家
予　　算	100 万円	10 億円	1 兆円
1997 年	60 年	22 日（☆）	32 分
2002 年	6 年	2.2 日	3.2 分
2007 年	210 日	5.3 時間	19 秒
2012 年	21 日（☆☆）	32 分	1.9 秒
2017 年	2.1 日	3.2 分	0.19 秒
2022 年	5.1 時間	19 秒	0.019 秒

表2.3-12　768bit　RSA相当の 場合の 解読時間

攻撃者	個　人	企　業	国　家
予　算	100 万円	10 億円	1 兆円
1997 年	$4*10^{5}$ 年	$4*10^{2}$ 年	150 日
2002 年	$4*10^{4}$ 年	40 年	15 日
2007 年	$4*10^{3}$ 年	4 年	1.5 日
2012 年	$4*10^{2}$ 年	150 日	3.5 時間
2017 年	40 年	15 日	21 分
2022 年	4 年	1.5 日	2.1 分

表2-13　1,024bit　RSA相当の 場合の 解読時間

攻撃者	個　人	企　業	国　家
予　　算	100 万円	10 億円	1 兆円
1997 年	$6*10^{8}$ 年	$6*10^{5}$ 年	$6*10^{2}$ 年
2002 年	$6*10^{7}$ 年	$6*10^{4}$ 年	60 年
2007 年	$6*10^{6}$ 年	$6*10^{3}$ 年	6 年
2012 年	$6*10^{5}$ 年	$6*10^{2}$ 年	220 日
2017 年	$6*10^{4}$ 年	60 年（☆☆☆）	22 日
2022 年	$6*10^{3}$ 年	6 年	2.2 日

表2.3-14　2,048bit　RSA相当の 場合の 解読時間

攻撃者	個　人	企　業	国　家
予　　算	100万円	10億円	1兆円
1997年	$2*10^{18}$年	$2*10^{15}$年	$2*10^{12}$年
2002年	$2*10^{17}$年	$2*10^{14}$年	$2*10^{11}$年
2007年	$2*10^{16}$年	$2*10^{13}$年	$2*10^{10}$年
2012年	$2*10^{15}$年	$2*10^{12}$年	$2*10^{9}$年
2017年	$2*10^{14}$年	$2*10^{11}$年	$2*10^{8}$年
2022年	$2*10^{13}$年	$2*10^{10}$年	$2*10^{7}$年

2.3.8 主な 公開鍵暗号の 性能比較

ここでは、ＲＳＡ暗号、ElGamal 暗号、および楕円曲線暗号（楕円 ElGamal 暗号）について同じ安全性における性能比較を行う（1,024bit のＲＳＡ暗号を基準とする）。

(1) 鍵サイズ

ＲＳＡ暗号、ElGamal 暗号、および楕円暗号の鍵サイズを表 2.3-15 に示す。

表2.3-15　鍵サイズ

	秘密鍵(bit)	公開鍵(bit)
ＲＳＡ暗号	1024[3]、2048[4]	1024[5]＋［ｅ］[6]
ElGamal 暗号	1024	1024
楕円曲線暗号	160	161　（圧縮時）[7] 320　(非圧縮時)

(2) 処理量

ＲＳＡ暗号、ElGamal 暗号、および楕円暗号の処理量を表 2.3-16 に示す。

ここでは、以下に示す一般によく利用されているケースについて比較する。

① ＲＳＡ暗号および ElGamal 暗号における指数演算には通常の binary 法を利用

② 楕円暗号における楕円点のｎ倍演算には、addition-subtraction 法を利用

③ ＲＳＡ暗号において公開鍵ｅのサイズは 20bit とする。

また復号処理および署名処理は Quisquater 法を利用するものとする。

表2.3-16　処理量　(512bit の 乗算剰余演算[8]の 回数に 換算(概数))

---

[3] ＲＳＡ暗号の 公開鍵 eのビットサイズ

[4] Quisquater 法利用時

[5] ＲＳＡ暗号の 秘密鍵 dのビットサイズ

[6] eの ビットサイズ 32bit 程度に 選ばれることが 多い ）

[7] 楕円暗号の 公開鍵 Yを 圧縮変換 (x 座標と 符号ビット(1bit))したときのビットサイズ (x 座標と 符号ビットから y座標を 求めることができる ）

	暗号化	復　号	署　名	検　証
ＲＳＡ暗号	180	1552	1552	180
ElGamal 暗号	12292	6188	6188	7208
楕円暗号	424	221	205	253

2.3.9 公開鍵暗号使用時の 一般的な 留意点

公開鍵暗号使用時の一般的な留意点について以下に示す。

(1) **認証局の必要性**

公開鍵暗号では、入手した公開鍵が所望する相手の正しい公開鍵かどうか確認する仕組みが必要である。さもなければ、第３者が仕掛けた偽の公開鍵を用いて重要な秘密文書を送ってしまうからである。

これを防ぐため、ＥＣシステムなど不特定多数の利用者が利用するようなシステムでは、公開鍵が誰の公開鍵であるかを証明する機関（認証局(Certification Authority)と呼ばれる）を利用するのが一般的である。

(2) **故障に基づく攻撃**

最近、故障に基づく攻撃が提案された。これは、暗号装置に故意に誤動作を起こさせ（たとえば秘密鍵の特定ビットを１bit 反転させ）、元の正しい結果と、誤った結果に基づいて秘密鍵を解読しようという攻撃である。対策として、暗号装置に誤動作が生じたことを検出して出力しないようにするか、暗号装置を厳重に管理する必要があると考えられる。

2.3.10 個々 の 暗号方式使用時の 留意事項

(1) **ＲＳＡ暗号使用時の留意事項**

- 処理を高速にするため、利用者全員が、共通の n(=pq)を利用する場合は、解読される危険があるため推奨できない。
- 処理を高速にするため、公開鍵 e を小さく（例えば３）する場合は、解読される危険があるため推奨できない。e は、32 より大きくすることが推奨する。
- 素数 p、q の生成においては、素因数分解等の攻撃を困難にするため、以下の条件を満たす素数を利用することを推奨する。このような素数の求め方については例えば[3]参照。
  - ① p-1 は大きな素因数ｒを含む
  - ② p+1 は大きな素因数ｓを含む
  - ③ r-1 は大きな素因数ｔを含む
- ＲＳＡ暗号を利用して、ＤＥＳ鍵を暗号化配布する場合や、ハッシュ値に対して署名を施す場合など、入力データがＲＳＡ暗号の処理サイズに満たない場合は、安全性を高めるため、適切な方法で padding 処理する必要がある[11]。

(2) ElGamal　**暗号使用時の留意事項**

- 同一の乱数 k を用いて異なる平文に対する暗号文を作成する場合は、解読される危

---

[8] ａ＊ｂmod ｎ(a,b,n は 512bit) ：ａと ｂの 積に 対して、 ｎで 割ったときの 余り を 求める 演算

険があるため乱数 k は毎回変える

- 素数 p の生成においては、離散対数問題などの解読を困難にするため p-1 は大きな素因数を持つように選ぶ。

(3) **楕円暗号使用時の留意事項**

- 同一の乱数 k を用いて異なる平文に対する暗号文を作成する場合は、解読される危険があるため乱数 k は毎回変える
- 楕円暗号を構成する際は以下の条件を満たす必要がある。
  ① 楕円曲線上の離散対数問題の解読を困難にするため、楕円曲線のベース点の位数が、大きな素因数をもつようにする。
  ② ＭＯＶの解法が成り立たないように楕円曲線を選ぶ。
  ③ Smart と佐藤－荒木の解法が成り立たないように楕円曲線を選ぶ。

  楕円暗号の構成法の詳細については例えば[4]を参照。
- 楕円暗号を利用して、ＤＥＳ鍵を暗号化配布する場合など、入力データが楕円暗号の処理サイズに満たない場合は、安全性を高めるため、適切な方法で padding 処理する必要がある。

2.3.11 今後の 動向

現在商用にはＲＳＡ暗号が広く利用されているが、最近楕円曲線暗号が国内外のいくつかの企業によって実装／実用化されている。また国際標準化機構ＩＳＯ／ＳＣ27［5］やIEEEp1363［6］において、楕円曲線を利用する方式の標準化が進められており、今後はＲＳＡに加えて利用されていくものと思われる。

このとき、複数暗号方式の併用による相互運用性の問題が生じるが、この点については、送受信間で利用可能な暗号方式を選択しネゴシエートするプロトコルにより解決されていくものと思われる。

また、最近、従来の公開鍵暗号とは異なる数学的問題に安全性の根拠をおく新しい公開鍵暗号が提案された。これはユークリッド空間の格子に関する数学的難問に安全性の根拠をおく方式であり理論的に興味深いが、１bit 毎に暗号化するため非常に効率が悪く、現時点では実用化レベルに達していない[7]。

参考文献：[1] J.J.Quisquater and C.Couvereur,”First Decipherment Algorithm for RSA Public Key Cryptosystem”, Electronic Letters, v.18,1982, pp155-168

[2] “Security estimate for 512bit-RSA”,RSA Lab、1995.
URL: http://www.rsa.com/rsalabs/html/tech_notes.html

[3] 岡本栄司著：「暗号理論入門」、共立出版、1993

[4] 岡本龍明、山本博資共著：「現代暗号」、産業図書、1997
ISO/SC27/DIS17770-3、“Key Management –Part3 Mechanisms using asymmetric Techniques”

[5] ISO/SC27/CD14888-3 、 ”Digital Signature with appendix –Part3

Certificate Mechanisms”URL: http://www.iso.ch:8080/jtc1/sc27/

[6] IEEEp1363 Working Draft、URL: http://stdsbbs.ieee.org/groups/1363

[7] 下山武司：「Ajtai, Dwork による Lattice Based Public-Key Cryptosystem の解説」、第 1 回代数曲線とその応用シンポジウム、1997.9.1

[8] 楠田浩二、櫻井幸一：「公開鍵暗号方式の安全性評価に関する現状と課題」、Discussion Paper No.97-J-11、日本銀行金融研究所、1997.7.
URL: http://www.imes.boj.or.jp/idps/97-J-11.htm
URL: http://www.imes.boj.or.jp/idps/97-J-11.pdf

[9] 岡本龍明、太田和夫編：「暗号、ゼロ知識証明、数論」、共立出版、1995.

[10] 宮地充子：「楕円曲線理論の応用数理における開花」、応用数理学会誌、1997.9.

[11] PKCS#1:RSA Encryption Standard、RSA Lab、1993.11.

2.4 ハッシュ関数

2.4.1 ハッシュ関数とその要件

ハッシュ関数（Hush Function）は、パスワード認証・デジタル署名・メッセージ認証などの目的で幅広く用いられ、現代暗号の重要な分野を構成している。一般的には、ハッシュ関数とは、ある大きな領域 D からより小さな領域 R への多対一のマッピングを行う関数 h であり、

$$h : D \rightarrow R \text{ かつ } |D| > |R|$$

$|x|$ は領域の大きさ

とかける。

特に、現代暗号におけるディジタル署名などではハッシュ関数としては、一方向性ハッシュ関数が重要となる。一方向性ハッシュ関数とは、関数値 y から $h(x)=y$ となるような元の値 x を求めることが「計算量的に困難な」ハッシュ関数である。

［一方向性の定義］

以下の条件を満足するとき、関数 f を一方向性と呼ぶ。

(1) {f の容易性} x を入力して f(x)を出力する多項式時間アルゴリズムが存在する。

(2) {f の困難性} 全ての（確率的）多項式時間アルゴリズム A に対して、以下が成立する。

$$\mathbf{Pr}\left[A(f(x)) \in f^{-1}(f(x))\right] < \varepsilon$$

デジタル署名に必要な一方向性ハッシュ関数としては、理論的には、Naor と Yung により定義された汎用一方向性ハッシュ関数と Damgrad により定義された衝突困難ハッシュ関数がある。汎用一方向性ハッシュ関数とは、関数 h とその定義域内にある x が与えられたとき、$h(x)=h(z)$ となる z を求めることが困難であるような関数であり、衝突困難ハッシュ関数とは関数 h が与えられたとき $h(x)=h(z)$ となる一対の値(x, z)を求めることが困難な関数といえる。

この２つのハッシュ関数は一定の仮定の下で、安全性が証明されている点で重要であるが、現実の世界ではハッシュ関数の出力である $y=h(x)$ は固定された有限のビット長であることが必要であり、計算時間も有限の長さであることが要求されている。そこで、実際の「実用的」ハッシュ関数は非衝突一致性（Collision Resistance）を目標に設計されている。この非衝突一致性とは、

$$M \neq M' \quad \text{かつ} \quad h(M)=h(M')$$

なる関係を満たす異なるメッセージ M, M'を見つけることが困難であることと定義される。この性質は、正しいメッセージ M から偽のメッセージ M'を作り出すことが難しいことを意味している。

このような非衝突一致性をもつ実用的なハッシュ関数の構成法としては、

① 既存のブロック暗号を連鎖（ＣＢＣ）モード的に用いて実現する方法

② 専用のハッシュ関数を独自に設計する方法

③ 法剰余演算を利用する方法

がある。

2.4.2 ハッシュ関数への攻撃

ハッシュ関数の安全性（非衝突一致性）の検証を行う上で、共通鍵暗号の安全性の検証と同様、「攻撃」の立場から検証を行う必要がある。

ハッシュ関数に対する攻撃法としては、(1)Birthday attack (2)Pseudo-collision and compression function attack (3)Chaining attack などの攻撃法がある。以下に、最も重要な攻撃法である Birthday attack の概略を示す。

Birthday attack は関数に依存しない攻撃法であり、この攻撃法に対する安全性は、ハッシュ関数の出力 $h(M)$ の長さに依存する。一般にハッシュ関数の出力の長さが、N bit であるとき、おおよそ $2^{N/2}$ 個程度の M' に対してハッシュ関数を計算することで、$h(M) = h(M')$ なる M' を見いだすことができることが知られている。この攻撃法の由来は、ランダムな 23 人が集まったとき、確率１／２で同一の誕生日の人間が２人以上存在するという直感に反する事実（Birthday attack）に由来している。

この Birthday attack に対処するためにハッシュ関数の出力 $h(M)$ はある程度長くなければならず、現在実用化されているハッシュ関数の出力 $h(M)$ の長さは、128bit から 196bit 程度がふつうである。

2.4.3 ハッシュ関数の種類と用途

一般に、ハッシュ関数はその用途に応じて、Keyed Hush 関数と Unkeyed Hush 関数を使い分けています（図 2.4-1 参照）。改ざんなどの不正を検出する場合には、Unkeyed Hush 関数を使用するのが一般的であり、メッセージの作成者の確認などのメッセージ認証のような場合には、秘密通信の場合の共通鍵暗号と同様に、Keyed Hush 関数を使用するのが一般的です。

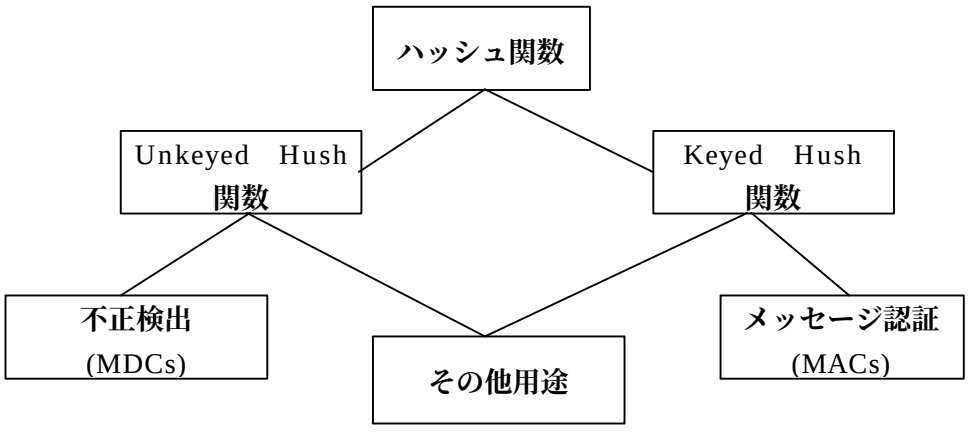


図2.4-1 ハッシュ関数の種類と用途

2.4.4 メッセージダイジェストの意味と利用法

メッセージダイジェストとは、大きな文章に電子署名を施すとき、ハッシュ関数を用い、元になる長いメッセージや文書、データファイルといった可変長の入力で受けたメッセージに対しハッシュ関数を用い、固定長の文字列で返す、コンピュータ処理手法のことである。

この固定長の文字列をメッセージダイジェストという。これは、長いメッセージや文書をコンピュータ処理することによって、簡潔なダイジェストとして表すという考え方で、元の入力情報の「デジタル指紋」ととらえることもできる。

メッセージダイジェストを用いる場合には、使用する通信路の安全性やメッセージ自体の重要性を考慮してどのような利用形態を選択するか決定する必要がある。その代表的な場合を図 2.4-2 に示す。図 2.4-2(a)はメッセージ自体は公になっても問題が少ないが、改ざんされると問題が生じるような情報（例えば、電子商取引での発注数量など）を扱う場合の例であり、この場合にはメッセージ認証のためにハッシュ関数を利用している。図 2.4-2(b)は、メッセージ自身も公になると問題が生じるような情報（例えば、電子商取引でのクレジットカード番号など）を扱う場合であり、メッセージダイジェストを付加した後に暗号化を行い、送信する情報全体を保護している。図 2.4-2(c)は扱う情報は容量の関係で、安全でない通信路（例えば、internet）で送らざるを得ないが、通信路容量に制限があるが安全な通信路（例えば、ダイジェストのみをハンドキャリーする様な場合）の使用が可能な場合である。この場合には、メッセージ自体の保護は別として、メッセージダイジェストのみを安全な通信路で送る方式である。この場合には、Keyed Hush 関数を用いなくても十分に安全性が確保できる。

2.4.5 各種メッセージダイジェスト 用ハッシュ 関数

(1) ＭＤ２、ＭＤ４、ＭＤ５

ＭＤ２、ＭＤ４、ＭＤ５（MDはメッセージダイジェストの略）は広く使用されているハッシュ関数で、Ron Rivest が暗号化専用に設計したものである。これらは、128bit のダイジェストで、徹底的に検索を行う以外に解読する方法はない。

処理速度は、MDが最も遅く、MDが最速である。ＭＤ５は Rivest が「安全ベルト付きＭＤ４」と称したもので、ＭＤ５がＭＤ４よりも保守的な設計で安全性は高いものの、速度の面ではＭＤ４よりも約 33%遅くなっている。3 つのアルゴリズムの中ではＭＤ５が最もよく使用されている。ＭＤ４とＭＤ５の使用に制限はなく、だれでも利用できる。ＭＤ２はＰＥＭとともに使用される。ＭＤ２、ＭＤ４、ＭＤ５の各詳細は、サンプルのＣコードプログラムとともにインターネットＲＦＣ（Requests For Comments）の 1319、1320、1321 でそれぞれ入手できる。

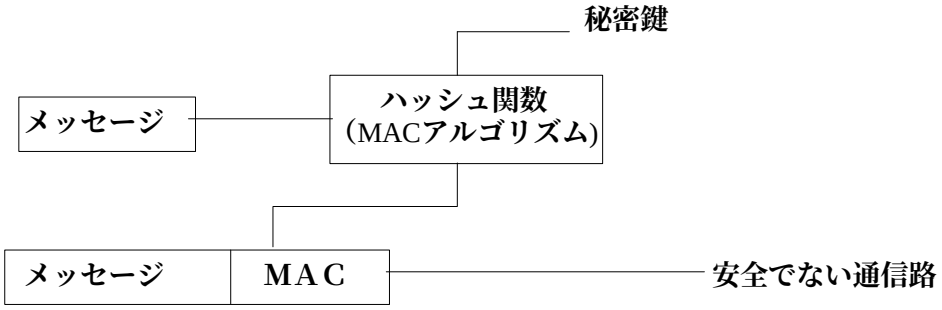
最近の理論研究で興味深い構造特性が分かってきているが、MDアルゴリズムに対する不正な侵入行為は発見されていない。

(2) SHA

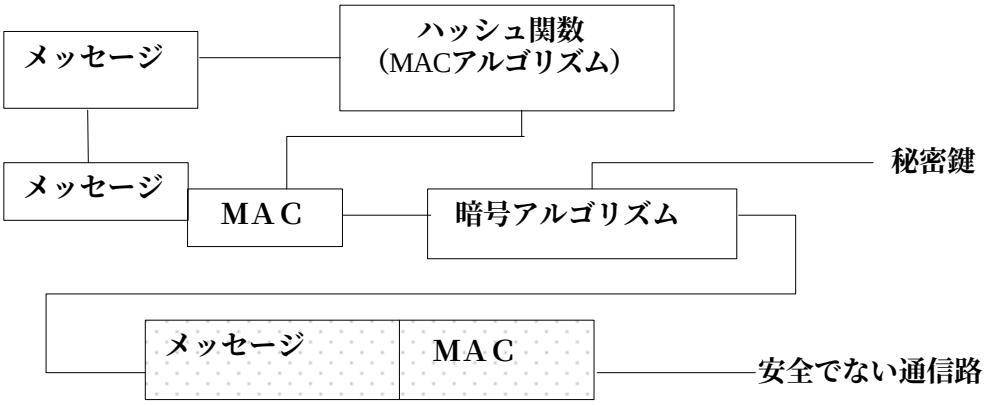
ＳＨＡは、米国商務省に所属する標準化機関であるＮＩＳＴが規定したＳＨＳ（Secure Hash Standard）として定めたハッシュ関数で、米国政府の規格（federal information processing standard)として採用された。このＳＨＡを改訂したハッシュ関数がＳＨＡ－１である。ＳＨＡは、デジタル署名規格ＤＳＳとともに使用するよう設計されており、政府の Capstone プロジェクトの一環ともなる標準です。ＳＨＡにより、2^{64}bit 以下のさまざまな長さの入力から 160bit のハッシュ値が作成される。ＳＨＡの構造はＭＤ４やＭＤ５とよく似ているが、作成されるメッセージダイジェストはMD関数のものより 25%長く、速度は約 25%遅い。安全性に関してはＭＤ５よ

り高いと言われている。現在、Capstone プロジェクトの中ではＳＨＡだけが正式に政府の規格として採用されている。具体的なアルゴリズムの詳細は、参考文献を参照されたい。

(a)　ＭＡＣだけのデータ完全性保証方式



(b)　暗号化を併用したデータ完全性保証方式



暗号化されたメッセージとＭＡＣ

(c)　安全な通信路を仮定したデータ完全性保証方式

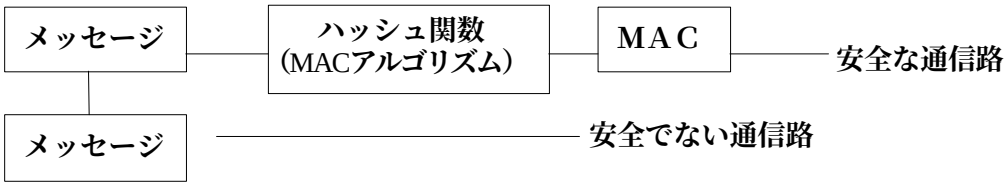


図2.4-2　メッセージダイジェストの 種類と 用途

2.4.6 ブロック 暗号を 利用したハッシュ 関数

暗号化鍵Kおよび（固定長）メッセージMを持つブロック暗号を $B_K(M)$とし、関数Hを

$$H(M)=B_K(M) \blacksquare M$$

と定義するとき、ブロック暗号 $B_K(M)$が安全ならば、H(M)は非衝突一致性を持つと考えられている。

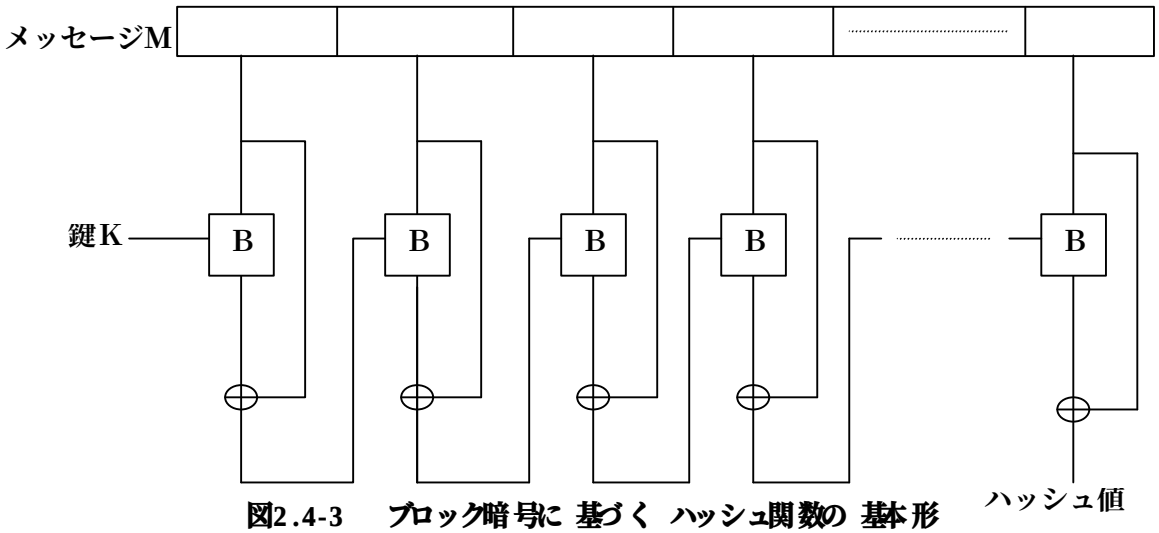


図2.4-3　ブロック暗号に 基づく ハッシュ関数の 基本 形

この例では、メッセージMの長さは、基本となるブロック暗号Bが規定しているブロック長の整数倍の長さに限定される。そのため、可変長のメッセージに対しては、ブロック長の整数倍となるようにパディングを施す必要がある。この方式の特徴は、以下の通りである。

［長所１］：ハッシュ関数の安全性をブロック暗号の安全性の問題に帰着させることができる。

［長所２］：ブロック暗号の並列構造を利用し、処理速度の向上が可能となる。

［短所１］：ハッシュ値の長さと、ブロック暗号のブロック長が同一であるとハッシュ関数の強度が不足する場合が多く、２つ以上のブロック暗号を組み合わせることが必須である。

［短所２］：一般的に、総計算量が多く、高速化のためには、並列処理可能な環境が必要となる。

2.4.7 ハッシュ 関数の 特性と 応用

ハッシュ関数のＭＡＣ以外の応用として、

(1) **鍵の導出（鍵管理への応用）**

ＰＯＳ端末などで、現在のセッション鍵を露呈することなく、次のセッション鍵を生成する場合や、一方向性関数をベースにした「One-time password」の生成。

(2) **疑似乱数の生成**

一方向性関数を使って、疑似乱数生成を行う場合には、発生される系列の性質が明らかにしておかねばならない。一般に暗号化関数を疑似乱数生成に使った場合、生成される系列は「暗号学的に良い乱数である」と保証することはできない。

, 1997, CRC Press
[4] 櫻井孝一監訳：「暗号理論の基礎」、1996 年、共立出版
[5] “Open Design”No.14, 1996 年、CQ 出版
[6] 池野信一、小山謙二：「現代暗号理論」、1986 年、電子情報通信学会
[7] 辻井重男：「暗号」、1996 年、講談社選書メチエ
[8] 電子情報通信学会：「『暗号アルゴリズムの設計と評価』ワークショップ講演論文集」、1996 年、電子情報通信学会
[9] 日本アール・エス・エー株式会社：「FAQ World」
URL: http://www.rsa-japan.co.jp/faq/index.htwl

2.5 乱数

2.5.1 乱数とは

乱数は、サイコロを振る、ルーレットを回す、といった方法でゲームに偶然性を導入する道具として古くから使われている。その後、モンテカルロ法・オペレーションズリサーチ、サンプリング統計・品質管理などの道具として、その性質が論じられるようになった。近年になりセキュリティシステムにも欠かせない道具として注目されている。

乱数を発生させる方法には大別して二つあり、その一つは物理的手段で発生させる物理乱数であり、もう一つは算術式で発生させる算術乱数である。言い換えれば、**真性乱数**と**疑似乱数**である。

(1) 真性乱数の性質には、全くの不規則、平等な確率、前後が無関係、周期が無い（無限に続く）などがある。従って算術式で発生させることは出来ない。また、常に使い捨てである必要がある。乱数表などに収録したものなど切り取られ固定的に記録されたものは、真性乱数とは言えない。

物理乱数は、ランダムに変化する物理現象を利用して発生させるものであり、パワー・スペクトルが白色スペクトルであり自己相関係数がゼロとなるような物理現象を選択し、適切に数値化すれば一様乱数が得られる。単に予測し難く変化するというだけであれば、真性乱数とは言えないので **ランダム要素**とでも言うべきである。

物理乱数は独立事象であり、棄却法（0-0:棄却, 0-1:0, 1-0:1, 1-1:棄却）やくくり合わせ法（0-0:0, 0-1:1, 1-0:1, 1-1:0）で性質を改善できる。

(2) 算術乱数は一定の算術式で発生させたものであり、何らかの規則性が残り、真正乱数とはなり得ないので疑似乱数と呼ぶ。疑似乱数の発生法には、古くから使われているレーマー法のほかブロック暗号やストリーム暗号を利用したもの、カオス関数およびハッシュ関数を利用したものなどがある。

算術乱数には発生法に固有の規則性があり、目的と発生法に合わせて性質を改善する方法を考案する必要があるが、ソフトウェアのみで手軽に発生できることと再現性があるため広く使われている。クヌースの準数値算法／乱数にも、算法M、算法Bなど複数の乱数生成法と蓄積テーブルによる改善法が示されている。再現性は、モンテカルロ法への利用やシステム開発時のテストとしては好ましい性質であるが、セキュリティシステムの本番運用としては好ましくない。

モンテカルロ法などに使う場合と、暗号やチャレンジレスポンスといったセキュリティシステムへの応用では、算術乱数に対する要求事項に差がある。主なものとして、不可逆性と初期値のアドレス空間がある。生成した疑似乱数列から多項式時間で生成式と初期値が推定できないこと、過去に生成した疑似乱数列から以後生成する疑似乱数列が予測できないこと。初期値として 128bit 以上のアドレス空間がほしい。

セキュリティシステムへの利用では、初期値としてランダム要素を導入することで、発生数列の予測を困難にすることも可能である。しかし、ランダム要素が真性乱数何ビット分換算の変動範囲かを検証した上でシステム設計する必要がある。

2.5.2 セキュリティシステムにおける 乱数の 役割

セキュリティシステムにおいては、システムの仕組みと運用方法を全て公開しても高いセキュリティレベルが達成できる事が望ましい。そういった要求を満たすには、暗号技術による通信・保存、相手の認証など様々な局面で誰にも予測できない真性乱数の継続的供給が必要となる。コンピュータ犯罪の７割が内部犯罪であるという統計やソフトウェア・ハードウェアに対するリバースエンジニアリング技術の進歩という背景があり、秘密を守ることで達成するセキュリティから仕組みとして安全なセキュリティへの移行が望まれる。

真性乱数の「全く規則性がない」、「全く統計的偏りがない」という性質がセキュリティシステムにおいて重要な位置を占める。真性乱数のそれらの性質を言い換えれば、いかなる方法でも理論的に（計算量的にでなく）予測不能であり全数探索の範囲を狭めるような工夫も不可能である。

主な用途として、各種パスワードの生成、暗号鍵の生成、ＩＤ情報の生成、署名付加情報の生成などがあり、暗号技術、ハッシュ技術と並ぶ重要な技術である。良質な真性乱数から直接生成した暗号鍵であれば、実用上他の鍵との重複を配慮する必要はない。

2.5.3 乱数の 種類と 性質

(1) **乱数サイコロによる乱数（物理乱数）**

極めて安価に、かつ手軽に採用できる方法である。人間によるオペレーションが必要であり、高速・大量の乱数を発生させることはできないがパスワードの類には適している。

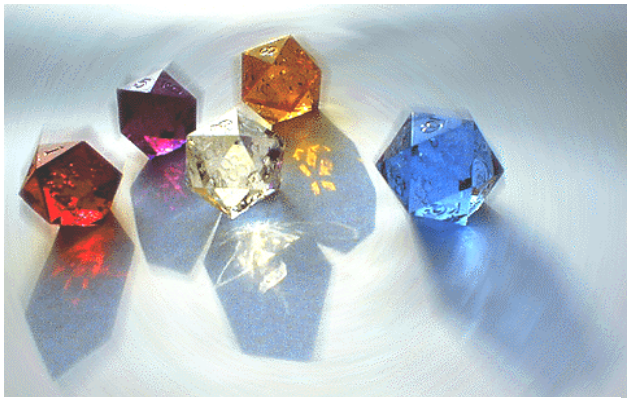


図2.5-1　乱数 サイコロ

乱数サイコロ無しでパスワードの設定を要求すると、氏名・生年月日・電話番号・住所の一部・口座番号・日付・時刻・適当な単語などが採用される事が多い。その結果として、辞書アタックといった最も一般的なハッキング手段が有効となる。パスワードの構成として英字・数字・特殊記号の混在を義務付けるなどの手法もあるが根本的な対策にはならない。

人間の生理として、まったくランダムな数字や文字を毎月考えることは不可能である。したがって、高いセキュリティレベルを望む場合は、パスワードが４桁なら４個、パスワードが８桁なら８個の乱数サイコロをアクセスソフトに添付し利用を義務付け、ソフトウェアで可能な範囲でチェックするべきであろう。しかし、切り取り固定化した乱数は乱数ではない、という原則通りパスワードの値を乱数サイコロにより求めるだけでは、辞書アタックなどを防止するだけであり遠方からのＶＴＲによる解析などの攻撃に対しては無力であることを考慮する必要がある（図 2.5-1 参照）。

(2) **抵抗体の熱雑音による乱数（物理乱数）**

抵抗体の熱雑音は、非常に多数の電子が作る微少電流を合成したものであり、それ

ぞれの電子の衝突・錯乱は相互に無関係に発生しているため、確立密度関数はガウス型になる。すなわち、一様乱数を生成することになる。

高抵抗の両端に発生する微少な雑音電圧を、ハイインピーダンス・低雑音のアンプで増幅すれば使いやすい雑音電圧を得られる。その電圧をＡ／ＤコンバータでデジタルデータとしてパソコンやＷＳに入力することで乱数列が得られる。

いったんデジタルデータになれば、乱数のある程度の検定および性質改善は容易である。棄却法、くくり合わせ法、変換誤差と電圧の不安定さを避けるための下位ビットと上位ビットの切り捨て、デジタル信号処理による周波数シフトなど低域成分の切り捨て、ＭＴ法など（後述）良質な疑似乱数との排他的論理和など様々な方法が考えられる。

この方法の利点は、自動的にかつ大量の乱数を高速生成できることであり、パスワードへの応用ではアルゴリズムが漏洩すれば無力な一定の数式で発生させる方式と違い、完全なワンタイムパスワードが実現できる。

一方特別なハードウェアを必要とする欠点がある。マルチメディア機能を持つパソコンでは、マイクＰＣＭ入力機能の利用も考えられるが外付けで雑音発生源が必要である。したがって、特に重要なクライアントに採用するか、サーバに設置して生成した乱数を暗号化してクライアントに伝送する方法がよさそうである。その場で発生させ、毎回使い捨てする真正乱数の伝送においては、一般的な共通鍵暗号で暗号化するだけで解読不能である（図 2.5-2 参照）。

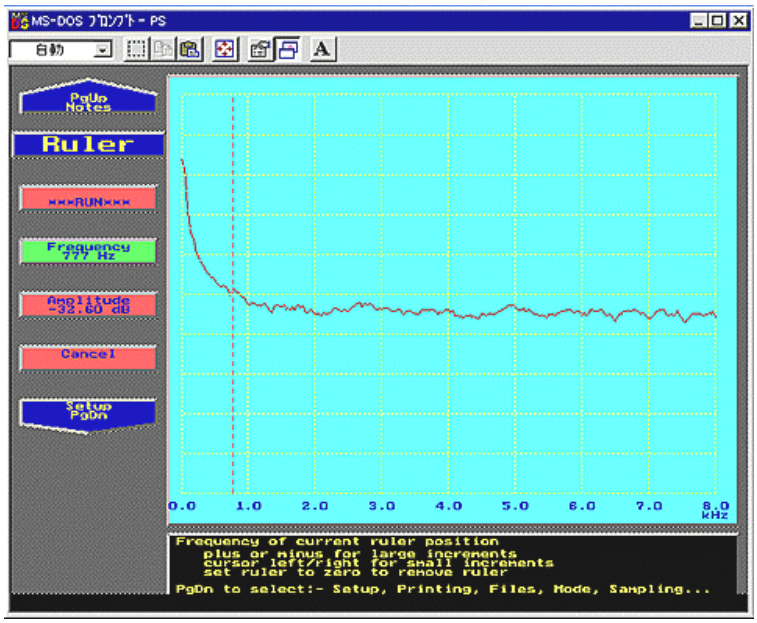


図2.5-2　熱雑音の 実測 パワースペクトルの 例

簡易に実験するには、高抵抗＋ハイインピーダンス・低雑音オペアンプの代わりに小型ラジオの離調雑音で代用できる。ラジオ・テレビの離調雑音（無信号時の雑音）は、回路内の半導体（１／ｆ雑音）や抵抗体(白色雑音)の集合体であり白色スペクトルにはならない。その実測パワースペクトルの例を示す。この例では、約１kHz 以下で１／ｆ雑音が優勢になっている。

理論的には白色スペクトルになるはずの抵抗体のパワースペクトルにも熱雑音の他１／ｆ雑音が観測される。これは、抵抗体に電流を流すことにより抵抗値の１／ｆゆらぎが電圧変動として観測され、電流値の二乗に比例する１／ｆ雑音が発生するためである。

熱雑音は抵抗値と抵抗体の絶対温度のみに比例する。一方、１／ｆ雑音は温度係数がほぼ０のマンガニン抵抗などでは発生しないほか、抵抗に電流を流さなければ発生しない。方法論として、熱雑音以外の雑音を実用レベルで無視できる範囲に抑制するか、発生を抑制せずに後の処理でマイナス１kHz の周波数シフトなどで切り捨てるといった幾つかの対処法が考えられる。

低周波雑音としては、熱雑音、１／ｆ雑音の他にドリフト雑音（熱、電源 etc）、バースト（ポップコーン）雑音などがあり実機の作成ではそれらにも配慮する必要がある。

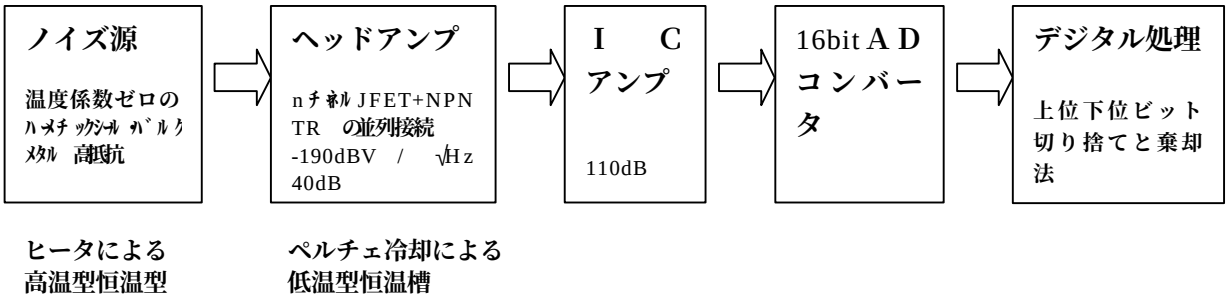


図2.5-3　正当的処理の 概念図

(3) **放射性物質の崩壊による乱数（物理乱数）**

放射性物質の崩壊によって発生する放射線は、放射性元素の各原子がいつ崩壊するかまったくわからない。したがって、その総体である放射線を検知管で拾ってカウントすることによりランダムなパルスが得られる。３～10[μSv/時]と十分な強度の放射線源が中古カメラレンズとして市場に出回っており、安価なガイガーカウンタキットと組み合わせてパソコンに繋ぎ容易に実験で確認出来る。

利点として、放射線の崩壊速度が外部環境による影響を受けないことがあるが、実用システムとしての採用は、現在の社会情勢として放射性物質の利用に理解が得られないこと、乱数の生成速度も遅いなど欠点が多く、あえてこの方式を選ぶ意味はないと考えられる。

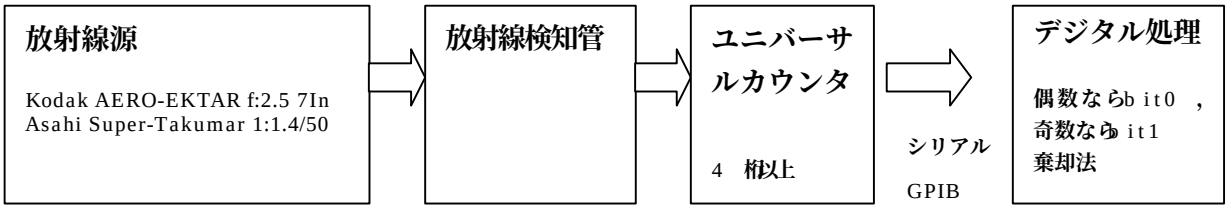


図2.5-4　実験システムの 概念図

(4) **水晶発振器の不安定さによる乱数（物理乱数）**

高安定の水晶発振器や原子発振器の周波数の揺らぎがアーラン分散を示すことが知

られている。したがって、非常に高い周波数の発振器の信号をパルス列とみなし、非常に低い周波数の発振器の信号をゲート制御信号として用い、パルス列の数の奇数偶数判別結果を二進数とみなすことで乱数が得られる。ただし、実際の周波数の揺らぎには１／ｆゆらぎが加わるため、その対策が必要である。

装置が高価で乱数の生成速度が遅く、高速・高精度のＡ／Ｄコンバータが安価に入手できる現在では特別な利点を見出せない。

図2.5-5　実験システムの概念図

(5) **アナログカオス回路による乱数（物理乱数）**

カオスチップによるほかアナログアンプに非線形素子を含むフィードバック回路を付加することでカオス信号を得、デジタル変換することでカオス数列が得られる。デジタルカオスと異なり、同一の初期値からスタートしても同一のカオス信号とならない。これは、乱数の生成という目的には好ましい性質である。

乱数に近い性質があるほか、適切な定数を選べばビット０と１の出現頻度が等しいなどランダム現象に比べてコントローラブルであるというメリットがある。その反面、カオスとしての規則性があり厳密には乱数ではない。

(6) **キー入力のコードおよび入力時間のバラツキによる乱数（ランダム要素）**

ランダムにキー入力させて、そのコードとキー入力時間のバラツキを計測することでランダム性のある数値を得られる。

この方法の利点は、特別なハードウェアが要らないことであるが、乱数としての性質が悪く規則性があること、人間によるオペレーションが必要なことが欠点である。また、キーボードプロセッサのクロック以上の分解能は得られない。

オペレーションについては、普段からキー入力時間を計測してその時間のバラツキを蓄積しておくことで回避できるが、規則性はさらに強くなる。したがって、他の要素と組み合わせて補助的情報として使うのが良いと思われる。

(7) **マウス移動の軌跡および移動速度のバラツキによる乱数（ランダム要素）**

ランダムにマウスを移動させて、その軌跡とマウス移動時間のバラツキを計測することでランダム性のある数値を得られる。

この方法の利点と欠点は、前記キー入力によるものと同様である。やはり、マウスの検出クロック以上の分解能は得られない。

オペレーションについては、普段からマウス移動時間を計測してその時間のバラツキを蓄積しておくことで回避できるが、規則性はさらに強くなる点も入力によるものと同様である。したがって、他の要素と組み合わせて補助的情報として使うのが良いのも同様である。

(8) **ＦＤ、ＨＤなど媒体へのアクセス時間のバラツキは る乱数（ランダム要素）**

ＦＤ、ＨＤなどのアクセス時間のバラツキを計測することでランダム性のある数値を得られる。

この方法の利点は、特別なハードウェアが要らないことの他、人間によるオペレーションが不要なことがあげられるが、乱数としての性質が悪く規則性があることが欠点である。ＣＰＵ実行命令のシーケンスにより確定的に決まるので原則として処理するデータや事象によるバラツキしかない。

したがって、この方法も他の要素と組み合わせて補助的情報として使うのが良いと思われる。

(9) **リアルタイムクロックの値による乱数（ランダム纂 ）**

ある瞬間におけるリアルタイムクロックのカウンタ末尾桁には、ランダム性があるとみなせる。

この方法の利点と欠点は、前記ＦＤ、ＨＤなど媒体へのアクセス時間によるものと同様である。したがって、この方法も他の要素と組み合わせて補助的情報として使うのが良いと思われる。

(10) **レーマー法／線形合同法（算術乱数）**

古くから多用されている乗算合同法および混合合同法であり、長い周期と比較的良好な数列が得られる反面、生成された疑似乱数列を座標とする点を多次元の単位空間に打点すると、少数個の平行超平面の上にすべての点が乗るという性質がある。他にＬＦＳＲ（線形フィードバックシフトレジスタ）法もあるが基本的には同様なので省略する。

乗算合同法　$x_{I+1} \equiv a x_I \pmod{M}$　経験的にａは√Ｍより大きくＭ－√Ｍより小さい値がよい
Ｍは素数

混合合同法　$x_{I+1} \equiv a x_I + b \pmod{M}$　ｂは、0.211Ｍに近く４ｋ+１（ｋは整数）を満たす値がよい

(11) **平方採中法（算術乱数）**

レーマー法より拡散度が高い反面、生成される疑似乱数列が可能な数全体の幾つかのグループに分かれ、各グループには１つのエンドポイントないしエンドループ（比較的短い周期）があるという性質と、ゼロの並びが拡大するという縮退現象があり、周期や系列相関が理論的に計算できないため単独で使われることはない。

＜例＞
$123{,}456^2 \rightarrow 15{,}241{,}383{,}936$
$241{,}383^2 \rightarrow 58{,}265{,}752{,}689$
$265{,}752^2 \rightarrow 70{,}624{,}125{,}504$
$624{,}125^2 \rightarrow 389{,}532{,}015{,}625$

(12) **暗号技術による法（算術乱数）**

ブロック暗号に対して運用モードとしてＣＦＢモードやＯＦＢモードを適用して疑似乱数列を得るほか、ストリーム暗号のをそのまま使って疑似乱数列を得る。その疑似乱数列の性質に関する研究発表例は少ない。

(13) **デジタルカオス法（算術乱数）**

数式によるカオス数列を発生させる。アナログカオスと同様の性質を持つが、アナログカオスでは初期値の微少な差違の増幅により生成されるカオス信号の予測が不可能であり周期がないのに対して、数式によるカオス数列では同一の初期値からは常に同じカオス数列が得られ極めて長いが周期がある。これは、乱数の生成という目的には好ましくない性質である。カオス数列の乱数としての性質に関する研究発表例は少ない。

カオスの性質：ストレンジ（カオス的）アトラクタ、長期予測不可能性/短期予測可能性（バタフライ効果）、軌道不安定性（リアプノフ指数が一つは正の値を取ること）。

カオスの解説：URL: http://plaza2.mdn.or.jp/~nif/paradigm/Chaos.htm

ＧＣＣカオス暗号の概要：URL: http://www.iisi.co.jp/reserch/GCC-gaiyou.htm

(14) **Ｍersenne　Ｔwister　法（算術乱数）**

2^{19937}-1 **というメルセンヌ素数の素数性を使い、**19937 **次元のベクトル空間における固有多項式の既約性を判定することで周期が** 2^{19937}-1(**約** 10^{6000})**で、**623 **次元超立方体の中に一様に分布する（ように見える）疑似乱数列を高速生成する。初期値のアドレス空間が** 2^{32} **と狭いのと非可逆的アルゴリズムでない（通常の線形合同法）など、このままでは暗号および認証用の乱数としては使えないので、他の方法と組み合せて使うことが考えられる。**

2.5.4 乱数の 生成と 配送

乱数は、必要となる都度必要とする場所で生成して使い捨てすることが原則である。しかし、クライアント側で個々に良質な乱数を生成できない場合、ランダム要素を初期値とした疑似乱数の生成で代用するか、サーバ側で発生させた良質な乱数を配送するかの選択がある。

乱数の配送は、セキュリティシステムにおける用途を考えると共通鍵暗号で暗号化するべきである。出来れば乱数配送専用の論理チャネルを設けて配送することが望ましい。真性乱数のみを伝送する論理チャネルの暗号解読は、暗号が解読できたか否かの終結条件が存在しないため理論的に不可能である。

なお、この用途に公開鍵暗号を使うと公開鍵を公開しているが故に理論的には解読可能となる。

図2.5-6　乱数の 配送

参考文献 [1]　宮武修、脇本和昌：「乱数とモンテカルロ法」、1978年、森北出版
[2] Stephen K. Park, Keith W. Miller 、西村恕彦訳：乱数生成系で良質のものはほとんどない」bit April 1993/Vol.25.No.4」1993年、共立出版
[3] 武者利光：「ゆらぎの世界」自然界の1／fゆらぎの不思議ﾞﾙｶﾞｯｸｽ B-442 」1980 年 講談社
[4] 橋口住久：「先端科学技術シリーズB5　低周波ノイズ、1991年、朝倉書店
[5] 飛田武幸：「入門：ホワイトノイズ解析」数理科学No.378 December 1994 」1994 年
[6] ＫＮＵＴＨ、渋谷雅昭訳：「準数値算法／乱数」、1981年、サイエンス社
[7] 岡本英司：「暗号理論入門」、1993 年 共立出版
[8] 合原一幸、徳永隆治：「カオス応用戦略」、1993 年オーム社
[9] 合原一幸：「カオスセミナー」、1994 年 海文堂版
[10] 高振宇：「ＧＣＣカオス暗号の原理と特徴」インターフェイス97. 6、1997 年 ＣＱ出版
[11] ザルツブルグ大学乱数研究チームのホームページ：
URL: http://random.mat.sbg.ac.at/
[12] パソコンパソコンソフトのリバースエンジニァリングツールとして利用可能なツールの例：
URL: http://www.iijnet.or.jp/TOOLCRAFT/seihin/ice.htm
URL: http://www.lifeboat.co.jp/product/sr97/sr97.html
[13] Mersenne Twister 法ホームページ：
URL: http://www.math.keio.ac.jp/~matumoto/mt.html

2.6 識　別

ＥＣシステムにおける認証技術については、「本人認証技術検討WG中間報告書」に詳しく述べられているので、ここでは他の局面でも必要となる実装技術として「識別」についてまとめる。

正しい認証のためには、通信路が暗号などにより保護されている他、正しいハードウェア（**ハードウェア／ファームウェア 環境**）・正しいソフトウェア（**ソフトウェア 環境**）により認証行為が行われているかの識別が必要である。

識別情報の系統的取得は、正しい端末であるかの識別の他、正しく動作するハードウェアやソフトウェアの組み合わせであるかを管理するための情報、ひいてはシステム全体の安定運用にも役立つ。

セキュリティレベルが高いものとして、24 時間体制でガードマンによる監視下に置かれた密閉構造の機器、耐解析構造のシングルチップＩＣだけで出来た機器、高度なセンサー・モニターシステム・高性能爆薬で構成された自爆構造の機器などが考えられるが、現実にはあり得ない。一方、セキュリティレベルの低いものとして、パソコン上で動作するアプリケーションソフトウェアがある。ここでは、セキュリティレベルの低いものを前提として検討する。

利用主体の識別（認証）には、端末（パソコン）の識別と利用者（人間）の識別がある。これらを区別して考えないと混乱を招く。センターが直接通信するのは、端末であり利用者は端末を経由して識別されるため、正しい端末が正常な状態で動作していることの確認が重要である。

識別は、識別対象を表すＩＤ情報と公開鍵暗号系における秘密鍵で暗号化された署名か共通鍵暗号系における機密鍵の共用によってなされる。通信路の安全性は、乱数によるワンタイム鍵による暗号化などによって確保できる。

2.6.1 識別対象

識別対象として、**ソフトウェア 環境**　**ハードウェア環境**　**利用者**がある。**利用者**の識別は、パスワード、身に付けた物、本人のみが知る情報、バイオメトリックスなどでなされるがここでは取り上げない。

2.6.2 実装技術

Windows95／ＮＴを例に取れば、レジストリにもある程度の情報が収集されている。しかし、本格的な識別としては、収集される情報の範囲が狭い、情報の深さが足りない、その時点での直接的情報ではないといった問題がある。反面ハードウェアに依存しないという利点があるので、レジストリの情報をハードウェアから直接収集した情報で補完する形態が望ましい。

現在、ハードウェアから直接広範囲かつ深いレベルで情報を収集するツールがない。セキュリティシステムで使うことを前提とした情報収集ツールの登場が待たれる。

＜レジストリから得られる情報の例＞

本　　体：搭載メモリ量

マザーボード：製造元

ＢＩＯＳ：製造元、製造年月日、リビジョン、ＢＵＳタイプ
ＣＰＵ：製造元、型番（クロック・ステッピング番号・モデル番号、プロセッサ数等はWin32ＡＰＩ）
ネットワークカード：製造元、型番、リビジョン、ＭＡＣアドレス
ＯＳ：名称、バージョン、プロダクトＩＤ、ＯＳフォルダ、テンポラリフォルダ、会社名、使用者名、バージョン番号、サブバージョン、初回インストール日

(1) **ソフトウェア環境**

① **識別プログラム自身の識別**

ネィティブなハードウェア上で動作しているかの判別と識別プログラム自身が改ざんされていないかの判別を同時に行う。

動作空間、ＣＰＵの各種レジスタ、時刻、乱数を識別プログラムに埋め込んでハッシュ値を求め、センターで管理する値とマッチングをとることで確認するなどの方法が考えられる。

② **ＯＳやドライバーなど動作環境の識別**

a) **個別環境の識別**

Windows95／ＮＴにおけるＯＳ名称とバージョン、ＣＤ番号とプロダクト番号、利用者名と会社名、製造およびサポート元、インストール日時などをセンターで管理する値とマッチングをとる。

＜例＞ＯＳ名称とバージョン:Microsoft Windows 95 4.00950 B,
プロダクト番号:2296-OEM-0014187-*****,
利用者名:Kenji Ashihara、会社名:ＣＩＣ,
インストール日時:97/07/28 13:09:02

b) **改ざんの検出**

ＯＳのカーネル部、割り込みハンドラ、デバイスドライバ、ＤＬＬなどが改ざんされていないかを識別プログラムがハッシュ値を求め、センターで管理する値とマッチングをとることで確認するなどの方法が考えられる。ウイルスに感染していない確認および英語版、日本語版、バージョンなど正しい動作が保証できる組み合わせであるか否かの判定にも役立つ。

＜例＞KERNEL32.DLL　Ver 4.00.1111 96/9.5 11:11:00 417,280byte
EXPLORER.EXE　Ver 4.71.1712.6 97/09/19 17:54 185,104byte
SPOOL32.EXE　Ver 4.00.950 96/09/05 11:11 20,992byte

＜例＞WNASPI32.DLL, WINASPI.DLL, ASPI32.SYS, WOWPOST.EXE のバージョンの一致

③ **アプリケーションプログラムの識別**

ＥＣなどの業務処理を実行するアプリケーションプログラムのハッシュ値を求め、センターで管理する値とマッチングをとることで確認するなどの方法が考えられる。

(2) **ハードウェア環境**

ＣＰＵ内部に（ヒューズＲＯＭなど）個体識別の番号を持つことが望ましいが、現状ではネットワークカードのＭＡＣアドレス（ＬＡＮ接続機構）や発信者番号通知（モ

デム／ＴＡ接続）をその他ハードウェア環境の総合的環境値で補強するなどの方法が考えられる。

ＭＡＣアドレス、ハードディスクのメーカ機種名＋ファームウェア番号、装置シリアル番号、バッドセクタ情報（プライマリディファクトリスト）、ボリュームシリアル番号、ＣＰＵ情報、ＢＩＯＳ情報、接続モデム／ＴＡ情報など。

ハードディスクのメーカ機種名＋ファームウェア番号、装置シリアル番号、バッドセクタ情報については書き換えるツールもなく識別能力が高い。

＜例＞ＭＡＣアドレス:00C01D807C17,

ハードディスクのメーカ機種名＋ファームウェア番号

:QUANTUM FIREBALL ST6.4A　A0F.0400,

:QUANTUM FIREBALL ST6.4A　A0F.0800,

:IBM-DAQA-32160　R40RA52A

:FUJITSU M1638TAU　5043

:FUJITSU M1638TAU　5043

:WDC AC32100H　22.06N06

装置シリアル番号:156707442044

:156719918436

:1W51WF79974

:03003273

:01058253

:WD-WT3350071882

バッドセクタ情報:シリンダ/ヘッド/セクタ

(00141/20/00088, 00260/11/00101,……..)

(00008/01/00055, 00019/03/00101,……)

ボリュームシリアル番号:2DE5-FACA,

:3402-F837

:18F7-4243

:88B1-EFBC

:3B57-14FF

:3673-1100

ＣＰＵ情報:Intel Pentium(MMX) 198.892MHz Family5 Model4 Step4,

ＢＩＯＳ情報:Acer 07/01/96,

接続モデム/ＴＡ情報: COM2=ZyXEL Elite 2864I JAPAN DSS1:V 4.13c

:COM3=NTT-TE MN128 2.12

プログラムによる自動的な情報の取得はできないが、人手によりパソコンの後部プレートから型番や製造番号を読み取りセットアップすることで固体識別能力は向上する。

＜例＞MODEL FMV6200D7C5 T96-0233-0 P/N CA04104-A101 1996-12 Rev.A P6Z00323

2.7 暗号アルゴリズムの公開に関する考え方

暗号アルゴリズムの公開／非公開については、一般に以下のように運用されている。

① アルゴリズム公開

ＤＥＳやＲＣ５など、誰でも入手できる技術資料としてアルゴリズムが公開されている。

② 有償でライセンス取得時に公開

特許や使用ライセンスの関係上、一般に公開はしないが、開発者に対して有償で公開（多くはツールを含めて）される場合がある。

利用者もライセンス契約により知る事ができる。

③ 非公開

開発者以外は知る事ができない。

現代暗号は、鍵の強さによってのみ暗号の解読を防ぐ事を主眼として開発されている。実際には、非公開としていても、関連機器の開発者からもれたり、リバースエンジニアリングなどでアルゴリズムを知る事は容易である。もちろん、このようなことは法律に抵触する可能性があるが、暗号を解読しようとするものに対しては意味を成さない。アルゴリズムが公開されているか、既知のものであれば、総当たり法以外の解読手段での耐攻撃性を検証する事が容易である。また、広く公開されているアルゴリズムであれば、破られたり、問題点があった場合にも広く伝えられる事となり、対策の検討も多くの協力が得られるとともに、迅速に対応できる。

このように暗号アルゴリズムに対する評価をシステム開発者だけでは下し得ない場合は、アルゴリズムが公開されているか広く知られている事により、外部の多くの協力を得ることができる。通常、商用システム構築の検討には、有限の資源（人的、金銭的）しか割けないためこのようにする考え方が一般的である。一方、アルゴリズムを秘密にしている場合、多くの人の手での検証がなされないだけでなく、問題点を指摘された場合に開発者が否定も肯定もできないために、対策が施せない可能性があるが、不特定多数による研究により、その暗号の解読技術向上を防ぐ意味合いもある。この考え方によれば、１万人が３年かけて解読法を研究する結果と、（特定の）300 人が 100 年かけて得られる結果が、統計的に同等になるということであり、構築したシステムの寿命を延ばし、結果として開発投資を抑えることと無用なシステムの更新を利用者に強いる必要が無いというメリットが生まれる。実際、英国などでは政府が採用する暗号製品についてはアルゴリズムが非公開であることが前提になっているし、Skipjack が非公開にされた理由も（盗聴云々は別にして）同じような発想ではないかと考えられる。

暗号機能を利用する場合には、上記のようにそのアルゴリズムの公開／非公開によりメリット・デメリットがあるため、コスト、利用目的・システムの重要度・システムの寿命などを考慮に入れ比較検討する必要がある。

3 暗号の利用と実装

前章では暗号技術に関する個々の技術要素について述べてきたが、暗号の実際の利用においては個別技術の安全性はもとより、その利用方法、実装技術、更には運用まで含めた、トータルのシステムとしての安全性が必要である。また、暗号の各種応用や利用技術には、個別に標準技術や注意点などもあり、システム構築にあたってはこれらについての知識、技術が求められることも少なくない。

本章では、暗号の応用面での各種事項についての技術内容、動向や留意すべき点について述べるものであるが、必ずしも前章の内容を理解している必要はない。むしろ、暗号利用のシステムを構築するにあたり、提供者側、利用者側いずれの当事者にも理解していただきたい内容をまとめたものである。

3.1 通信

そもそも第３者に知られることなく情報を伝えるということは、通信における暗号の基本的な要件であった。シーザー暗号から現代の最新の暗号技術に至るまで、通信での利用は暗号を考案する際の最優先課題であった。

これは、次に述べる保管については物理的なセキュリティにより安全が確保できるのに比べ、第３者が介在したり、第３者が容易にアクセスできるなど、通信においては安全確保が困難であるためである。今日においても、一般の通信手段である電話やＦＡＸ、電子メールなどのインターネット通信は（法的な規制があるにもかかわらず）決して安全とは言えないのである。

ここでは、通信における暗号利用について概説を行う。

(1) 通信とは

ここでは、一般的なデータ通信を指し、物理的な移動を伴わない情報伝達で、主として１対１で情報を伝達する（放送型でない）手段をいう。

通信では音声、画像（アナログ、ＦＡＸ）、デジタルデータなどが伝達される。これらの情報は一般的にリアルタイムに行われることが多く、また集約処理（バッチ処理）される場合も通信機器側から見ると即時応答が必要なことが多い。また、多くの場合、通信そのものはエンド・ツー・エンドで行われるものであり、送信側対受信側あるいは双方が送受信を行う形態を取る。多地点間を接続する場合、その接続は網（キャリア）側で行われることを前提にすることが多い。

一方、インターネットの通信システムでは、通信システム自身がオブジェクトとして振る舞えるような、分散システム型であり、網そのものをコンピュータなどの情報機器が直接利用する形態となっている。ただし、この場合も従来はエンド・ツー・エンドで利用される場合が多かった（この点は今後マルチキャストシステムが普及すると変わる可能性がある）。

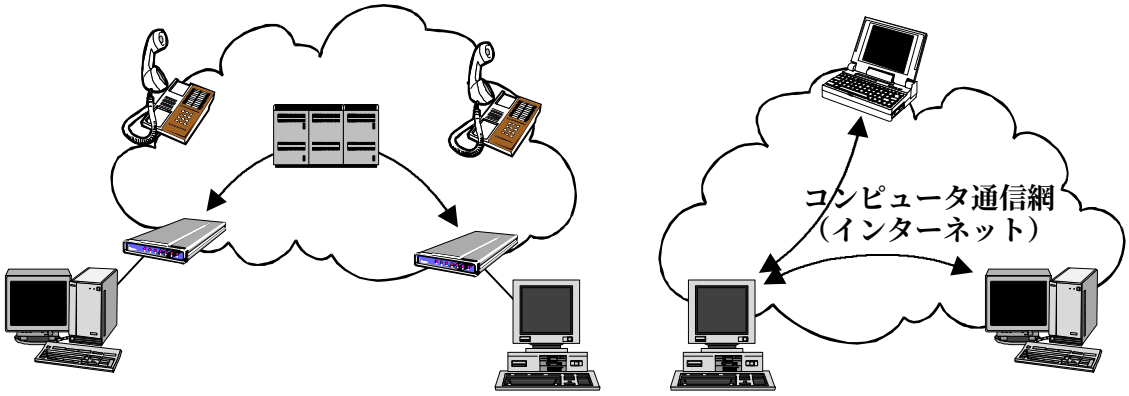


図3.1-1　通信形態

なお、インターネットなどにおける一般的な電子メールシステムに関しては、通信という観点と情報の共有保管という両面の見方があることに注意していただきたい。インターネットメールやLotus Notesなどは、メールをやり取りするクライアントと、メールの送受信（共有）と保管を行うサーバからなるクライアント・サーバ・システムである。このような形態の電子メールでは、クライアントとサーバの間、あるいはサーバ間では情報伝達のための通信がリアルタイムないしはバッチにて行われる。このとき、電子メールの情報自体はクライアント間で直接行われずに、サーバ上で保管されるのである。また、サーバ間の情報伝達も直接通信以外にバケツリレー式に運ばれることも多い。

上記以外のメールシステムでは、データをネットワーク上で共有することでメールシステムを構築している（ただし最近はクライアント・サーバ型への移行が進んでいる）。さらに、送受信した電子メールは通常クライアントの記憶装置内に保存されるという点も問題を複雑にしている（通常、電子メール通信とメール管理は同一のソフトで行われる）。ここでは一般的な通信とは別に、電子メールにおける暗号利用を述べる。

(2) 通信における暗号の利用

秘密保持のための通信の暗号化は、音声通話時代から特定用途（主として軍事、安全保障）向けに用いられてきた。また、データ通信が一般化する以前は、コンピュータ間の通信にも音声通信路が利用されて、金融業務や機密情報の通信には同様の音声の暗号化装置が利用されてきた。

これらは、当初はアナログ情報を暗号化するための特別な回路を用いていた。近年では、A／D変換装置の小型・低価格化と安価なデジタル暗号チップを組み合わせたものが一般的である。また、有線通話と異なり常に盗聴の危険にさらされる無線通話においても、このような暗号化機能は必須であり（その安全性は別にして）、全ての携帯電話、移動体電話に用いられている。

一方、デジタル・データ通信の発達により、デジタル情報を直接暗号化することが最近では主として行われている。この場合、暗号化／復号をエンド側が直接行う（通信データ生成時／受信データ確認時に行う）方式と、専用の暗号化装置を通信経路（RS－232C、電話回線、LAN回線など）に置き処理させる方式がある。また、最近では通信装置（モデム、TA）やルータ、ファイアウォールなどで行わせる方式も一

般的である。

エンド側が直接暗号化／復号を行う場合はデータの特性などを考慮し、安全性を高めることが可能であるが、システム（主にソフトウェア）開発時の設計・製作・検証が必要である。専用装置などで暗号化／復号する場合は、逆に安全性は装置に依存することとなる上、データ内容に応じた処理など融通が利かない面もある。

通信における暗号の要件は次のようなものである。

a) 送信側で暗号化されたものは受信側で直ちに復号される
したがって、その暗号文が存在する時間はごく短い時間に限られる
b) リアルタイム性が要求される場合、暗号化／復号に要する時間が限られる
c) 通信相手が特定できる場合鍵配布が安全に行ればよいが、不特定多数もしくは非常に多くの相手と通信を行う場合、鍵の共有メカニズムが必要となる
d) 通信データは時間とともに累積し、データ量が増大するため、解読の危険性にさらされない程度の頻度で定期的に鍵を更新する必要がある
e) 網自身が帯域削減のための圧縮処理を行う場合、暗号化データが圧縮しにくいため暗号化と圧縮処理をあわせて行えることが望ましい。
但し、圧縮処理では①情報の冗長度が低下するので暗号データの長さが短くなる、②圧縮ヘッダ部が入るのでこの部分の解読の手がかりとなる、という点に注意が必要である。

(3) 留意点など

暗号通信においては、一般の通信と異なり以下の点に注意する必要がある。

a) 遠隔地との間で定期的な鍵の更新が必要である
b) 暗号化データはランダムなビット列であるため統計的手法による圧縮が困難である
c) 内容を秘匿することはできるが情報伝達を行ったという事実は秘匿しにくい

(4) 電子メールにおける暗号利用

インターネットなどで用いられる電子メールは、従来の郵便や音声通話に比べて一般には盗聴、改ざんなどの危険が大きい。これは、そのようなサービスの利用がそもそも信頼できるグループ内や、閉じたネットワーク上での利用を前提としていたためである。しかし、エンタープライズ・レベルでこのようなアプリケーションを全世界規模で運用するには、セキュリティの面からも暗号利用がのぞましい。また、Lotus Notes などのようにワークフローを実現する上では、電子署名などで文書作成者・承認者を特定できる機能も必要である。

電子メールで暗号を用いるには、以下の方法がある。

a) 送信内容をファイル化し暗号化ツールで暗号化の上添付ファイルとして送信する
b) 電子メールシステムに備わっている暗号化機能を利用する Lotus Notes など
c) インターネットメールで標準的な暗号ツールを使用する

送受信する内容をファイル化し暗号化する手法は、現在ではどのメールシステムでも添付ファイルが可能であるため、もっとも汎用的な手法である。ただし、通信相手とあらかじめファイルの形式、暗号化・電子署名方式など取り決める必要がある。

プロプライアタリな電子メール製品の暗号通信機能はもっとも利用が簡単であり、一般にはボタンや送信時のオプションをチェックするだけで済む。ただし、当該製品ユーザ同士でしか利用できない、別々の組織の利用者間では認証などのしくみが必要、海外製品の場合暗号化鍵長が短いものが多いなど問題も多い。

インターネットでの標準的な暗号メールとしてはＰＥＭ、ＰＧＰ、Ｓ／ＭＩＭＥなどがある。

ＰＥＭ（Privacy Enhanced Mail）はもっとも早く標準化された方式だが、ＤＥＳおよびＲＳＡ方式（Ｘ．509）を採用しているため、鍵長が短い、米国外への輸出が困難、パテント使用料が必要などの理由で普及にはいたっていない[3]。ＰＥＭの商用製品は米国内でしか入手できない。

ＰＧＰ（Pretty Good Privacy）は Phil Zimmerman 氏の開発によるフリーソフトが原形で、ＣＡ（認証局）を必要としない相互認証方式により、インターネットコミュニティ内で非常に普及した方式である。データの暗号化方式に 128bit のＩＤＥＡを使用し、米国外へもソースコードの出版物の形で持ち出され、ＯＣＲによりソースコードの復元、再コンパイルされて流通している。なお、ＩＥＴＦではＰＧＰをＭＩＭＥ（Multipurpose Internet Mail Extension）に適用したＰＧＰ／ＭＩＭＥを標準化しＲＦＣとして公開している。また、ＰＧＰ社が商用ソフトとしてＰＧＰ Mail を商品化しているが、こちらは輸出規制を回避しているが、そのために鍵回復（Key Recovery）システムを内蔵しており、賛否両論があった。ただし、ＰＧＰ社が Network Associates 社に買収された際に、鍵回復は重視しない旨の発表があり、Zimmerman 氏とＰＧＰ利用者にとっては朗報であった。

Ｓ／ＭＩＭＥはＲＳＡ社（ＲＳＡＤＳＩ）が開発した暗号メールシステムで、ＩＥＴＦに標準化提案しようとしているが、ＲＳＡ社が暗号アルゴリズムのライセンス放棄に消極的であるため難航している。Ｓ／ＭＩＭＥは商用製品としては種類が多く、Netscape Communicator、Microsoft IE4 のほか、Frontier Technologies 社の Super ＴＣＰメールなどにも実装されている。なお、Ｓ／ＭＩＭＥはＲＳＡ社の可変長鍵アルゴリズムを利用しているため、輸出は比較的容易である。

参考文献 [１]　白橋明弘：「インターネットのセキュリティ技術」、NETWORLD+INTEROP ’97 TOKYO URL: http://www.netone.co.jp/netone/library/interop/ni97t320.pdf

[2] 白橋明弘：「ＰＣ環境におけるセキュリティ－電子メールを中心に－」／『SoftwareDesign』97 年 6 月号、技術評論社 URL: http://www.netone.co.jp/netone/library/pressclip/software_design976.html

[3] RFC 1421-1424

3.2 保　管

通信における暗号が、第３者に知られることなくコミュニケーションを行うのに対し、情報の保管における暗号技術は様々な目的に使用される。

一般的に情報の保管は、

a)　コンピュータ内部（クライアントおよびサーバ内）ＨＤＤなどでの一時的保管

b)　フロッピィやMO、磁気テープなど可搬メディアへの保存

として行われるが、その目的は様々である。

十分なアクセス制御の行えるコンピュータ内のハードディスクであれば、あえて暗号化機能を付加せずとも、情報を第３者に秘匿して保管することは基本機能として可能であるし、サーバに保存する場合も、同様のアクセス制御が可能であれば問題は少ないといえる。ただし、一般的に入手可能なＯＳでこれらの条件を完全に満たした物は少ないので、用途に応じて安全性に十分配慮したうえで、暗号化などの手法を追加することは有効である。一方可搬メディアへ保管する場合は、そのメディア自体に対するアクセスが容易であることから、十分に安全な暗号化を施すことが、情報の漏洩・改ざんを防止する唯一の方法となる。

(1) データ保存を目的とする場合

重要な情報は一般に誤消去防止や機械的破壊に備えるために、コンピュータ外部の磁気テープなどに保存する。これらの保存メディアが盗難に遭ったり、データのコピーを取られた場合は、企業などにおいては大きな問題となるため、少なくとも物理的なセキュリティ（入退室の管理など）を施す。

しかし、物理的なセキュリティも完全とはいえないし、災害対策での多重化保管やアウトソーシングなど、自分だけではセキュリティを守り切れない場合などもあるため、これらの媒体への保存時に暗号化を行うことがある。この場合、保存が長期化するようであれば鍵の管理についても十分配慮が必要であるし、媒体自体の安定性も考慮する必要がある。

また、トランザクションの正当性を保証するとともに、障害に備えるためのログデータが一般に取られるが、これらについても対象となるデータベースなどと同レベルのセキュリティが必要である。したがって、外部に保管される場合も、データベース自体のバックアップと同じ環境で、同じレベルの暗号化を行うことが望ましい。

なお、特殊な場合として、トランザクションの通信ログとして、暗号通信データを保存する場合がある。この場合は、ワーク鍵の変更もログとして残す必要があるが、D-H 法などで鍵交換をした場合はログ内容からワーク鍵を算出できなければならない（4.3.5 参照）。

(2) データ移動のための保管

通信などでは不安である場合や、そもそも通信インフラが不十分である場合、また、非常に大量の情報を受け渡ししたい場合など、可搬メディアにデータを保管し輸送することが行われる（現在においてもギガバイトクラスの情報伝達には最も安価な方法である）。この場合には、物理的なセキュリティは前述の場合に比べはるかに低くなるため、データの暗号化が非常に有効である。

ただし、受け渡しする双方が同じデータを扱えるシステムを準備する必要があるの

と、鍵の受け渡しが必要になる点が問題になる。

(3) **コンピュータ内での一時保管**

現在では、多くの情報がコンピュータ内で作成され、コンピュータ通信網を介して大量に流通する状況である。従来であれば、紙文書ベースで行われていたことが電子化されることで、電子情報の蓄積は指数関数的に増大している。そのため、コンピュータで作成する情報においても機密を要するものが増え、これらを暗号化するニーズが出てきた。

前述のように十分な安全性を持ったＯＳが普及し、本人認証手段もパスワード以外にＩＣカードやバイオメトリクスが利用できれば問題が無いが、現実に入手可能なＯＳにおいてはここまでの機能が提供されていない（パスワードを入れなくてもファイルが読めるのはひどいですが）。そのため、必要な安全性は自身で付加するしかない状況である。

この場合は、データの盗難・改ざんの恐れとして、第３者によるそのコンピュータ自体の操作があげられる。特にモバイル用のコンピュータやノートパソコンは物理的盗難や紛失の危険性が高いため、十分な注意が必要である。また、最近はデスクトップマシンやノートパソコンでもＨＤＤを取り外して内容を盗むことが非常に簡単であるため、これらについても同様に注意が必要である。

(4) **ＯＳなどが管理・保管する場合**

データの送受信や、アプリケーションの作業エリアなど、通常は記憶装置（メモリ）内に置かれる情報も、ＯＳにより補助記憶装置（ＨＤＤなど）に一時的に待避させられることがあり、俗にスワップファイルと呼ばれるものが作成される。

一般的にはこのようなファイルの内容はプログラムが実行する命令や操作するデータが含まれるが、一時待避が完了後も補助記憶装置内に書き込まれたデータは、一部が消去されただけの状態にとどまっている。このような状態のデータが盗み見られないようにすることは非常に困難であるため、特にセキュリティを重視するアプリケーションでは、スワップファイルの内容をすべて消去できるよう、特別な配慮が必要となる。

また、サーバなどでハードディスク障害に対応するためＲＡＩＤという手法でディスクアレイを構築することがあるが、この場合は利用者や管理者が意識すること無くデータの複製が作成されている。したがって、これらに対する物理的なアクセス制限を十分に行わないと、複製された内容をハードディスクごと盗まれかねないので注意が必要である。

(5) **電子メールでの保管**

電子メールなどで暗号化された送信文や受信文を保管する場合、第３者に盗み見されないように暗号化された状態で保管し、開く都度復号することが望ましい。

また、保管されている暗号化データの一部あるいは全部が、第３者により改ざんされるのを防ぐためのアクセスコントロールや、改ざんされたことを検出するための機能を付加することが望ましい。さらに、重要暗号化データの保管については、分散保管が考えられる。但し、データの運用性を考慮する必要がある。

6( )ＩＣカード等の耐タンパー性[1] があるものの保管(4.4.3 特殊媒体への保管(2)ＩＣカード を参照)

暗号化データ等を直接、ＩＣカードのＥＥＰＲＯＭへ書込むことが考えられるが、ＩＣカードの最大記憶容量は現状で、８ＫＢ～16ＫＢ程度であり保存データ量が限られる。したがって、一般的にデータはＨＤなどに暗号化して保存し、その暗号化鍵をＩＣカードに保存する方法が用いられる[1]。

［耐タンパー性］

不当なアクセス手段（電気的、電磁的、光学的、機械的など）を用いて、保管されている情報を盗み出すような攻撃に対して耐えられる性質(tamper resistance)をもつとき、一般的に耐タンパー性があるという。

参考文献：[1] 1997 年電子情報通信学会基礎・境界ソサエテイ大会講演論文集、電子情報通信学会

3.3 認　証

3.3.1 概要

認証は、情報の正当性・完全性を確保するための技術である。暗号では情報の秘匿を目的とするが、認証では情報が変えられていないことの確認を目的とする。

認証自体は暗号とは独立した概念であるが、暗号技術を用いることにより、その安全性・信頼性を高めることができる。認証は、正当性を示すべき対象により、メッセージ認証、ユーザ認証、端末認証、時刻認証などに分かれる。特に、メッセージ認証とユーザ認証を合わせもったもので、ある情報を確かに生成したことを保証する方式を電子署名（デジタル署名）という。電子署名の詳細については、次項 3.4 で説明する。

証明書は、認証機関（ＣＡ）の秘密鍵で署名され、発行される。最も簡単な証明書のフォームには公開鍵と名前のみで、一般的に使用されているものは、鍵の満了日、証明書を発行した認証機関の名称、証明書の連番などの情報が含まれる。

3.3.2 認証の 利用法

現在、公私を問わず各種情報のやりとりが活発化してきており、情報発信者が本当に本人であるのか、あるいは情報内容が改ざんされていないかなどを確認するため認証は、様々な分野で利用されていくことが予想される。例えば、以下のような利用法が考えられる。

(1) **企業間の取引や情報交換等への利用**

企業間で行われる取引や業務情報などのやりとりの際に、通信相手が本当に自分が意図する相手に間違いないか、あるいは情報内容が改ざんされていないか認証を利用し確認を行う。

(2) **企業内の決済や情報交換等への利用**

企業が社内の決済や情報交換を行う場合、本当にその権限を持った人が行っているか、また内容が改ざんされていないか認証を利用し確認する。

(3) **対消費者販売への利用**

ネットワーク上に仮想店舗を開設し、消費者に対してサービスや販売を行ったり、各種の情報提供を行う際に、その顧客の本人確認や注文内容の確認に認証を利用する。また、消費者側も仮想店舗が本当に存在するかどうか確認する際に認証を利用することができる。

(4) **金融サービスへの利用**

金融機関が、ネットワークなどを使用して顧客からの送金、口座振替などの注文を受けた場合、認証を利用しその顧客の本人確認や内容の確認を行う。

(5) **遠隔医療への利用**

医療機関が、ネットワークなどを通じて患者の診断を行う遠隔医療サービスにおいて、医療機関が患者の確認を行ったり、また患者側が医師の有資格などを確認するのに利用する。

(6) **電子公証への利用**

今後、電子的に作成された契約書などが特定の者によって真正に作成されたことを公証したり、電子的な形で証明書を作成し、これらの電子文書を保存し、その存在、

内容などを証明する電子公証制度が導入されると考えられるが、こうした電子公証においても公証申請者の確認や内容の確認に認証を利用することができる。

(7) **行政手続への利用**

行政機関への各種申請や証明書の交付をネットワークなどで行う場合、申請者の確認や申請内容の確認を行うのに利用する。

(8) **その他**

個人間での電子メールなどのやりとりに、認証を利用し相手の確認や通信内容の確認を行う。

3.3.3 認証における 留意点

今後ネットワークを活用した取引や情報交換が展開されていく上で、ネットワークを介した通信の信頼性が不可欠となってくる。取引情報などを暗号化し送信しても、その相手が他人のなりすましだった場合、セキュリティの意味をもたなくなる。よって取引や情報交換を行ったものが、確かにその本人であるか確認するための認証技術（本人認証）が、極めて重要となってきている。

その本人認証としてパスワードなどを用いた方法があるが、現在認証局が発行する公開鍵の証明書により、認証を行う方法が主流となってきている。そのことから認証局は、技術的に優れているだけではなく、経済的にも安定しているなど、利用者から信頼される機関であることが絶対条件となる。認証局が発行する証明書とは、ＩＳＯが制定したX．509に基づいており取引を行う人々が使用する公開鍵に対して、それが本人の鍵であることを確認した認証局の署名がなされたものである。

本人認証についての詳細は、電子商取引実証推進協議会（ＥＣＯＭ）ＷＧ０６の「本人認証技術検討WG中間報告書」を、また認証局の詳細は、ＷＧ０８の「認証局運用ガイドラインα版」を参照のこと。

3.4 電子署名

3.4.1 概要

通常、公的な文書には捺印や署名によって偽造を防止しているが、電子的な情報に対して同様の機能を果たさせるのが電子署名である。

電子商取引を行う場合には、取引相手が本人であるか、文書が偽造されていないか、データが不正でないかなどを見極める必要がある。本人の確認や偽造、不正をチェックする方法として電子署名が使用させる。

(1) **電子署名の原理**

一般的に電子署名には、公開鍵暗号を使用し、以下の手順で確認や偽造、不正のチェック行われる。

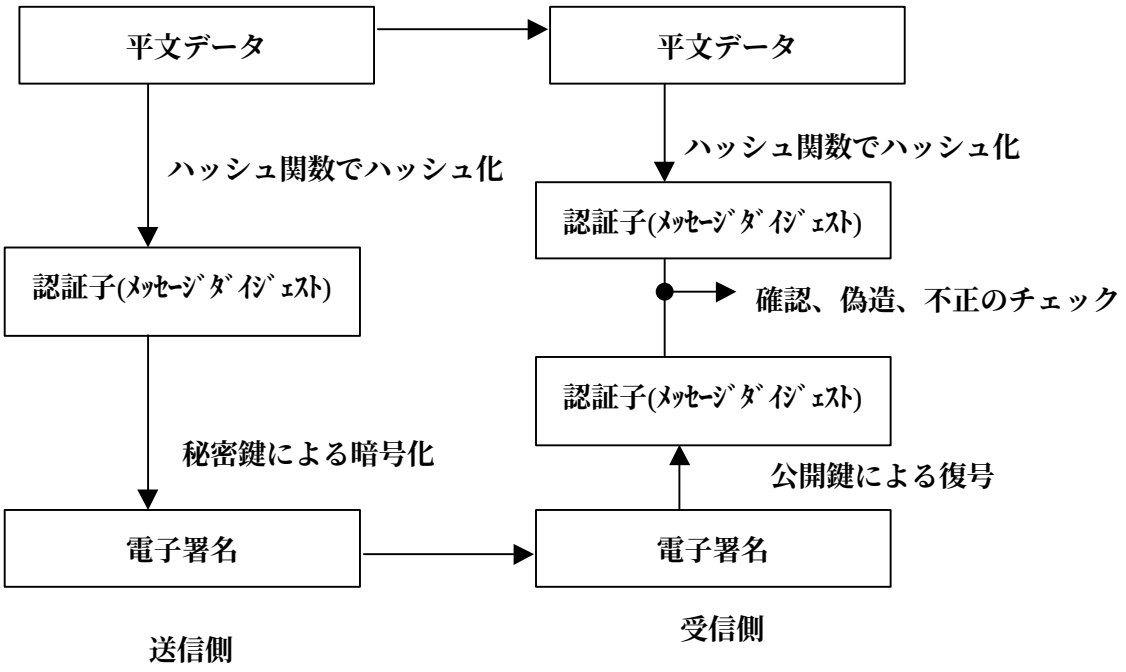


図3.4.1 基本的な 電子署名の 手順

(2) **電子署名によるデータの保全と問題点**

電子署名には通常公開鍵暗号が用いられるため、データが正当であるかどうかは公開鍵暗号の攻撃に対する強弱によって変わってくる。公開鍵暗号に対する攻撃については、2.3.4 の公開鍵暗号方式の種類、攻撃法、性能の項で詳しく述べられている。

また、いくら公開鍵暗号の攻撃に強くても平文データをつじつまが合うように変えられる場合も考えておかなければならない。このようなことを防ぐためには、平文データのメッセージダイジェスト作成方法、つまりハッシュ関数についても考慮しなければならない。

もう一つ問題点として、紙に書かれた文書の署名や捺印と違い、電子署名は簡単にコピーができることである。紙に書かれた文書の署名や捺印は、物理的に本物と偽物を区別できる可能性があるが、電子署名では区別かつかないからである。この点につ

いては、平文データに日時や連番などを付けて使用するなど使い方に注意が必要である。

3.4.2 代表的な 電子署名方式

電子署名の主な方式としては、以下に示す(1)～(7)の方式などがある。

また、次に販売店などが購入要求の受理などをカード会員に送る場合に、電子署名を使用した実際の手順例を示す。

① 販売店はメッセージを作成し、ハッシュ関数によりメッセージのハッシュ値（認証子またはメッセージダイジェスト）を作成する。ハッシュ関数は、メッセージの一部を改ざんしてもハッシュ値が大幅に変わる性質を持っている。

② このメッセージダイジェスト（MD）を販売店が秘密裡に保持する署名用の秘密鍵で暗号化する。MDはデータ長が短いため暗号化の処理時間が短くて済む。もとのメッセージ全体を署名用の秘密鍵で暗号化すると処理時間がかかってしまうためMDを使用するわけである。この暗号化した結果が電子署名と呼ばれる。

③ 販売店は作成した電子署名ともとのメッセージをカード会員に送る。

④ カード会員は、公開されている販売店の署名用の公開鍵で電子署名を復号し、MDを得る。

⑤ 送られてきたメッセージをハッシュ関数でハッシュ化を行い、その結果と電子署名を復号し得られたMDを比較する。

⑥ その結果が一致していれば、次のことが確認できる。

- 送られてきたメッセージは販売店が作成したものである。
- また、そのメッセージは改ざんされていない。

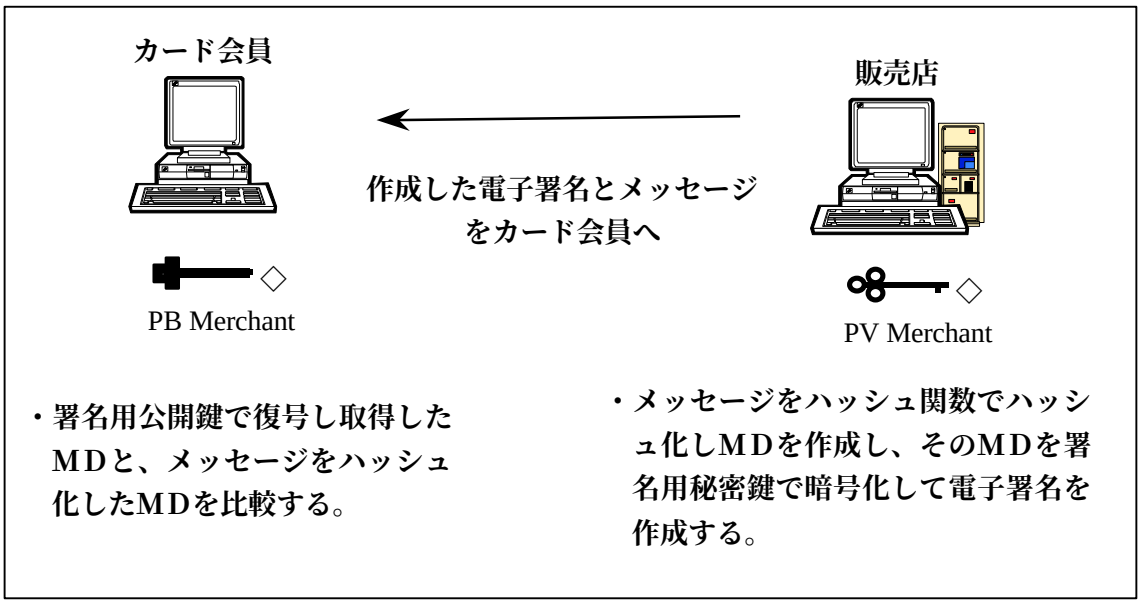


図3.4-2　電子署名を使用した 実際の 手順

(1) **ＲＳＡ方式**

電子署名の方式として代表的なものが、このＲＳＡ方式である。一般的にＲＳＡ方式による電子署名は、秘密鍵を用いて署名文を作成し、公開鍵を用いて署名文を復号し検証を行う。この場合、平文全体に署名する方法と平文を圧縮した文に署名する方法の二種類がある。実際には、ハッシュ関数などでデータを圧縮し署名を行う方法が多く用いられている。

(2) ElGamal **方式**

この方式もＲＳＡ方式と原理は同じで、秘密鍵で署名文を作成して、公開鍵で復号し検証を行う。ＲＳＡ方式は、「落とし戸つき一方向性関数」がベースになっているが、ElGamal 方式による電子署名は、「離散対数問題」をベースとしている。ElGamal 方式では、「認証子」の作成がＲＳＡ方式より高速に行えるが、大きさは約二倍となり検証時間もＲＳＡ方式よりかかってしまう。

(3) **ＤＳＡ（**igital Signature Algorithm **）**

ElGamal **方式をアメリカで改良しようと、**1991 **年に「デジタル署名標準（ＤＳＳ）」として提案されたのが、ＤＳＡ（**Digital Signature Algorithm**）である。ＤＳＡの基本は、**ElGamal **方式を修正し、平文からの「認証子」のビット長を短縮したものである。**

(4) **ＩＤに基づく電子署名方式**

公開鍵暗号を用いた場合、公開鍵ファイルが大きくなると鍵管理が大変となるため、これを解決する一つの方法が、ＩＤに基づく電子署名方式である。

この方式は、システムの加入者がそれぞれ自分で公開鍵を管理し、必要な時にそれを相手に送るという方式である。したがって「認証子」を検証する場合、相手の公開鍵をファイルより検索する必要もなく、鍵管理の問題も解消される。

しかし、相手より送られてきた公開鍵が本当にその人の公開鍵であるか確認をしなければならない。そこで送り手側は、鍵センターに自分のＩＤ（通常ＩＤ＝公開鍵となる）を送り、それに鍵センターで署名をしてもらう。相手側は、鍵センターの公開鍵を用いて送り手側の公開鍵であるかを検証し、確かならば、その公開鍵を用いて電子署名の検証を行う。この方式は、自分の公開鍵に鍵センターの「お墨付き」をもらうと言うことからお墨付き方式とも呼ばれることもある。

(5) Rabin **方式**

ＲＳＡ暗号と同様に暗号通信と電子署名の双方に使用可能な方式で、ＲＳＡ暗号よりも暗号化速度が高速といった利点がある。

(6) Schnorr **方式**

この方式は、ElGamal **方式の改良型である。鍵の生成で素数のサイズに束縛されなければ、ＤＳＡと同じである。**

(7) **楕円ＤＳＡ方式**

ＤＳＡ方式を楕円曲線上の離散対数問題に基づき署名を行う方式。楕円暗号についての詳細は、2.3 **章の公開鍵暗号を参照。**

3.4.3 デジタルタイムスタンプ

デジタルタイムスタンプサービス（ＤＴＳ）とは、暗号化技術によりデジタル文書に日付と時刻を付加させるためのタイムスタンプを発行する方法である。

このデジタルタイムスタンプは、タイムスタンプで示されている日付と時刻に、このデジタル文書が存在していたことを証明するために利用される。

デジタルタイムスタンプサービスが、タイムスタンプの信頼性を長期間保持しようとするならば、非常に長い鍵を持たなければならない。またその鍵は、不正から保護するために最高級のセキュリティのもとで保管されなければならない。

日付と時刻についても、長期間正確であるような時計に基づかなければならない。

よってタイムスタンプを生成するには、鍵と時計が同じセキュリティのもとで保管されていなければ不適切としなければならない。

参考文献：[1] 今井秀樹：「暗号のおはなし」、1993 年、財団法人日本規格協会
[2] 菊池豊彦：「インターネット世紀のコンピュータネットワーク暗号システム」1995 年、ＮＥＣクリエイティブ

3.5 暗号利用とそのコスト

暗号を利用する場合は、しない場合に比べて当然コストが増大する。このコストはセキュリティを守るために必要なコストであるが、実際のシステム構築においては、算出することが困難であったり、想定されないコストが発生することもある。

一方、安全性とコストの比較でいえば、10 万円の価値の情報を守るために 10 万円のコストをかけることは、自由競争下の企業においてはナンセンスであるが、どのような情報がどれだけの価値を持っているかを実際に検討している場合は少ないと思われる。

ここでは、暗号利用に必要とされるコスト要因を検討し、リスクとコストの考え方を提示することにより、暗号利用システムの設計・構築にあたる諸氏の参考になればと考える。

3.5.1 暗号利用の 問題点

(1) 原理的問題点

暗号を利用する場合の大きな問題点として、一度導入した暗号技術は、常に進歩する解読技術と計算機能力の向上により、安全性の低下が避けられないことがあげられる。一方では、解読技術の進歩が新たに強力な暗号技術を生み出すというジレンマもあり、解読技術の向上を止めるわけにもいかない。このことから、暗号利用に際してはそのシステムの寿命を考慮しその期間中十分な安全性を保てるような対策が必要である。

最も簡単な手法としては、十分に長い鍵を使用することである。2.2.5 でも述べたように、計算機能力の向上やアルゴリズムの高速化を考慮しても、100bit 以上の共通鍵暗号を容易に解読するのは、今後十数年は困難であると考えられている。同様の手法としては、鍵長が可変の暗号アルゴリズムを使用し、時間とともに鍵長を増大させていくことが考えられるが、システム構築コストとしてはたいして変わらない上、鍵長を判別する手法や、鍵長を変化させた際の検証など複雑化する要因が多くなる点に注意が必要である。よほどの理由が無い限りは、はじめから十分に長い鍵を使用する方が望ましいと考えられる。ただし、どれだけ長い鍵を用いても、安全性は「確率的に」保証される点は注意が必要である。すなわち、総当たり法で次々と鍵を試行していく場合、運が良ければ最初に鍵が見付かる確率もゼロではない。一方最後の鍵が正しい暗号化鍵である可能性もあり、平均として全ての鍵の半数を試行すると 50%の確率で解読できるに過ぎない。

別の手法としては、暗号アルゴリズムを複数利用できるようにすることである。すなわち、ある暗号アルゴリズムの安全性が極端に低下した場合、直ちに別の暗号アルゴリズムへの移行ができるようなシステムを構築することである。理想的には、常に複数のアルゴリズムが選択可能であることが望ましいが、後述するようなロイヤリティや工業所有権に関わる費用が増大する、使用されているアルゴリズムを特定するための管理手法が必要であるなど、コスト要因もある。また、アルゴリズム毎に鍵の選択基準（弱い鍵などの排除）が異なるため、単に暗号化／復号処理だけでなく、鍵生成・更新処理も対応が必要である。仮に、様々なアルゴリズムが選択可能であったとしても、相互運用性を保証するには少なくとも１つの共通なアルゴリズムについては利用可能でなければならない。

実際、ＤＥＳが非常に普及した背景には標準的手法を持たないものが淘汰されるという市場選択原理が働いたともいえる。しかしながら、このように単一のアルゴリズムにシステム全体が依存してしまうと、そのアルゴリズムの欠陥が判明した際には、対策が膨大な量になるとともに、特に電子商取引など信用が重視される場合においてはシステムに対する不信感が、場合によっては取引きの停止など社会的に大きなダメージを与える恐れがある。従って、暗号技術の実装においては複数アルゴリズムを利用可能なようにシステムを設計・構築し、できる限り複数の標準手法をサポートすることが望ましいといえる。

余談であるが、ＩＳＯにおいてもそのような配慮で暗号アルゴリズムについては標準制定を断念し、登録制度を導入している。ここに登録されているものは公開／非公開の別はあるがアルゴリズム毎に標準的な実装方法と検証データが規定されているので、そのような標準的アルゴリズムを採用していただきたいと考える。

(2) **実装上の問題点**

暗号機能を実装する場合は、個々のアルゴリズムの安全性よりもシステムとしての安全性に対する配慮の方が重要である。どんなに優秀なアルゴリズムもその価値を十分に引き出す鍵の生成、配送、保管システムが無ければ、絵に描いた餅にすぎない。最適な暗号アルゴリズムは、解読のための時間や手間、コストが、他の手段で元の情報を入手するよりも、多くなることを提供できればよいのである[1]。

特に、システム全体への攻撃の想定は非常に重要であり、どの部分が弱点になるかを予め判断しながらシステム設計を行う必要がある。たとえば、ＩＣカードなどでは、物理的所有者と情報の所有者が、同一の場合と異なる場合で、攻撃者の攻撃内容が異なるからである。すなわち、ＰＣ内のデータのように個人が必要とする情報を個人が保持する場合は、第３者のアクセスを防止するという物理的なセキュリティが確保されれば、比較的安全といえる。

一方、他者の情報を保持する場合には、物理的・論理的なアクセスが不可欠である。たとえば、電子商取引におけるサーバや、認証局のサーバ、不特定多数の操作が前提となる端末機器などは、通常そのシステムの利用者の情報を保管することとなる。ＩＣカード型電子マネーやネット上のｅキャッシュに格納されている価値（バリュー）はもともと発行元の物であると同時に、特定の所有者に限定されない転々流通性を持つ物であり、所有者を問わず同じ価値を持つものとなる。

これらの中に保管されている情報は通常、強固なトランザクション・セキュリティや耐タンパー性のメディアにより保護されるが、逆に言えばこれらにより防止しなければ、情報や価値を自分の都合の良いように改ざん・偽造することで利益を得ようと、誰もが考えるということである。したがって、このような場合は情報を格納している個別要素そのものの安全性を高めると同時に、改ざんや偽造を検知し他のシステムに影響をおぼさない仕組みが必要である。

鍵や秘密情報を守るための最上の方法は、物理的な（静電・電磁気的結合を含む）接触そのものを完全に遮断することである。たとえば、認証システムで使用されるＯＴＰ（使い捨てパスワード）生成パッドなどは、物理的に認証システムとはつながれず、利用者の目視と手入力が介在するし、計算機のＣＰＵとＯＳを使用しない認証シ

ステム（認証機能付きキーボードやキーボード・インタフェースの認証装置）は、万一攻撃者が特殊なソフトウェアを仕込んでも、ＰＩＮを盗聴することは不可能である。また、実装時に特に注意が必要な点としては、ソフトウェア上のエラー処理やプロセッサのエラー処理などの例外処理のパターンが解析されることにより、暗号の解読が容易になることがある。実際、電磁波放射によるＩＣカードの誤動作パターンによる攻撃事例がある。例外処理は、アルゴリズムの規定には当然記述されてないので、実装者がシステムにあわせてどのような処理を行うべきか判断する必要がある。

実際のシステムの実装時には、どのような点に留意すべきかを以下に例として述べるので、上で説明した各項目を考慮しながら、各自で検討を行っていただきたい。

① 秘密情報の暗号化

本人認証の為の情報、プライベートキーファイル、プライベートＩＤファイルなどの秘密情報は他人に不正使用されないように暗号化されていることが望ましい。一般的にはパスワードなどを用いて暗号化する。

② 秘密情報の削除

パスワードなどで暗号化された秘密情報は、パスワードを全数検索（全数探索、総当り）する事によって暴露される可能性がある。従って全数検索をさせないように、一定回数以上のパスワード不一致が生じた場合、秘密情報を削除などによって使用不能にすることが望ましい。

③ 秘密情報のコピー防止

上記のような使用不能処理をしても、秘密情報のコピーが容易に得られては効果が無い（使用不能になっても復帰が可能）。従って、コピーされた秘密情報は正常に動作しないようにする事が望ましい。

④ 秘密情報の格納場所

一般的にフロッピィディスク（ＦＤ）は複製が得られやすい（いわゆるデッドコピー）。秘密情報をＦＤに格納した場合はコピー防止機能が充分に効果を発揮しない場合がある。また、ネットワーク上に格納する場合はアクセスを適切に管理しなければコピーされる場合がある。従って、これらの場所に秘密情報を格納する場合は、その安全性、運用を慎重に検討する必要がある。

⑤ 本人認証の機能

一般的にパスワードの全数検索には、値を順次に変えて入力するプログラム（パスワードクラッカーと呼ばれる）を用いて自動的に行なう。これはパスワード検証（パスワードの一致、不一致を検証する）のみを行なう関数（機能）が用意されている場合に有効となる。従ってパスワード検証機能は単独で機能するのではなく、暗復号等の処理と一体化しており、本人認証が必要になった場合にのみ機能する方が良い。また、パスワードは関数のパラメータとして与えるのではなく、ダイアローグを開いてユーザが直接入力するなどのリダイレクトできない入力方法を取ると、全数検索の自動化がし難くなる（＊デバッガなどを使用すれば必ずしもリダイレクトできない訳ではないが、その実行には非常に時間がかかる）。

⑥　　本人認証情報入力時のカモフラージュ

パスワード入力は背後から覗かれる危険が常に伴うが、入力されたデータをそのまま使用するのではなく、前方一致や後方一致などの方法を取り込めば、入力時に余分なデータを入れ込むことでカモフラージュとなる。

⑦　　処理速度の均一化

例えばデータベースからデータを検索する場合、前半に当該データが存在していれば検索時間は短いが、後半に存在していれば長くかかる。これによりデータベース中のデータの分布が判る。同様に、ある関数の処理時間がその関数に与えるパラメータと相関がある場合、予期せぬ情報が漏洩する場合がある。これを防ぐ為に時間調節の為のダミー処理や、パラメータそのものの均一化（値の幅、ビットの分布）を計る必要がある。

⑧　　関数の深階層化

実行オブジェクトから逆アセンブルを経てプログラムを解析する場合、幾重にも関数コールが存在すると解析者の負担は非常に増大する。階層のレベルによって独立オブジェクトとして存在していると、構造を理解する手助けになるので、全ての階層が一つにまとまっている方が解析の精神的な妨げになる。

⑨　　情報のカプセル化（クラスなど）、消去

情報のカプセル化を行い、その存在を隠す。その他に、使用したメモリは全てクリア（またはダミーデータの書込み）する。また、プログラム終了後に秘密データの痕跡が残らないようにする。

⑩　　実行モジュールの暗号化

多くの場合、実行モジュールを解析するにはまず、実行モジュールのファイルを逆アセンブラで処理し、解析ファイルを紙に印刷し机上でトレースを行う。実行モジュール（の本体）が暗号化されていればこの処理がおこなず、デバッガを用いて少しずつプログラムを走らせ、メモリに展開されてからファイルに出力させなければならないので手間がかかる。

3.5.2 リスク 分析

そもそもコストをかけて暗号化を施すデータは、そのデータが解読されたりした場合にどのような影響を及すかのリスクを算出したうえで、必要となる対策を施さなければならない。ここでは、おのおののデータが持つリスクとして何を考えなければならないかを検討する。

(1) 解読（漏洩）リスク

第一に､暗号化データが解読された場合、どのような影響があるかを検討しなければならない。この場合の解読は、鍵を特定できるレベルの解読と、鍵は特定できないが暗号文に対応する平文を求めることができるレベルに分類できる。

鍵が特定される場合は、解読されたデータだけでなく同一の暗号化鍵（共通鍵、公開鍵）で暗号化されたデータについても復号が可能となるので、全体としてのリスクを考慮する必要がある。

また、鍵の特定無しに暗号文から平文が解読されるリスクについては、通常特定の

平文と暗号文のペアがすでに入手されていることや、同一の鍵による複数の暗号文が必要になるため、セクション鍵の更新頻度を適切にすることで原理的に困難とすることが可能である。

一般的には秘密情報などが読み出される場合にどのような影響があるかを考慮する必要があり、秘密情報自体の価値の算出と、波及する影響の総和を検討しなければならない。具体的には情報の作成、入手に関わる費用、情報を第３者が入手し使用した場合の損失、契約書類など相手がある場合の補償、プライバシーなど判例から想定できる補償などが対象となる。

(2) **改ざんリスク**

上記のうち、鍵が特定されてしまう場合には、改ざん、偽造という操作が可能になる。そのうち、改ざんについては、鍵を特定されなければ（確率的には）一般的に不可能なレベルであるといえる。改ざんのリスクは、本来改変されないはずの情報が改ざん（変形）させられることにより、誤った情報を伝達することによるリスクである（保管も時間軸の離れた伝達といえる）。

ただし、改ざんは本来伝達されるべき内容のごく一部について行われることが多く（金額の改変や口座番号の改変など）、アプリケーション上で内容の改変を特定できるしくみが無ければ検出が困難である。この場合のリスク算定は、条件付けにより大きく変動する。たとえばクレジット決済のように上限を設けても、どの程度の回数で改ざんが行われるかにより、被害が数桁変わることもまれではない（テレホンカードやパチンコカードなどの例）。

(3) **偽造リスク**

偽造は、一見困難なようであるが、鍵が特定できずに解読が可能なレベルであれば、複数の暗号文から別の暗号文を偽造することが可能な場合がある。

ただし、偽造の場合はたとえば元の平文が存在しない状態で新たな暗号文が生成されるので、通信の場合にはシーケンス番号が重複する別の暗号文が存在したり、保管の場合にはもとの情報やメディアを消去しない限り２つのもの（情報）が存在するという矛盾から、検出されることが用意である。このように、アプリケーション上で対策を施すことが用意である点、再犯が困難である点などを考慮すると、改ざんほどシビアな算定は必要無いともいえる。

(4) **なりすましリスク**

なりすましとは、広義には認証手段をごまかして、別の対象（エンティティ）になりすますことを言う。一般には別人になりすまして秘密の情報へアクセスしたり、架空の取引きを行うなど、情報の漏洩や損害の発生（この場合他の利用者）がつきものである。これらについては、個々に上記のようなリスクを算出しなければならい。

暗号技術を利用しない認証（暗証番号やバイオメトリクスなど）の場合は、認証技術としてのリスク管理（誤認証や認証システム自体のトラブル対策）はもちろん必要である。ただし、本人認証のためのシステムであるため、認証システムを更に人間がチェックするのであれば、なりすましのリスクは小さいといえる。

一方、3.4 のような電子署名を用いた認証や、ＵＮＩＸのパスワードやＣＨＡＰ（Challenge Handshake Authentication Protocol）など暗号技術を利用した認証の

場合、前述のような解読・改ざん・偽造などの手段によりなりすまされた場合は、通常はそのメッセージだけが到着するため、本人かどうかを別の手段で確認することは不可能に近い。従って、なりすましそのものの回数がどの程度まで許容されるかも考慮して、リスクを算定しなければならない。

(5) **鍵遺失リスク**

一般にマスター鍵など重要な情報は厳重に保管するが、情報の性質上無闇にバックアップをたくさん取ったり可読性の有る媒体（メモなど）にして残すことはセキュリティ上問題である。

一方、計算機上で保持する情報は、地震や火事などの災害や機器の故障、ソフトウェアの不具合、操作ミスや人為的な破壊行為により失われる可能性が常に存在する。したがって、共通鍵暗号による情報（データ）の保管や、公開鍵暗号による電子署名と暗号化を行ったものを保管する場合は、鍵が失われた場合にどうなるかを予め検討しておく必要がある。

3.5.3 コスト 要因

(1) **暗号処理コスト**

暗号機能をシステムに付加する場合は開発・実装費用が必ず必要になる。これには設計レベルから検証の費用などに加え、通常の処理部分においても、暗号化機能を付加する構成要素との連携のために必要な設計上の配慮や、検証作業が困難になる点など通常のシステム以上の費用が必要である。また、暗号機能部分を外部から調達する場合は、調達のコストも必要である。

また、一般に暗号は特許対象になることも多く、ロイヤリティ不用の場合を除き、知的所有権に対する費用が発生する。外部から調達した場合などは、モジュールのインストールベースで課金されることもあり、調達条件と含めて考慮が必要である。

以上はソフトウェアや人的資源などに関するコストであるが、むしろ問題となるのは暗号化・復号のために必要な資源（の増大）である。通常の処理以外に暗号化・復号を行うために、処理プロセッサを強力なものにしたりＣＰＵを増やすなどの対応が必要となる。また、暗号化処理はかなり時間が必要であり、応答時間に対する要求がシビアな場合は、その分処理能力を上げるか、専用回路などの高速化の手法を導入せねばならず、これらのコストもあらかじめ想定しなければならない。

また、3.1 でも述べたように暗号データは圧縮が困難であるため、通信経路のスループット低下やバックアップ媒体の容量不足などを招かないよう、すべて非圧縮状態での算出が必要となる（一般的には２～３倍程度の容量を必要とする）。

更に、システムを稼動させるにあたっては障害対策や障害の切り分けのための運用・管理ツールが不可欠であるが、暗号化機能を付加したシステムでは暗号化されたデータの処理経路（トランザクションや通信経路）上でのトラブル時に対応できるようなツールの作成時にも、暗号化や復号の機能や、トレース情報の比較など、通常の処理システムに比べて設計・製作・検証に手間がかかる点も見落としてはならない。

最後に、どのような暗号化機能であっても、一般的なコンピュータのハードウェア、ＯＳ環境下で処理を行う以上、暗号化・復号機能モジュールや作業データなどを安全

,

COUNTERPANE SYSTEMS

URL: http://www.counterpane.com/whycrypto.html

3.6 暗号機能の検証

3.6.1 検証の 必要性

(1) 複数機種によるシステム構築時の問題

データ交換、トランザクション処理での接続におけるプロトコル、各種ツールとのインタフェース設計に配慮しなければならない。

① 異なる文化で育ったプラットホーム間での問題

メインフレーム、オフィスコンピュータ、ワークステーション、パソコンでのアーキテクチャの違い、データ表現やコード体系の違い。

② 日本固有の問題として、同一文化圏であってもメカ 毎の差違

メインフレーム文化圏であっても富士通・ＩBM・日立・ＮＥＣ・ＵＮＩＳＹＳなど、多数のメーカ・機種・ＯＳに対応しなければならない場合がある。データ管理における制約の違い（最大レコード長、削除レコードの扱い、etc）、ＣＯＢＯＬにおける演算精度・データ表現などの非互換、高速処理が要求される場合の言語（アセンブラ、ＨＰＬ：ＮＥＣ、ＲＰＧ：ＡＳ400、etc）、ネットワークアーキテクチャ（ＴＣＰ／ＩＰ、ＦＮＡ、ＳＮＡ、ＨＮＡ、etc）の違いなどにも配慮する必要がある。パソコンにおいても多様なプラットホームという問題がある。MacＯＳ・Windows3.1／95／ＮＴなどＯＳの違い、ＡＴ互換機・ＰＣ９８・Mac などハードウェア環境の違いがある。

③ 相互間で開発時の問題以外にデータ交換における問題、運用における問題があり、検証および保証をどのように行うかが重要である。

(2) 複数開発者による相互運用性の保証

① 同一プラットホームであっても開発者が異なれば設計思想と実装方法の違いがあり得る。

② 性能追求かインターフェイスの標準化重視かなど。

③ ユーザインターフェイスおよび機能面での差違があるかの検証。

④ 実装上の差違によりデータ交換・相互接続面で問題がないか比較検証。

これら複数の開発者が開発したシステムの相互接続、および複数開発者が開発した製品やツールを採用したシステムの検証を、各メーカ・ユーザ・第３者機関がどのように分担するかという問題がある。

3.6.2 暗号機能の 検証方法

(1) 暗号ロジック実装の検証

通常のシステム検証と同様に、サブシステム単位での検証を行った上で、それらを順次結合しより上位のブロックとしての検証を進めるのが普通である。ここでは暗号化／復号機能を提供するモジュールの暗号ロジック単体の動作確認時に必要な点を挙げる。

①インタフェース仕様の確認

あたりまえのことであるが、どのような単位で暗号化／復号処理を行うかはシステムの要件や使用する暗号方式、モードに左右される。モジュール間の引数の型、

長さ、引き渡し方式（スタック利用 or ポインター）など、想定されるデータに対してオーバーフローなどを起こさないことが前提である。通常はブロック単位で暗号化を行うため、パディング処理もどのように行われるべきであるかを確認する（アルゴリズムによりある場合と無い場合がある）。また、引き渡す平文／暗号文／鍵などは、利用後直ちに記憶装置上から消去する必要があるが、どのモジュールが行うかも予め確認が必要であろう。なお、市販の暗号機能モジュール利用の際は、上記の点がどうなっているかメーカに確認が必要である。

② 暗号化結果の確認（サンプルテスト、ランダムテスト）

実際に暗号化を行い実行結果を検証する。サンプルテストでは、暗号アルゴリズムといっしょに提供される、特定の平文／鍵／暗号文の組み合わせが実現できるかどうかを確認する。このテストでは、実装結果が暗号アルゴリズム上正しいかどうかが判断できる。あるいは、すでにある製品やモジュールとの整合性をとるのであれば、それらの製品へ適当なサンプルデータを入力した結果と比較することも良い。

ランダムテストでは、とりうる範囲のできるだけ多くの平文と鍵をランダムに選択し、これにより出力される暗号文を統計的に検査し、分布の偏りやビット毎の０／１の割合などを検査する。これにより、実装された暗号アルゴリズムと、入力される平文が妥当な範囲であるかどうか判断できる。極端な偏りが発生するような場合は、その平文の集合はそのアルゴリズムを使用するのには向かないこととなるので、平文に対する何らかの処理（冗長データの付加、圧縮など）を行う方が良い（通常はアルゴリズムを変更することができないため）。

なお、公開鍵暗号方式では鍵の生成に時間がかかる上、暗号結果のばらつきは公開鍵を実現する数学的性質を考慮する必要があるため、通常このテストは行わない。

③ 復号機能の確認

上記と同様に（通常は同時に）、実際に復号を行い、正しい結果が得られることを確認する。

④ 相互の暗号化／復号テスト

通信やメディアの受け渡しが発生する場合、実際の通信やメディアの交換を行い、他の機器やソフトウェアで暗号化されたものが正しく復号できることも確認する。特に、3.6.1 の機種毎の差異についての確認が重要な点であり、データ列の並びに気をつけなければならない。

⑤ セキュリティ上の欠陥チェック

暗号モジュールは解析されたり不正な改造がなされないように、厳重に管理される必要がある。実行モジュール内部でも不正な改ざんが無いかどうかをチェックする機能は不可欠で、これらは実際の解析／攻撃を行ってみなければ確認できない。また、モジュール内およびモジュール間で使用される変数領域も、不正に監視されないようプロテクトされあるいは解読できないことを確認しなければならない。

(2) 鍵管理システム実装と運用の検証

鍵管理システムにおいても機能検証は欠かせない。特に鍵交換・鍵配布については、相互のやり取りが発生するために、機種毎の差異があると問題となるので十分な検証が必要である。

その他、鍵の生成（および更新）時には、正しく鍵がランダムに生成されているかも確認しなければならない。また、鍵の登録／廃棄については、一種のデータベースシステムとなるので十分なトランザクション能力と長時間耐久性なども検証しなければならない。

最後に、鍵管理システムとしてもっとも重要なのがセキュリティである。セキュリティに関しては、実際に攻撃を行い、問題が発生しないことを証明するだけではなく、実運用時にあらかじめ想定／検証した以外の手口で攻撃されないかどうか、また運用上のログや記録は正しく安全に保管されそのチェックと対策がなされているかなど、運用上の監査が必要である。セキュリティに関しても通常のビジネスプロセスと同様に、ＰＬＡＮ－ＤＯ－ＣＨＥＣＫ－ＡＣＴＩＯＮというサイクルが重要であり、後２者が欠けていては意味を成さないことを十分にご注意いただきたい。

(3) **標準化されている検証方法**

ＪＩＳやＩＳＯのような機関では様々な仕様は定義するが、実際の検証方法や相互運用性確保のための検証方法などは定義しないのが普通である。あくまで、暗号開発者の提供する検証データにより確認することが求められる。ただし、米国においてはＮＩＳＴがＤＥＳの実装と利用に関するガイドライン[1]を規定している。これには実装方法と鍵管理、キャラクタセットとの対応などが述べられている。

また、一般的な暗号モジュール（ここでは広くハードウェアや鍵管理、ＯＳなども含める）の検査プログラムも規定されている[2]ので、参考としては使用できる（ただし、ＤＥＳ、Skipjack、ＤＳＡ、ＳＨＡ－１だけである）。

(4) **暗号実装者による検証**

実装者による検証は、当然必要である。この場合も、できるだけ検証方法を体系付けて定義し、それに則った検証作業が必要である。また、検証結果については、不正な改ざんがされないよう正しい手続きで安全に保存する必要がある。

(5) **第３者による検証**

実装者と利用者だけでなく、第３者が客観的な検証作業を行うことも必要である。現在はそのような組織や企業はほとんど見られないが、今後はこのようなこともビジネスとして取り組むところがあるかもしれない。また、暗号機能に関する保険的考え方をすれば、保険機関などがこのような業務を行うかもしれない。

いずれにしろ、利用者にとって評価基準が広く統一できることは良いことであるが、検証手続きの正当性や、コストと内容などサービスの対比など、ユーザの考慮すべき点が増えることも注意が必要である。

参考文献 [1] "FIPS PUB 74, GUIDELINES FOR IMPLIMENTATION AND USING THE NBS DATAENCRYPTION STANDARD",NIST
URL: http://www.itl.nist.gov/div897/pubs/fip74.htm

[2] "Implementation Guidance for FIPS PUB 140-1 and the Cryptographic Module Validation Program",NIST
URL: http://csrc.ncsl.nist.gov/cryptval/140-1/1401ig.htm

4 鍵管理

4.1 鍵の種類と機能

情報セキュリティで扱う“鍵”はその用途によって様々な種類と呼び方がある、以下にその主な使い方と呼び方を説明する。

(1) **暗号化鍵と復号鍵**

図 4.1-1 に暗号鍵と復号鍵の図を示す。

- 暗号化鍵は、平文データや情報を暗号化するときに用いられる。
- 復号鍵は、暗号化されたデータや情報をもとの平文データや情報に戻す時に用いられる。

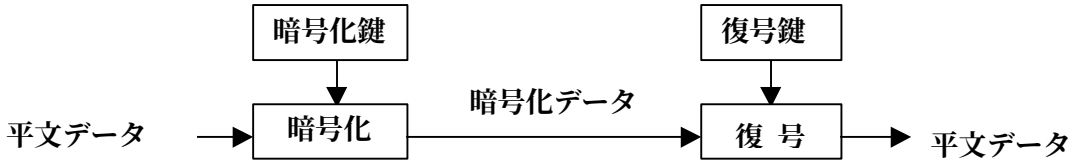


図4.1-1　暗号化鍵と 復号鍵

(2) **共通鍵**

共通鍵暗号（対称鍵暗号、慣用暗号）で暗号化、復号に用いられる鍵（暗号化鍵と復号鍵）で通常、共通の値である。

また秘密にしておくことから、秘密鍵と呼ばれることもある。

(3) **公開鍵と秘密鍵**

図 4.1-2 に公開鍵と秘密鍵の図を示す。

- 公開鍵は、一般的に公開鍵暗号方式における鍵ペアの一つで、公開するものであり、もう一方の秘密鍵が秘密に保たれている間のみ公開するものである（公開鍵方式における暗号化鍵、およびデジタル署名を検査する鍵として用いられる）。
- 秘密鍵はその使用を許された者のみが使用できるものである（公開鍵で暗号化されたデータの復号鍵、およびデジタル署名を生成する鍵として用いられる）。

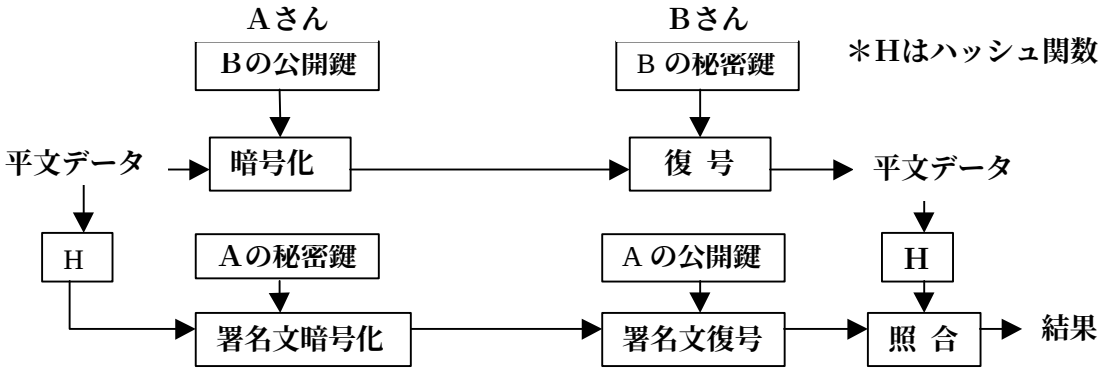


図4.1-2　公開鍵と 秘密鍵

(4) **作業鍵**

通常、長いデータの暗号化、復号を行う場合、共通鍵暗号アルゴリズムを用いることが多く、常に同一値の鍵を用いると、解読されやすい場合が考えられるので暗号化を多段で用いる。そのときに用いられる鍵を作業鍵と呼ぶことがあり、一般的にはその時に発生した乱数（実際には疑似乱数）データを用いる。また作業鍵は公開鍵暗号方式や、共通鍵を生成させる鍵管理アルゴリズムを用いて暗号化して、相手に配送する（作業鍵を暗号化する鍵をマスター鍵と呼ぶこともある）。図 4.1-3 に作業鍵の図を示す。

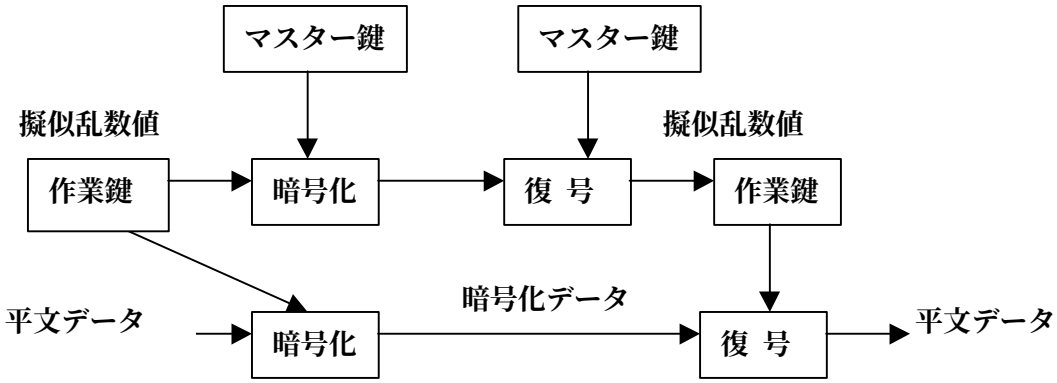


図4.1-3　作業鍵

4.2 鍵のライフサイクル

秘密鍵が常時変化しない簡単なシステムを除いて、暗号に用いられる鍵は、定期的に更新要求される。

鍵管理のライフサイクルを図 4.2-1　鍵管理のライフサイクルを示す。

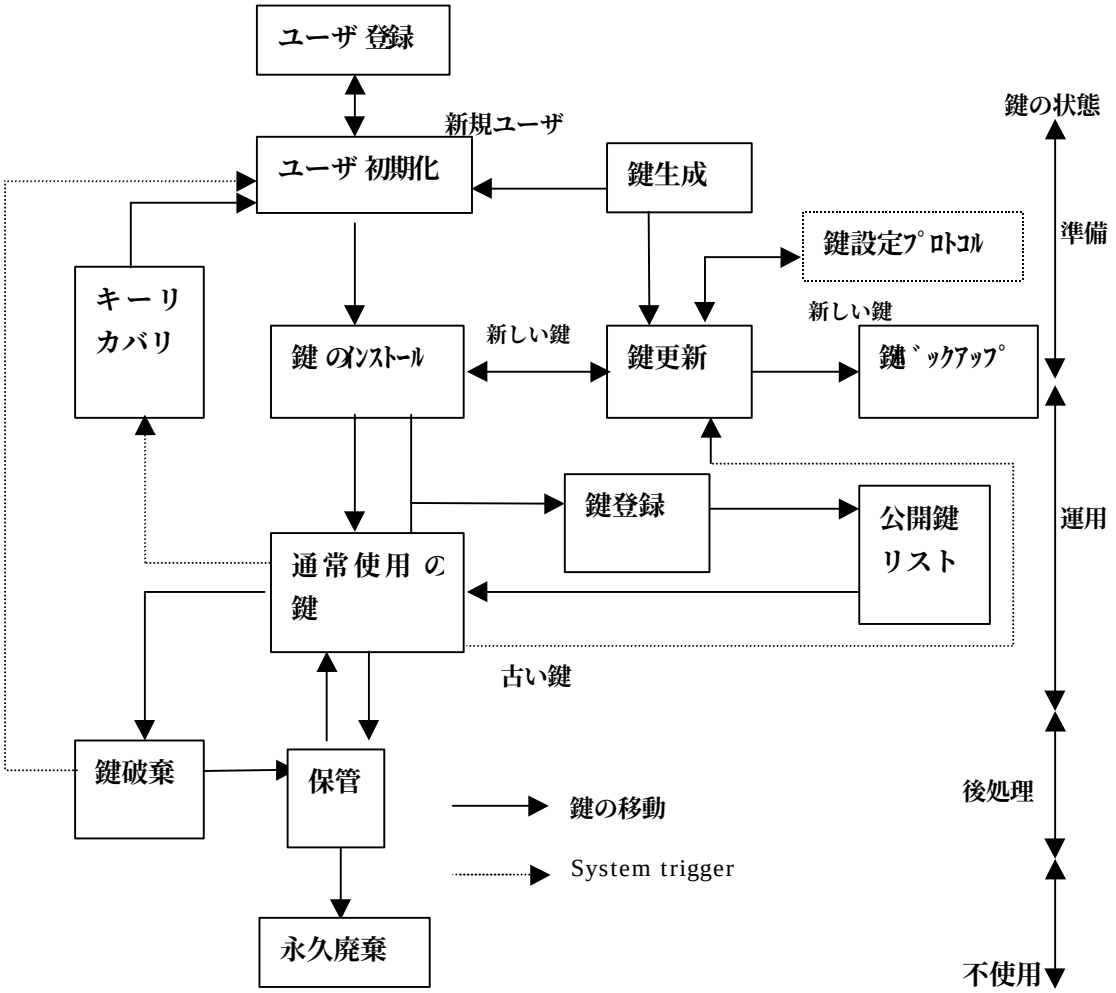


図4.2-1　鍵の ライフサイクル

(1) **ユーザ登録**（User Registration ）

エンティティ（企業・個人を含む使用者）はこの登録により認証される。これはパスワードやセキュアなＰＩＮ（個人識別子）、ワンタイム技術などを用い、初期証明書の取得が行われる。

(2) **ユーザ初期化**（User Initialization ）

エンティティはユーザ登録時に得た初期証明書の使用やインストレーションを含み、暗号アプリケーションを初期化する。

(3) **鍵生成**（Key Generation ）

暗号化鍵生成はエンティティが自身の鍵を生成するか、信用された機関から鍵を得ることが望ましい。

(4) **鍵インストール**（key Installation ）

証明書は使用するユーザによりエンティティのソフト・ハードウェア内に、以下の技術などによりインストールされる。

パスワードまたはＰＩＮの手動登録、ディスクの配布、書込専用デバイス、ＩＣカードまたは他のハードウェアトークンかデバイス（例えばキーローダー）。

(5) **鍵登録**（key Registration ）

鍵インストールに関して、証明書はエンティティを識別する名前として公式に記録される。

公開鍵方式の場合、公開鍵証明書は認証局で作られる。

(6) **通常使用**（normal Use ）

ライフサイクルの目標は暗号の使用に対して証明書の運用操作を容易にすることであり、この状態は暗号使用期限まで続けられる。また公開鍵は暗号化に使用しなくなったとしても、秘密鍵は復号に（通常）使うために残される。

(7) **鍵バックアップ**（key Backup ）

証明書をバックアップするセキュアな保管メディアは、キーリカバリのためのデータソースを供給することが望ましく、またバックアップは運用使用のために短期間保管となる。

(8) **鍵更新**（key Update ）

暗号使用期限が終了すると、証明書は新たなものに置き換えられる。

(9) **保管**（archival ）

使用しない証明書は、否認を含む論争の解決などで必要となる鍵回復のために、特別な環境の元で保存される。

(10) **鍵登録解除と鍵消去**（key De-registration and destruction）

エンティティに関連した鍵や証明書の要求が長期間なかった場合、鍵に関するのすべての公開記録から鍵は登録を解除され、すべての鍵のコピーは消去される。

(11) **キーリカバリ**（key Recovery ）

証明書は鍵の漏洩と関係なく紛失した場合に、バックアップコピーから証明書を復元できる。

(12) **鍵破棄**（key Revocation ）

鍵の漏洩を含めて、使用している鍵の期限終了には、使用する運用鍵を取り除く必要がある。

4.3 鍵の配送

4.3.1 鍵の 配送方法

秘匿通信を行うとき、文書などのデータを暗号化して送信するが、このとき暗号化には暗号化速度の優位性から主に共通鍵暗号アルゴリズムが用いられる。図 4.3-1　鍵の配送の図のように共通鍵暗号アルゴリズムは、データの暗号化と復号に同じ鍵を用いる。

データは暗号化されて送信されるが、この鍵を受信者にどのように知らせるかが問題となる。第３者に知られないように Kab を共有する必要がある。

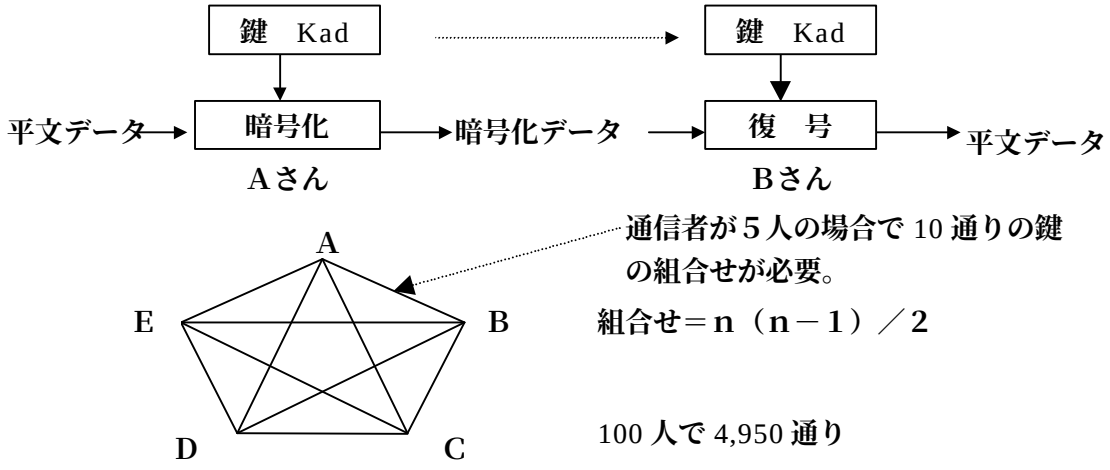


図4.3-1　鍵の 配送

したがって、インターネットのような大規模ネットワークでの鍵配送は非常に手間のかかるものとなる。

4.3.2 鍵配送アルゴリズム

鍵配送を行うために、主に下に示すような数種類の鍵配送アルゴリズムが使用されている。

① 共通鍵の事前配送

② Kerberos（時刻印を用いた共通鍵方式）による鍵配送

- 公開鍵系暗号アルゴリズムを使用

③ ＲＳＡ（1977 年に Rivest、Shamir、Adleman によって開発された暗号）による暗号化鍵共有

④ ElGamal（1982 年に ElGamal によって開発された公開鍵暗号）による暗号化鍵共有

- 直接共通鍵を生成する

⑤ Diffie-Hellman（1976 年に Diffie、Hellman によって開発された暗号）による共通鍵の生成

⑥ ＫＰＳ(Key Predistribution System、1986 年に松本・今井によって開発された暗号鍵共有方式)による共通鍵の生成

⑦ その他　楕円暗号、ＭＴＩ（松本、高島、今井）

4.3.3 共通鍵暗号系暗号アルゴリズムによる 暗号鍵共有

予め通信相手との共通鍵を鍵管理帳として設定し、各通信相手へ安全な経路を使用して配送する。また平文を暗号化するにはセッション鍵（作業鍵）を共通鍵暗号アルゴリズムを使用し暗号化して配送する。図 4.3-2 に鍵管理帳の図を示す。

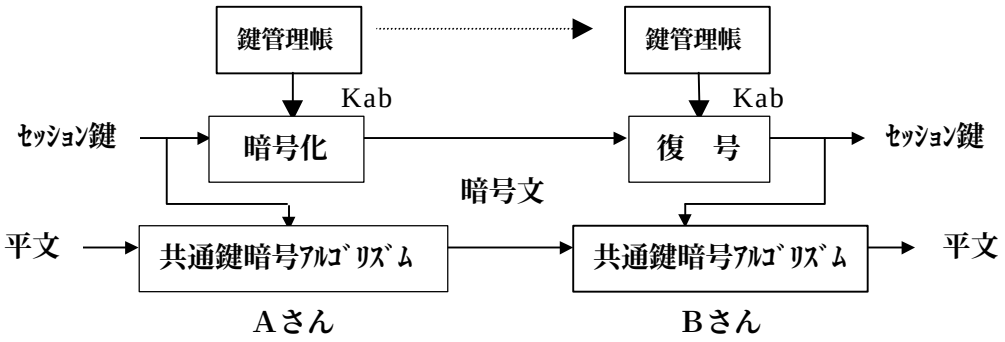


図4.3-2　鍵管理帳

(1) kerberos

共通鍵暗号ではいかに送信者と受信者で秘密の鍵を配送するかが大きな問題であった。この鍵配送の問題を見事に解決したのが公開鍵暗号である。しかし鍵配送の問題は、公開鍵暗号を用いる方法以外にもいろいろな解決策が提案されている。その一つが、秘密鍵暗号に基づき、鍵配送センター（サーバ）を用いた方式の Kerberos である。

ここで、Kerberos のプロトコルを簡単に説明する。

エンテイテイＡが鍵配送センターＳを通じて、エンテイテイＢと鍵 Kab を共有しようとする。

ここで、Ａ、Ｂは、Ｓと事前に秘密鍵暗号の鍵K_A、K_Bを それぞれ共有しているものとし$E_K(X)$は鍵Ｋによる平文Ｘの暗号文を意味する。

① Ａは、Ｓに（ID_A、ID_B）を送る。ここで、ID_A、ID_Bは個々の公開されている情報である。

② Ｓは、ＡにE_{K_A}（Ｔ、Kab、ID_B）、E_{K_B}（Ｔ、Kab、ID_A）を送る。ここで、Ｔは時刻印である。

③ Ａは、ＢにE_{Kab}（ID_A、Ｔ）、E_{K_B}（Ｔ、Kab、ID_A）を送る。

④ Ｂは、Kab をＡとの共通鍵とする。Ｂは、ＡにE_{Kab}（Ｔ＋１）を送る。

⑤ Ａは、正しくＴ＋１を復号できれば、Kab をＢとの共通鍵とする。

Kerbers では、時刻印を用いるため、ネットワークの各装置の持つ時刻が設定される必要があり、運用上かなりの負担になる可能性がある。

4.3.4 公開鍵暗号系暗号アルゴリズムによる 暗号化鍵共有

公開鍵暗号系暗号アルゴリズムを使用して、暗号化／復号のときに使用されるセッション鍵（作業鍵）を暗号化して配送する。

図 4.3-3 に公開鍵暗号系暗号アルゴリズムによる暗号化鍵共有の図を示す。

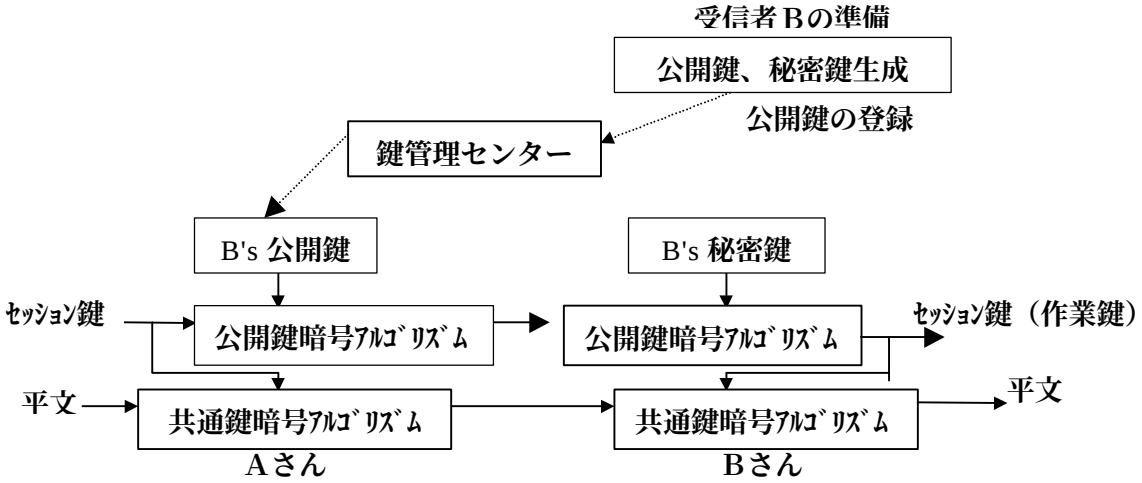


図4.3-3　公開鍵暗号系暗号 アルゴリズムに　よる暗号 化鍵共有

ＲＳＡなどの公開鍵暗号アルゴリズムを使用して、セッション鍵を相手の公開鍵で暗号化して相手に送り、受信者は自分の秘密鍵で暗号化されたセッション鍵を復号してもとのセッション鍵として、鍵共有を行う（詳細は 2.3 の公開鍵暗号を参照）。

4.3.5 共通鍵の 生成

図 4.3-4 に示すように、ユーザA、Bで公開鍵、秘密鍵を生成し相手の公開鍵を鍵管理センタ経由、または通信相手から直接入手し共通鍵を生成する。

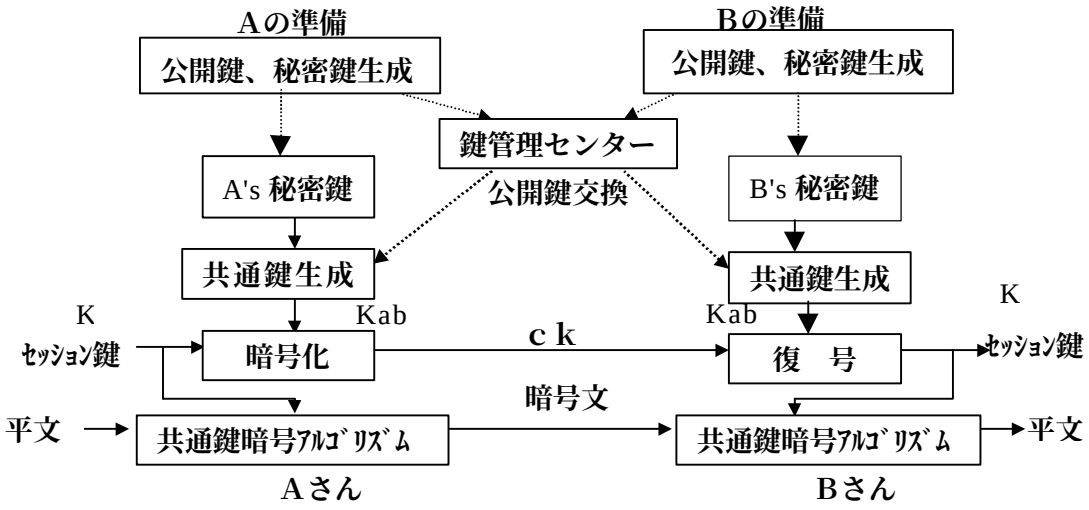


図4.3-4　共通鍵の 生成

① ＤＨ(iffie-Hellman)による共通鍵生成（図 .3-4 参照）

通信者が公開鍵、秘密鍵を保有した準備を行い（またはすでに実装している）、通信相手同士に自分の公開鍵を交換し、共通鍵を生成する。

準備：素数ｐ、原始元ｇをA、Bとで予め共有する（秘密ではない）。

送信者Aはランダムな数Xa を秘密鍵として保有する。

受信者Ｂはランダムな数Ｘｂを秘密鍵として保有する。

暗号通信：送信者Ａは$y_a = g^{Xa} \bmod p$ を計算し、Ｂに送信する。

受信者Ｂは$y_b = g^{Xb} \bmod p$ を計算し、Ａに送信する。

送信者Ａは $Kab = y_b^{Xa} \bmod p$ を計算し、セッション鍵（ｋ）を生成して平文をｋで暗号化し、ｋを Kab で暗号化してＢに送信する。

受信者Ｂは $Kab = y_a^{Xb} \bmod p$ を計算し、ｃｋを復号しセッション鍵ｋを得て、さらに暗号文をｋで復号し平文を得る。

② ＫＰＳを使用した場合（図 .3-5　参照）

図 4.3-5 のＫＰＳを使用した鍵共有の図を示す。

準備：鍵管理センターがランダムなある大きさのＧを保有し、Ａ、Ｂの識別子（電子メールアドレスなど）の情報y_a、 y_b、とＧからそれぞれの秘密鍵（秘密アルゴリズム）Ｘａ＝ y_a・f1・（Ｇ）、Ｘｂ＝ y_b・f1・（Ｇ）を計算し、Ａ、Ｂそれぞれに予め配布する。

（f1 は一方向性関数で、１：ｎにデータを拡大する。）

暗号通信：送信者Ａは、Ｂの識別子を入手し（電子メールでは送られて来る）、Kab＝f2 {y_b・f1・（Ｘａ）}を計算しセッション鍵ｋを生成して平文をｋで暗号化し、ｋを Kab で暗号化してＢに送信する。

（f2 は一方向性関数で１：１）

受信者Ｂは、Ａの識別子から Kab＝ ｆ２{y_a・f1・（Ｘｂ）}を計算し、ｃｋを復号しセッション鍵Ｍを得て、暗号文をｋで復号し平文を得る。

ＫＰＳは、公開情報に電子メールアドレスや名前などの通信相手を識別しやすい情報を使えるので、鍵管理センターへ相手の公開鍵を取りに行く必要もなく、また通信相手と公開鍵の対比リストなども必要ないため、手間がかからない。

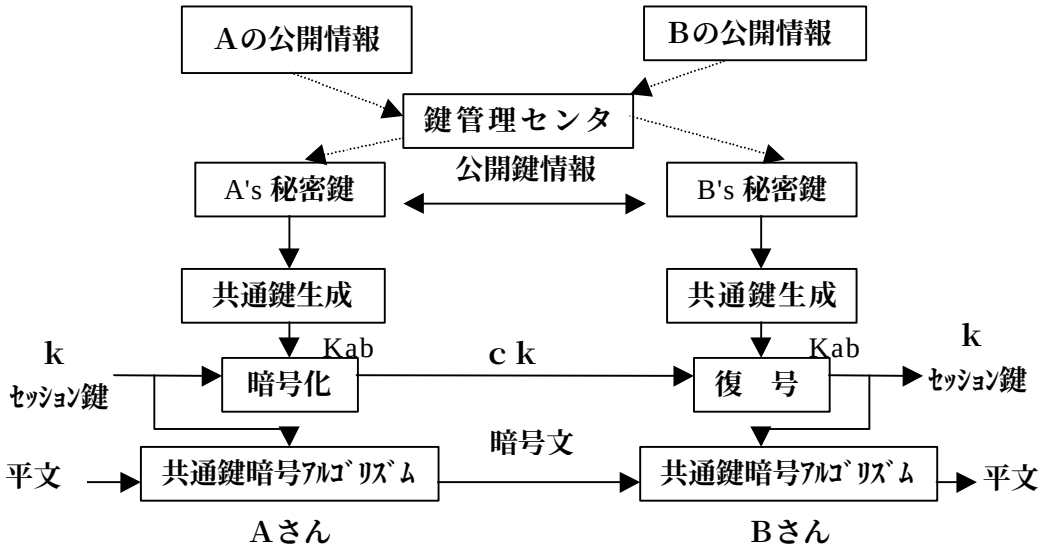


図4.3-5　KPSを使用した 鍵共有

4.3.6 留意点など

鍵管理は、システム全体の運用、仕様によりその方法を選択して使用する、また鍵管理を行う場合においても、秘密情報を使用するのでそれらの保管方法についても十分考慮する必要がある。

参考文献：[1] 今井秀樹：「暗号のおはなし」、1993 年、財団法人日本規格協会、
[2] B.Schneier,“Applide Cryptography 2nd edition”, John Wiley&Sons,
[3] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone,
[4] “Applied Cryptography”, CRC Press

4.4 鍵の保管

鍵の保管には、システム設計で一般的に要求される性質（信頼性、経済性、高速性 etc)の他に **耐タンパー性**(それが持つ情報を不正に読んだり、改ざんしたりするのが難しい性質）が重要である。

4.4.1 物理的隔離

鍵の保管場所を物理的に隔離することで耐タンパー性を確保する。バイオメトリックス（指紋、網膜パターン、虹彩パターン etc）と警備員により厳重な入退室管理が行われている部屋に設置された複数の管理者と鍵でロックされた金庫の中に保管することから、一般事務室の鍵付ロッカーへの保管まで色々なレベルがある。

4.4.2 分散保管

鍵の単純な分割、乱数との排他的論理和による分割、暗号化して鍵と別媒体に分離、特殊な関数による分割などで分割した鍵を複数の管理者または記録媒体に分散して保管することにより耐タンパー性を高める。運用面を含めた耐タンパー性の向上が期待できる。

4.4.3 特殊媒体への 保管

(1) **磁気カード**

深層記録、特殊な形状のヘッドによる記録、特殊な信号規格での記録、信号の揺らぎの利用、高磁力記録などがある。

(2) **ＩＣカード**

単なるメモリカードとＣＰＵを内蔵して特殊なプロトコルでやりとりするカードに分類できる。単なるメモリカードでは高い耐タンパー性を期待できない。ＣＰＵを内蔵カードの場合は、媒体としての性質でなくそのシステム設計に依存する。他に、非接触型・接触型、電源内蔵・無電源など様々な分類方法がある。

いずれにしても媒体の特殊性が保たれている間だけの耐タンパー性であり、不正な運用・盗難・ロット余り品・破棄品などでリードライト機器や媒体が市場に出回らないよう、不正な管理が無いように配慮する必要がある。

4.4.4 特殊記録フォーマットでの 保管

一般的なフロッピィディスクであっても、ＯＳ／ＢＩＯＳで扱えないような特殊な記録方式で記録することで耐タンパー性を高める方法がある。特殊なトラック形式（多数のレコードＩＤ、巨大レコード、極小レコード、単密倍密の混在、異常レコード、etc）、オーバートラック記録、アナログ的な不安定性の導入、媒体に付けた傷など。かつては、高価なビジネスソフトやゲームソフトのコピープロテクトとして盛んに使われた。

しかし、リバースエンジニアリングに耐えられない、ハードウェアが限定される、動作が不安定であるなどの問題点がありＥＣ関連での採用例は少ない。

4.4.5 暗号技術による 保管

パスワードの保管など一致か否かの参照のみで良ければハッシュ化して保管し、暗号化鍵の保管など書いた内容を読む必要が有れば暗号化して記録することで耐タンパー性を確保する。

一例として、ハッシュ化・暗号化の鍵として真性乱数の累積値を使用し、アクセスの都度再ハッシュ化・再暗号化するにより、予測不能な変化の継続で耐タンパー性を高める事が出来る。

4.5 鍵の更新

4.5.1 作業鍵の 更新

作業鍵とは、文字どおり対象データを暗号化するだけに利用される鍵であり、通常は共通鍵暗号方式が利用され、１回限りもしくは一定限度内（回数、期間）で更新されることが前提である。

新たな鍵に更新する際には、鍵の生成や、共有のための手続き（配送や共通鍵の生成）が必要となる。鍵の生成においては、万一それまでの鍵が解読された場合でも新たな鍵との関連性が無いように、十分性質の良い乱数を用いる必要がある（2.5 参照）。もちろん、暗号アルゴリズム毎に理論的に強度の低い鍵は使用を避けるべきである。鍵の配送と共通鍵の生成に関しては 4.3 を参照のこと。

なお、鍵の配送手段としては通信を用いるほか、人手を介して配送することが考えられる。この場合は通信よりも安全が保ちやすいが、以下のような点などで人間のセキュリティレベルに注意が必要である。

① ８文字程度の短い鍵は電話連絡で済ますことも可能であるが、盗聴や、声を聞かれてしまう可能性がある

② 文書による受け渡しでは、配送中の紛失や使用後の廃棄を厳重にしなければならない

③ 電子メールなど、盗聴されやすいメディアは避けるべきである

④ フロッピィや磁気テープなどの可搬媒体を使用する場合は、途中で複写される可能性があるので直接の手渡しが必要である

⑤ 同様に使用後の可搬媒体はデータを消去の上物理的に読めないようにデータの上書きなどを行う

また、配送時の安全性を高めるために、鍵の階層構造を取ることがある。すなわち、平文を暗号化するための作業鍵を、上位の暗号鍵１で暗号化し、更にその上位の鍵で暗号化し、・・・ということを繰り返し、最終的にはマスター鍵で暗号化する。

この方式の優位点は、

① マスター鍵以外に共通鍵方式を用いる場合は、公開鍵方式による交換に比べ受け渡しするデータ量が少なくて済む

公開鍵：数百ビット（数十バイト）、共通鍵：～百数十ビット

② 共通鍵を n 回まで利用できるのであれば、作業鍵は n 回毎に更新が必要だが暗号化鍵１は n^2 回毎、・・・というように交換回数は少なくて良いといった点で

ある。

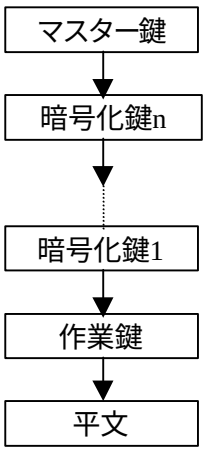


図4.5-1　鍵の 階層構造

4.5.2 マスター 鍵の 更新

マスター鍵も、作業鍵と同様にある一定限度内での使用にとどめ、期限がくれば更新しなければならない。例外的に、認証局のルート鍵だけは証明書が全て無効となるため更新は行わない。ただし、今後相互認証の仕組みが整備され、相互認証を行っているグループ全体でルートの役割を果たせば、鍵の更新も可能である。一般的には、共通鍵暗号方式によるマスター鍵は利用一定回数以内で更新が必要である。また、公開鍵の場合には、鍵のビット長と、その鍵の利用目的により認証局側で有効期限を設定する。たとえば、エンドユーザの個人鍵は一般には２年程度である。

マスター鍵の更新時の配送は、上位の鍵による方法が不可能なため、主に物理的な手段が用いられるので、上記のような点に注意を払う必要がある。仮に、それまで使用していた鍵を使って暗号化して配送してしまうと、危険度の高くなった古い鍵がいずれ解読された場合、その鍵を使って次の鍵を求め、・・・と直ちに現在使用中のマスター鍵が求められてしまうからである。マスター鍵は通常、鍵管理システムで廃棄鍵として保管される。公開鍵暗号方式での廃棄鍵の運用は 4.7 に述べる。

4.6 鍵管理システム

鍵管理システムは、一般には暗号鍵（共通鍵、秘密鍵）を管理し、攻撃以外の各種の障害（ハードウェアやソフトウェア、事故／天災／人為的ミス／不正行為）などの場合に、暗号化されているデータを復号可能にするための、インフラストラクチャである。

米国で提唱されている鍵回復システム（Key Recovery System）では（政府を含む）第３者機関による鍵の管理が提唱されているが、ここでは第３者機関に限らず暗号機能を利用するシステムの構成要素全般に対する鍵の管理と回復のためのシステムをいう。

公開鍵インフラストラクチャを利用する場合には、共通鍵の配付や管理に公開鍵暗号を利用することで、このような鍵管理システム自体が不要に思える。しかし、この場合鍵が失われたことを証明することはできるが、失われた鍵（およびそれにより暗号化された情報）は決して戻らない。暗号化した情報を失わないためには、公開鍵による管理とともに、

このような鍵を回復するためのインフラも欠くことが出来ない存在なのである。

そしてなによりも、公開鍵暗号系による鍵管理を行うには信頼できる認証局が必須であり、認証局の信頼度（暗号理論面、実装面、処理能力やフォールトトレランス性など）が鍵管理システムの信頼度よりも高くなければならない。

4.6.1 鍵管理システムの 必要性

暗号通信のように、仮に暗号鍵が失われたりした場合に、新たな鍵の生成と登録を行うだけで機能が回復でき、過去のデータが残っていなければ、鍵管理システムが無くても運用上の支障にはならない。しかし、暗号化によるデータの保存の場合は、鍵が失われるとその情報が利用不能となる。そのために、重要なデータの保存においては、暗号化に使用した鍵も何らかの手段でバックアップ／保存をしておかなければならない。

たとえば、会社などで個人にそれぞれ鍵を配付（または登録）し利用する場合、その従業員が万一死亡したり、過失やあるいは悪意を持って鍵を故意に紛失した場合、組織上の上位者がそれらを復号できなければ会社としての業務に支障が出るであろうし、場合によってはとんでもない損害を受ける可能性もある。

また、サーバや端末上のプログラムがそれぞれ認証や履歴データの保存などを行う場合に使用する暗号鍵などは、当然その機器内部で暗号化などの手法を用いて保管しなければならない。この時に用いる鍵が、全てのサーバや端末で同じであった場合、万一盗難や解読などが発生すると、全体が危険にさらされるため、通常は異なった鍵を利用しなければならない。しかし、システムの更新や履歴の解析などの際には、これらの鍵を利用する必要があるので、サーバや端末以外の場所に、これらの鍵のバックアップを取っておかなければならない。

なお、マスター鍵となる秘密鍵や共通鍵が、何らかの手段により解読もしくは漏洩してしまった場合、新たな秘密鍵や共通鍵を生成し運用を引き継ぎ、廃棄されたマスター鍵を用いて暗号化されたデータは復号し、新たな鍵で暗号化を行わなければ危険である。この場合、秘密情報でないものに電子署名がなされて、デジタルタイムスタンプが付加されている場合は（たとえば契約書など）、再度署名をし直すとその効力が失われるため、そのままにしておく必要がある。

4.6.2 鍵管理システムの 種類

一般的には、鍵を管理するスキームとして集中方式と階層方式、分散方式がある。

① 集中方式

ある一つのマスター鍵（もしくは特権）により全ての鍵を管理（または暗号化）する方式。管理システムを一個所に閉じ込めることができるので、物理的なセキュリティは保ちやすい反面、地理的に分散した場所に置けないデメリットがある。また、万一マスター鍵や特権レベルが突破されると、全てのシステムに脅威が及ぶ。

② 階層方式

鍵管理を階層化し、各階層の鍵はその上位層の鍵により管理（または暗号化）される方式。保管する場所は一個所でも良いし、場合によっては地理的に分散させることも可能である。階層を設定する場合も組織階層を流用することができ、その場

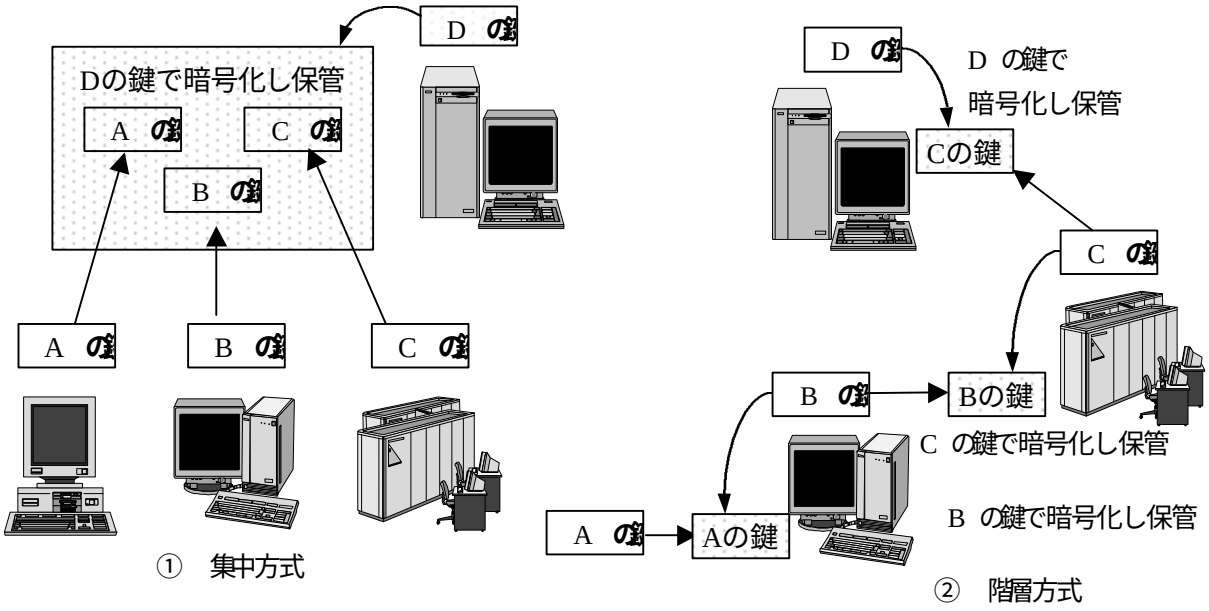
合にはたとえば、ある支社で用いられる鍵をその支社長の鍵で管理することが可能である。

地理的に分散させることで、下位レベルの管理鍵や特権レベルが破られても影響はその配下にしか及ばないというメリットがある。ただし、分散したそれぞれの場所で物理的なセキュリティを保つ必要がある。

③　　分散方式

①や②の方式で、管理する側の鍵を複数個組み合わせないと、鍵の管理や復号ができない方式。管理すべき鍵を分割して、別々に暗号化する場合と、別々の鍵で何重にも暗号化する方式が考えられる。この方式のメリットは、最上位層で危険分散が可能なため、攻撃や特権レベル所有者の作為による漏洩などが困難な点である。

実際に、鍵回復システムやＴＴＰ（信頼される第 3 者機関）などによる鍵の保管では、このような分散方式が利用されている。実際に鍵管理システムとして商品化されている物も、上記各方式およびその組み合わせに対応しているものが多い。



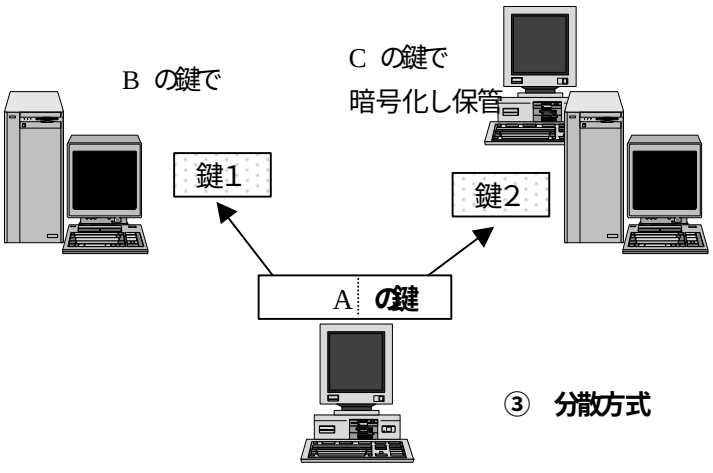


図4.6-1　鍵管理システムの手法

4.6.3 鍵管理システムのセキュリティ

鍵管理システムを構築・運用する場合は、これらがシステム上の最大の攻撃目標になることを前提にセキュリティを保つための工夫が不可欠である。

認証局は、公開することが前提なので、保持している情報が不正に書き換えられない限りはシステムの安全性は保たれる。一方、鍵管理システムでは、保管している内容が漏洩することがあってはならない。したがって、ネットワーク接続などの危険性を極力抑えた上で、安全な場所に隔離することが望ましい。また、公開鍵システムでは、本来秘密鍵（Private Key）は配送など行わず、本人だけが利用できることが前提であるが、鍵管理システムで管理するためには配送手段を用意しなければならなくなる。このことは、公開鍵システムの持つ本来のメリットを損ねることになるので、適用すべきかどうかを十分検討する必要がある。

最後に、特に鍵管理システムで個々人の使用する鍵を管理することは、プライバシーの問題を常にはらんでいる点に注意が必要である。もしあなたの上司が、業務上の代行処理をするためにあなたの鍵を管理しているとしたら、あなたが個人的に秘密にしている情報までも上司から見られることになる。本当にプライバシーの必要な場合は、業務で利用する場合の鍵とは別に、自分だけが知っている秘密の鍵を用いて暗号化しなければならない。このような注意点を利用者が意識しなければならないとしたら、その暗号システムは十分に活用されずに終わるであろう。

したがって、鍵管理システムの運用、特に鍵の回復時の運用について、あらかじめ十分な制度検討と説明および利用者の理解が不可欠である。できれば、鍵の回復にあたっては①本人の了解（もしくは事前承諾）が必要、②複数の人間による合意が必要、③回復した鍵による復号は（できれば）本人立ち会いで公開の場で行われるべきであること、④明確な理由なく本人が許可した物以外復号しない、⑤プライバシーを侵害した場合のペナルティ、などを運用ルールとして決めておくことが重要である。

4.7 公開鍵の廃棄

4.7.1 鍵の 寿命と 廃棄

公開鍵暗号では、一旦生成した鍵を比較的長期間利用することになる。このため鍵の寿命と鍵の廃棄についていくつかの注意を払う必要がある。

- 鍵の寿命について
  暗号解読者による長期間にわたる暗号解読に対抗するため、鍵のサイズと有効期限を適切に設定し、鍵を定期的に更新すべきである。有効期限は、予想される解読期間（2.3.7 参照）よりも短く設定し、有効期限内に解読される危険性を極力小さくする。
- 鍵の廃棄について
  鍵の有効期限内において、鍵の紛失や他人への漏洩が認められた場合や、鍵の所有者の移動などにより鍵が不要になった場合に、それらの鍵を廃棄し、それらの鍵が使われても無効であることを全ユーザに周知するしくみが必要である。このための手段としては、例えば X.509[1]で規定されたＣＲＬ（Certificate Revocation List：証明書失効リスト）と呼ばれる廃棄鍵のリストを利用する。

4.7.2 廃棄鍵のリストの 管理と 運用

公開鍵暗号を利用する場合、鍵が有効なものかどうかを知ることは非常に重要であり、全利用者が、リアルタイムに廃棄鍵のリストを参照できることが理想である。しかし、現実には、利用者の多い大規模ネットワーク環境での利用や、ＩＣカードなどオフラインで鍵が利用される場合など、廃棄鍵のリストを管理・運用する上での課題は多い。以下に、廃棄鍵のリストの管理運用上の課題について列挙する。

- 大規模ネットワーク環境での利用においては、廃棄鍵のリストを１個所で集中管理することは、リストが膨大となり、負荷が集中して非効率的である。
  廃棄鍵のリストを分散管理する場合、各廃棄鍵のリストを同期を取って更新する必要がある。
- オフライン環境での利用においては、廃棄鍵のリストを参照するしくみが必要である。
- 廃棄鍵のリストの運用管理には相応のコストがかかる。利用者の規模、対象データの重要度、ネットワーク環境など、局面に応じた運用管理を行う必要がある。

参考文献：[1]　CCITT,”The Directry-Authentication Framework”,X.509,1988

5 暗号に関する制度・法令

5.1 暗号の位置付けと政策問題

5.1.1 暗号に 関する 政策の 項目

(1) 暗号の利用と貿易に関する政策

暗号技術は、主に軍事や外交の分野において発展してきたという歴史的経緯があるため、その輸出に一定の規制をおぼしている国が多い。また、Wassenaar Arrangementにおいては、暗号機器は汎用品リストの３段階の規制ジャンル（basic/sensitive/very sensitive）のうち、sensitive に区分されており、スーパーコンピュータと同じ区分であり、輸出の際に通報が義務付けられるものである。

なお、暗号技術の国内使用については制限していない国がほとんどである。

(2) 認証に関する政策

公開鍵方式の暗号技術が人々の信頼の下に円滑・効果的に利用されるためには、通信の相手方の真性な公開鍵を確実に入手できることが前提になる。そこで、公開鍵の真性性を証明しつつ公開鍵の管理・配信などの業務を行う、認証局（Certification Authority：ＣＡ）の存在が必要になる。

特に、公開鍵の登録の際の本人確認が厳密に行われなかったり、公開鍵の管理・配信の在り方がずさんな場合には、社会秩序が混乱するばかりでなく、なりすまし犯罪やマネーロンダリングなどの蔓延を助長する恐れもある。そこで、認証局や認証の在り方については、各国や各種国際機関において各種の検討が行われている。

(3) 鍵管理と鍵預託インフラに関する政策

キーリカバリ（鍵の回復）機能を確保しておくことは、電子商取引を始めとする様々な場面において、鍵の紛失や事故などによる消失の際のトラブルを解決するために大変重要である。しかしながら、このキーリカバリ機能については、個人のプライバシーとの整合性に係る問題点も指摘されており、この点についても配慮しながら検討を進める必要がある。

暗号の不正利用による犯罪が発生した場合に実効的な対処手段を確保するためには、暗号化されたデータへの合法的なアクセスを可能にしておくこと、言い換えれば、社会システム上、キーリカバリ機能を確保しておくことの提案がある。

5.1.2 国際社会で の 議論の 動向

(1) Ｇ７／Ｐ８テロ関係閣僚会議

1996 年７月に開催されたＧ７／Ｐ８のテロ関係閣僚会合で、テロリズム対策に関する合意文書を発表し、この中で、暗号について以下のように述べている。

「合法的な通信のプライバシーを保護しつつ、テロ行為の抑止・捜査のために必要な場合に、政府によるデータおよび通信への合法的アクセスを可能にする暗号技術の

使用に関する協議を、適当な二国間または多国間のフォーラムにおいて促進することを全ての国に要求する」

(2) **ＯＥＣＤ暗号政策ガイドライン**

ＯＥＣＤ（経済協力開発機構）は 1995 年 12 月に、最初の会議「暗号方針に関するＯＥＣＤ専門家会議」を開催した。この会議で、国際的な協調が必要であること、またプライバシー保護と公共安全のための「法に基づく入手」のバランスに関して統一的な解決策が必要であることが確認された。暗号機能を適用するためのガイドラインの最初の原案は、国際商工会議所とＯＥＣＤの民間諮問委員会であるＢＩＡＣが作成した。

最終案は 1997 年３月にＯＥＣＤ委員会で承認され「ＯＥＣＤ暗号政策ガイドライン」として公開された。このガイドラインは、８原則からなり、プライバシー保護との整合性をとりつつも、暗号化されたデータを国家などの第３者が強制的に解読することを認めたものである。

(3) **ＥＵ（欧州連合）**

欧州連合閣僚理事会においては、1995 年９月、加盟国にたいする「情報技術に関連する刑事訴訟法の問題に関する」勧告を採択し、その中で、暗号の使用に関して次のような内容の勧告を行った。「暗号の合法的な使用への（抑止的な）影響を最小限にとどめつつ、暗号の（不正）利用が犯罪捜査におぼす否定的な影響を最小にするための手段を考えるべきである。

また、欧州委員会（Europe Commission）においては、ＥＴＳ（Europe-wide Network of Trusted Third Parties Services：ヨーロッパに地域おけるＴＴＰサービスのネットワークシステム）の調査研究、およびデジタル署名の法的側面に関する調査研究をそれぞれ約１年をかけて進めている。

(4) **ＩＳＯ（国際標準化機構）**

技術の標準化について、ＩＳＯ／ＩＥＣ ＪＴＣ１／ＳＣ２７（セキュリティ技術）において議論が行われており、ＴＴＰ（Trusted Third Party：信頼される第３者機関）、暗号技術、セキュリティ評価について、それぞれＷＧ１、ＷＧ２、ＷＧ３において検討されている。また暗号アルゴリズムの登録制度があり、多数の暗号が登録されている。

(5) **ＩＴＵ（国際電気通信連合）**

技術的側面について、電気通信に係る国際標準化を行う国連機関であるＩＴＵにおいて検討がすすめられている。ここでは、認証に係る標準フォーマットを提供しており、デジタル署名技術の標準としてはＩＴＵ／ＩＳＯ／ＩＥＣの「X.509 勧告」がある。

5.2 ＯＥＣＤガイドラインと注意点

5.2.1 ＯＥＣＤガイドラインの８ 原則

(1) 暗号手法に対する信頼

暗号手法は、利用者が情報システムや通信システムに適用する際に秘匿性が確保できるように、信頼性の高いものでなければならない。

(2) 暗号手法の選択

利用者は適用される法律のもとで、利用する暗号手法を自由に選択できなければならない。

(3) 市場主導の暗号手法の開発

暗号手法の開発は個々の利用者や産業界や政府からの要請にこたえるものでなければならない。

(4) 暗号手法に関する諸標準

暗号手法に関する技術標準や基準やプロトコルは国家および国際的レベルで開発その適用を推進していかなければならない。

(5) プライバシーおよび個人データ保護

通信の秘密や個人データの保護を含むプライバシーに関する個人の基本的権利は国家の暗号方針の策定や暗号手法の導入と運用において十分尊重されなければならない。

(6) 合法的アクセス

国家は暗号政策により、暗号化されたデータの平文、または復号のための鍵への合法的なアクセスを認めることができる。この政策実施にあたっては、本ガイドラインの他の原則を十分配慮しなければならない。

(7) 責任

暗号サービスを提供するか、鍵を保管または利用する個人または組織に関して契約書が作成される場合は、その責任について契約書または規則に明記しなければならない。

(8) 国際協力

政府は暗号政策を遂行するために国際協力をしなければならない。この一環として、政府は貿易を阻害するような暗号政策を制定してはならないし、そうした暗号政策が存在する場合には、除去しなければならない。

5.2.2 法に 基づく 復号の 問題点と 動向

(1) 問題点と動向

暗号の不正利用による犯罪に対処し公共の安全を保障するために、実効的な対処手段を確保するためには、「法に基づく入手」が必要となる。また一方、一般消費者を含む利用者が安全だと確信できるセキュリティ対策やプライバシー保護対策が益々重要となってくる。これらの二つのバランスに関して統一的な解決策が必要であることが確認された意義が大きい。今後、各国においてガイドラインに沿った各種制度が整

備されていくものと思われる。

ただ、具体化にあたっては「法に基づく入手」での情報を保管する鍵管理機関の運用には、監視などを含めて十分な注意が必要である。

(2) **その他の問題点**

企業などの組織が分割・合併・倒産などになった場合、署名の証跡の管理は難しい問題である。例えば、ある企業の一つの事業部門のみが他の会社に分割・譲渡された場合、新しい会社で過去の事業経過をトレースするために過去の証跡を確かめることが必要である。この証跡の確かめが可能となるためには、署名のための鍵管理が新しい会社に移管出来ることが必要である。このための法的な環境の整備も必要と思われる。

5.2.3 実装担当者が 留意すべき 点

実装担当者はＯＥＣＤガイドラインの(1)暗号機能の信頼性、(2)暗号機能の自由選択、(3)市場の要求に基づく暗号機能の開発、(4)暗号機能の標準、の４項目それぞれについて、定量的で具体的な実現をはかることが求められている。

また実装担当者は、ＯＥＣＤガイドライン項目(7)責務にあるように、鍵の管理・利用についての責務を契約書に明記することが求められる。

5.3 鍵管理と鍵預託インフラ

5.3.1 鍵管理インフラ（ＫＭＩ）

認証局を社会システムの一環をなすのもとして位置付けられた、ＫＭＩ（Key Management Infrastructure）構想が、1996 年５月に米国により発表された。

ＫＭＩ構想においては、用途・機能に応じ、多種多様の認証があり得るとしているが、これら複数の認証局相互の位置付けとしては、階層構造を想定し、階層の最高位として、政府の機関であるＰＡＡ（Policy Approving Authority）を位置付けている。そして、ＰＡＡは相対的に下位の認証局（ＣＡ）の認証ないし身分保証を行い、ＰＡＡに認証された認証局はさらに相対的に下位の認証局（ＣＡ）の認証ないし身分保証を行うこととしている。

また、ここでは認証自身の身分保証の必要性を示しており、認証局の認証のありかたとしては以下の３種類になる。

- 自己保証：自分自身の身分を証明する。
- 相互保証：複数の認証局が相互に保証する。
- 階層保証：階層を形成した認証局群の中で、上位の認証局が相対的に下位の認証局の身分を保証する。

これらは必ずしも択一的なものではなく、例えば、ＰＡＡは他国のＰＡＡに相当する最高位の認証局とは相互保証を行うことが出来るようになっている。

5.3.2 電子署名法

米国では、ほとんどの州で電子署名（electronic signature：暗号による署名と電子的処

理による手書き署名）についての法的検討がなされている。ユタ州、ワシントン州などではデジタル署名（digital signature：暗号による署名）に焦点をあてている。カルフォルニア州、イリノイ州、マサチューセッツ州、フロリダ州などでは電子署名一般を対象としており、全体として電子署名に限定した立法を検討している州は少数派となっている。また、ドイツ、イタリアにおいて、電子署名法が既に制定・発効されている。

5.3.3 鍵回復（Key Recovery）／鍵預託（Key Escrow）

暗号技術の利用が一般化した場合に、犯罪などへの暗号の不正利用による弊害が発生するおそれがある。現実に暗号の不正利用による犯罪が発生した場合に実効的な対処手段を確保するためには、暗号化されたデータへの合法的なアクセスを可能にしておくこと、言い換えれば、社会システム上、キーリカバリ（鍵の回復）機能を確保しておくことの提案である。また、キーリカバリ機能を確保しておくことは、商取引を始めとする様々な場面において、鍵の紛失や事故などによる消失の際のトラブルを解決することとなる。しかし、キーリカバリ機能については、個人のプライバシーとの整合性の問題があり、この面にも配慮して検討をすすめていく必要がある。

米国においては、当初（1993 年）犯罪捜査のために政府機関に強力な権限を与えるキーエスクロー（鍵預託）構想が提案された。これは非公開の暗号アルゴリズムを組み込んだ半導体チップの通信端末への設置を義務付け、鍵の管理機関も政府機関に限定するものであった。

これに対して輸出規制緩和の観点より強い反発があり、鍵の回復機能を確保するという原則は保持しつつも、輸出規制緩和と産業振興を重点を移した、公開された暗号アルゴリズムの採用や政府関連以外の機関でも鍵管理を可能とするなどの、政策を発表している（1996 年）。これらをキーリカバリ（鍵の回復）と呼んでいる。

5.3.4 信頼される第3 者機関（Trusted 3rd Party：ＴＴＰ）

文書処理においては本人確認のための実印登録と印鑑証明の発行、文書の内容証明、文書の配達証明、日付証明など、なんらかの証明を要する行為については、市町村や郵便局という公的機関が実施している。したがって、電子化されたデータの通信においても、同様の行為が必要となる。

「信頼される第３者機関」については英国などで提案されており、以下のように規定されている。

（１） 提供すべき機能（機能要件）
（２） 自身の安全性（保証要件）
（３） 運用条件
（４） 相互認証手続き
（５） 現行法制度との関連
（６） 国際標準化

「信頼される第３者機関」が提供する最小限の機能は以下のようである。

（１） 利用者の認証
（２） 証拠保管

（３） 事象証明
（４） タイムスタンプ
（５） 鍵保管

5.4 各国の政策

5.4.1 米国

(1) 輸出・輸入・使用についての政策の現状

米国においては主として国家安全保障上の目的から、暗号機器の輸出について規制が行われており、ハイグレードの暗号技術（鍵長 40bit 以上）については、国務省の管轄下で原則として輸出が禁止されている。但し、最近では、鍵長が 56bit 未満のものについては、管轄が国務省から商務省へ変更となり、一定の条件（キーリカバリシステム）が満たされれば輸出が許可されるようになった。暗号機器に係る輸出規制の法的根拠は、以下の「武器輸出管理法（ＡＥＡＡ）」および「輸出管理法（ＥＡＡ）」である。

- 「武器輸出管理法（ＡＥＡＡ）」：軍事用途と目される品目について行われるものであり、具体的には国務省管轄下の「国際武器通商規則（ＩＴＡＲ）」の基準のもとに、「武器リスト（ＵＳＭＬ）」掲載品目について行われている。
- 「輸出管理法（ＥＡＡ）」：軍事・民事両用途に使用可能な品目について行われるものであり、具体的には商務省管轄下の「輸出管理規則（ＥＡＲ）」の基準の下に、「商業統制リスト（ＣＣＬ）」掲載品目について行われる。

(2) 政策の今後の動向

キーリカバリの持つ製品の輸出がさらに自由化される。国内の大市場向け暗号内蔵型製品については、法執行側がアクセスできるよう、キーリカバリシステムの使用が義務付けられる。

5.4.2 英国

(1) 輸出・輸入・使用についての政策の現状

輸出については厳しい規制がある。法執行側のアクセス手段として、鍵のＴＴＰの導入を検討している。

(2) 政策の今後の動向

あらゆる市販暗号製品は解読可能とするか、またはキーエスクロー扱いとする。個人の使用についてはキーエスクローを義務付ける。

5.4.3 フランス

(1) 輸出・輸入・使用についての政策の現状

暗号化には首相の認可が必要だが、ユーザがそれを受けることは事実上不可能であり、政府に鍵を寄与しなければならない。

(2) **政策の今後の動向**

変化なし。ただしローグレードの暗号の利用なら、法執行側がアクセスできる裏口を用意しておけばＯＫである。

5.4.4 オランダ

(1) **輸出・輸入・使用についての政策の現状**

個人的な通信に使う暗号の利用は自由である。ファイル暗号化できるものについては認可が必要である。

(2) **政策の今後の動向**

現状維持である。

5.4.5 ドイツ

(1) **輸出・輸入・使用についての政策の現状**

輸出にたいしては緩やかな規制である。個人暗号ソフトウェアの使用は許されている。なお、ＥＵ諸国においては、ＥＣＲ（EU Council Regulation）やＥＣＲに基づく The Decision などのＥＵにおける取り決めをベースとした規制態様があり、具体的には、ＥＵ域外への暗号機器の輸出に限り規制されていることが多い。

(2) **政策の今後の動向**

イギリス、アメリカ、フランスよりはも自由である。ドイツは商業暗号市場に注目している。

5.4.6 イタリア

(1) **輸出・輸入・使用についての政策の現状**

財務省が暗号化の記録にアクセスできるよう義務付けている法律が唯一の暗号法である。

(2) **政策の今後の動向**

郵政省のキーエスクローまたはＴＴＰがある。

5.4.7 ロシア

(1) **輸出・輸入・使用についての政策の現状**

厳しい輸出入法がある。国の許可なしに暗号を開発、製造、設置することは禁じられている。

(2) **政策の今後の動向**

個人にたいしては厳しい政策を継続するものとみられる。産業界に関しては、キーエスクローを採用すれば、やや自由化の方向でる。

5.4.8 イスラエル

(1) **輸出・輸入・使用についての政策の現状**

輸出規制ある。しかし、鍵長についての制限はない。強い暗号の使用については軍からの許可が必要など、暗号についての規制はあるが、その範囲は明確でない。

5.4.9 シンガポール

(1) 輸出・輸入・使用についての政策の現状

輸出規制はない。輸入規制はあるようである。但し、ハードウェアの暗号装置については許可が必要のようである。暗号についての法律がないので、暗号使用は合法である。

5.4.10 韓国

(1) 輸出・輸入・使用についての政策の現状

暗号装置の輸入は禁止している。

5.4.11 マレーシア

(1) 輸出・輸入・使用についての政策の現状

輸出、輸入ともに規制がない。

5.5 日本における制度・法令

5.5.1 暗号製品輸出入に 関わる 各種法令

我が国においても、暗号機器に関する輸出規制が行われており、具体的には、「外国為替および外国貿易管理法」およびその下位法令である「輸出貿易管理令」に基づき、「暗号装置またはその部分品」などを輸出しようとする者は、通商産業大臣の許可を受けなければならないこととされている。一方、暗号機器の使用および輸入については、法律上なんら制限されていない。

5.5.2 日本における 今後の 課題

電子商取引における、認証機関、電子署名、キーリカバリ、プライバシーや個人情報の保護などに関する制度・法令については、法的な枠組みが必要かどうかについて様々な議論があり、法的な対応より技術や市場による解決を第一にするべきとの議論が多い。

認証機関などの電子商取引に関する法令・制度のインフラ整備に関しては、通産省の「電子商取引環境整備研究会」（委員長：内田 貴東大教授、通販事業者、メーカ、弁護士、消費者団体などからの委員）において議論されており、平成９年 11 月に中間報告がなされている。この報告書では、認証機関や電子署名などの取引きの安全性・信頼性に関することだけではなく、プライバシー保護や有害コンテンツ対策など、広い範囲にわたってなされた議論が報告されている。また、ＥＣＯＭにおいても制度関連ワーキンググループの活動をおこない、各種の提言・報告がなされている。その他に、平成８年７月に発足した法務省の「電子商取引に関する研究会」（法律学者、法務実務家、暗号学者、通信技術者などにより構成）において、電子商取引に関する法的整備の必要性の検討に加え、電子的公証制度や電子的本人認証制度に関する検討が行われている。

個人の利益への深刻な被害に対する最小限の法的整備については、法務省、大蔵省、通産省などにおいて、「電子署名法」、「個人信用情報に関する保護法」などの具体的な法制化の動きが始まっているようである。また、これらのインフラ整備は、ＯＥＣＤガイド

ラインなども踏まえながら、国際的に協調して進めていく観点からも重要である。

5.6 知的所有権について

5.6.1 特許法、商標法などへの 配慮

最近はアルゴリズム特許（ソフトウェア特許）が成立するようになり、暗号アルゴリズム特許への抵触を注意する必要がある。さらに、特許期限が切れているものや、すでに特許申請されているがまだ未成立であったり、状況が様々であるので、よく調査して世の中の特許状況を把握しておくことが重要である。また、国によっても特許制度が違ったり、ある特許については特許申請されている国が特定されていたりするので、各国の特許状況把握への配慮も重要である。さらに、商標登録についても同様な状況把握が大切である。

5.6.2 ソフトウェアの 配付に 関する 問題

暗号以外のソフトウェアと同様に、著作権に配慮した利用・配布が必要である。さらに、暗号であることにより、安全保障に関連する各国の各種輸出規制や利用規制に抵触するおそれがあるので、要注意である。

(1) **違法コピーによる利用**

一般ソフトウェアと同様であるが、違法コピーしたソフトウェアに暗号ソフトが入っていた場合に、知らないまま違法に暗号を利用していたことになるケースもあり、注意が必要である。

(2) **自動配付システムの利用に関する問題**

米国の企業や研究所におけるサイトから自動配布システムにより暗号ソフトを入手して利用することが可能なケースがあるが、上記のような各種法律に抵触することがあるので注意をする必要がある。また、日本からの輸出の場合にも米国製の暗号が入っている場合には同様に米国の法律に抵触することになり、注意が必要である。

また暗号が応用システムに組み込まれている場合、どの暗号が入っているのかがビジブルでない場合が多く、使用されている暗号の具体的な特定が必要である。また、ソフトウェアの購入契約の中に配布個数の制限がある場合があり、これについても注意が肝要である。

参考資料：[1] 「電子商取引環境整備研究会 中間論点整理」、通産省・電子商取引環境整備研究会、平成９年 11 月。
URL: http://www.ecom.or.jp/miti/971127/

[2]「情報セキュリティ調査研究報告書」、財団法人社会安全研究財団・情報セキュリティ調査研究委員会、平成９年４月１日。
URL: http://www.npa.go.jp/seiankis1/title.htm.

[3] 田渕治樹：「ＯＥＣＤガイドラインの概要とその影響」、NIKKEI COMPUTER 1997．5．26.

[4] The OECD Guidelines on Cryptography Policy,

URL: http://www.oecd.org/dsti/iccp/crypto_e.html
[5] 「世界暗号政策ツアー」、WIRED JUNE 1997.
[6] 「暗号化技術の輸出規制に関する FAQ」、
URL: http://www.rsa-japan.co.jp/faq/export.text
[7] Encryption Policy and Market Trends,
URL: http://guru.cosc.georgetown.edu/~denning/crypto/Trends.html
[8] Crypto Law Survey,
URL: http//cwis.kub.nl/~frw/people/koops/lawsurvy.htm
[9] The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption,
URL: http://www.crypto.com/key_study/

6 付録Ａ．検討メンバーリスト

ＥＣＯＭ

五味　俊夫　主査　　電子商取引実証推進協議会　主席研究員
辻　　秀一　副主査　電子商取引実証推進協議会　主席研究員
菅　　知之　副主査　電子商取引実証推進協議会　主席研究員
米倉　昭利　副主査　電子商取引実証推進協議会　主席研究員

リーダー・サブリーダー

青木　　康　リーダー　　㈱コンテック　第２開発・セキュリティ事業部
　　マーケティング課　係長
渡辺晋一郎　サブリーダー　㈱アドバンス　情報通信事業本部　技術部　統括課長

メンバー

芦原　憲司　㈱シー・アイ・シー　電子取引研究プロジェクトチーム　サブリーダー
船戸　秀紀　昌栄印刷㈱　生産本部　技術部
沼尾　雅之　日本アイ・ビ・エム㈱　東京基礎研究所　ネットワーク＆ソリューションテクノロジー　先進システムプロジェクト担当
伊藤　　稔　日本ユニシス㈱　ソフトウェア開発部ソリューション技術開発室
原田　俊治　松下電器産業㈱　マルチメディア開発センター　通信第５チーム　技師
山岸　篤弘　三菱電機㈱　情報技術総合研究所　情報セキュリティ技術部
　　チームリーダー主幹

オブザーバー

高橋　基二　情報処理振興事業協会（ＩＰＡ）セキュリティセンター暗号技術調査室
　　室長補佐

RFC 2040, URL: http://ds.internic.net/rfc/rfc2040.txt

[15] J.L.Massey：”SAFER K-64:A Byte-Oriented Block-Ciphering Algorithm” URL: http://www.cs.hut.fi/crypto/safer/safer_1.ps

[16] B.Schneier：URL: http://www.counterpane.com/blowfish.html

[17] E.F.Brickell, D.E.Denning, S.T.Kent, D.P.Maher, W.Tuchman：”SKIPJACK Review Interim Report”, CPSR, 1993 URL: http://www.cpsr.org/dox/program/clipper/skipjack-interim-review.html

[18] ftp://idea.sec.dsi.unimi.it/pub/security/crypt/code/khufu.tar.gz

[19] URL: http://www.entrust.com/library.htm

[20] URL: http://adonis.ee.queensu.ca:8000/cast/cast.html

URL: http://bsa.org/policy/encryption/cryptographers.html
[28] NIST AES　ホームページ：
URL: http://csrc.nist.gov/encryption/aes/aes_home.htm
[29] RC4, URL: http://www.cs.hut.fi/crypto/rc4.txt

2.3　公開鍵暗号（非対称暗号）

[1] J.J.Quisquater and C.Couvereur,"First Decipherment Algorithm for RSA Public Key Cryptosystem", Electronic Letters, v.18,1982,pp155-168
[2] "Security estimate for 512bit-RSA",RSA Lab、1995
URL: http://www.rsa.com/rsalabs/html/tech_notes.html
[3] 岡本栄司著：「暗号理論入門」、共立出版、1993
[4] 岡本龍明、山本博資共著：「現代暗号」、産業図書、1997.
ISO/SC27/DIS17770-3,"Key Management –Part3 Mechanisms using asymmetric Techniques"
[5] ISO/SC27/CD14888-3 、 "Digital Signature with appendix –Part3 Certificate Mechanisms" URL: http://www.iso.ch:8080/jtc1/sc27/
[6] IEEEp1363 Working Draft, URL: http://stdsbbs.ieee.org/groups/1363
[7] 下山武司：「Ajtai, Dwork による Lattice Based Public-Key Cryptosystem の解説」、第 1 回代数曲線とその応用シンポジウム、1997.9.1
[8] 楠田浩二、櫻井幸一：「公開鍵暗号方式の安全性評価に関する現状と課題」
Discussion Paper No.97-J-11、日本銀行金融研究所、1997.7
URL: http://www.imes.boj.or.jp/idps/97-J-11.htm
URL: http://www.imes.boj.or.jp/idps/97-J-11.pdf
[9] 岡本龍明、太田和夫編：「暗号、ゼロ知識証明、数論」、共立出版、1995
[10]宮地充子：「楕円曲線理論の応用数理における開花」、応用数理学会誌、1997.9
[11]PKCS#1:RSA Encryption Standard、RSA Lab、1993.11

[4] 櫻井孝一監訳：「暗号理論の基礎」、1996 年、共立出版
[5] “Open Design”No.14, 1996 年、CQ 出版
[6] 池野信一、小山謙二：「現代暗号理論」、1986 年、電子情報通信学会
[7] 辻井重男：「暗号」、1996 年、講談社選書メチエ
[8] 電子情報通信学会：「『暗号アルゴリズムの設計と評価』ワークショップ講演論文集」、1996 年、電子情報通信学会
[9] 日本アール・エス・エー株式会社：「FAQ World」
URL: http://www.rsa-japan.co.jp/faq/index.htwl

2.5 乱数

[1] 宮武修、脇本和昌：「乱数とモンテカルロ法」、1978 年、森北出版
[2] Stephen K. Park, Keith W. Miller 、西村恕彦訳：「乱数生成系で良質のものはほとんどない」bit April 1993/Vol.25.No.4 、1993 年共立出版
[3] 武者利光：「ゆらぎの世界」自然界の１／ｆゆらぎ 不思議ﾞﾙｶﾞｯｸｽB-442、1980 年、講談社
[4] 橋口住久：「先端科学技術シリーズ3 5 低周波ノイズ」、1991 年、朝倉書店
[5] 飛田武幸：「入門：ホワイトノイズ解析」数理科学No.378December1994、1994 年
[6] ＫＮＵＴＨ、渋谷雅昭訳：「準数値算法／乱数」、1981 年、サイエンス社
[7] 岡本英司：「暗号理論入門」、1993 年 共立出版
[8] 合原一幸、徳永隆治：「カオス応用戦略」、1993 年オーム社
[9] 合原一幸：「カオスセミナー」、1994 年 海文堂版
[10] 高振宇：「ＧＣＣカオス暗号の原理と特徴」インター フェイス97.6、1997 年、ＣＱ出版
[11] ザルツブルグ大学乱数研究チームのホームページ：
URL: http://random.mat.sbg.ac.at/
[12] パソコンソフトのリバースエンジニァリングツールとして利用可能なツールの例：
URL: http://www.iijnet.or.jp/TOOLCRAFT/seihin/ice.htm
URL:/ http://www.lifeboat.co.jp/product/sr97/sr97.html
[13] Mersenne Twister 法ホームページ：
URL: http://www.math.keio.ac.jp/~matumoto/mt.html

,
COUNTERPANE SYSTEMS URL:
http://www.counterpane.com/whycrypto.html

3.6 **暗号機能の検証**

[1] "FIPS PUB 74, GUIDELINES FOR IMPLIMENTATION AND USING THE NBS DATAENCRYPTION STANDARD",NIST
URL: http://www.itl.nist.gov/div897/pubs/fip74.htm

[2] "Implementation Guidance for FIPS PUB 140-1 and the Cryptographic Module Validation Program",NIST
URL: http://csrc.ncsl.nist.gov/cryptval/140-1/1401ig.htm

4 鍵管理

4.3 **鍵の配送**

[1] 今井秀樹：「暗号のおはなし」、1993 年、財団法人日本規格協会

[2] B.Schneier,“Applide Cryptography 2nd edition”, John Wiley& Sons,

[3] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone,

[4] "Applied Cryptography”, CRC Press

4.7 **公開鍵の廃棄**

[1] CCITT,"The Directry-Authentication Framework",X.509,1988

5 暗号に関する法令・規則

[1] 「電子商取引環境整備研究会 中間論点整理」、通産省・電子商取引環境整備研究会、平成９年 11 月。
URL: http://www.ecom.or.jp.miti/971127/

[2] 「情報セキュリティ調査研究報告書」、財団法人社会安全研究財団・情報セキュリティ調査研究委員会、平成９年４月１日
URL: http://www.npa.go.jp/seiankis1/title.htm

[3] 田渕治樹：「ＯＥＣＤガイドラインの概要とその影響」、NIKKEI COMPUTER 1997．5．26.

[4] The OECD Guidelines on Cryptography Policy,
URL: http://www.oecd.org/dsti/iccp/crypto_e.html

[5] 「世界暗号政策ツアー」、WIRED JUNE 1997.

[6] 「暗号化技術の輸出規制に関する FAQ」、
URL: http://www.rsa-japan.co.jp/faq/export.text

[7] Encryption Policy and Market Trends,
URL: http://guru.cosc.georgetown.edu/~denning/crypto/Trends.html

[8] Crypto Law Survey,
URL: http//cwis.kub.nl/~frw/people/koops/lawsurvy.htm

[9] The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption, URL: http://www.crypto.com/key_study/

平成 10 年２月発行
発行：電子商取引実証推進協議会
東京都江東区青海２－４５
タイム２４ビル１０階
Tel　03-5531-0061
E-mail　info@ecom.or.jp