暗号アルゴリズムの安全性のお話

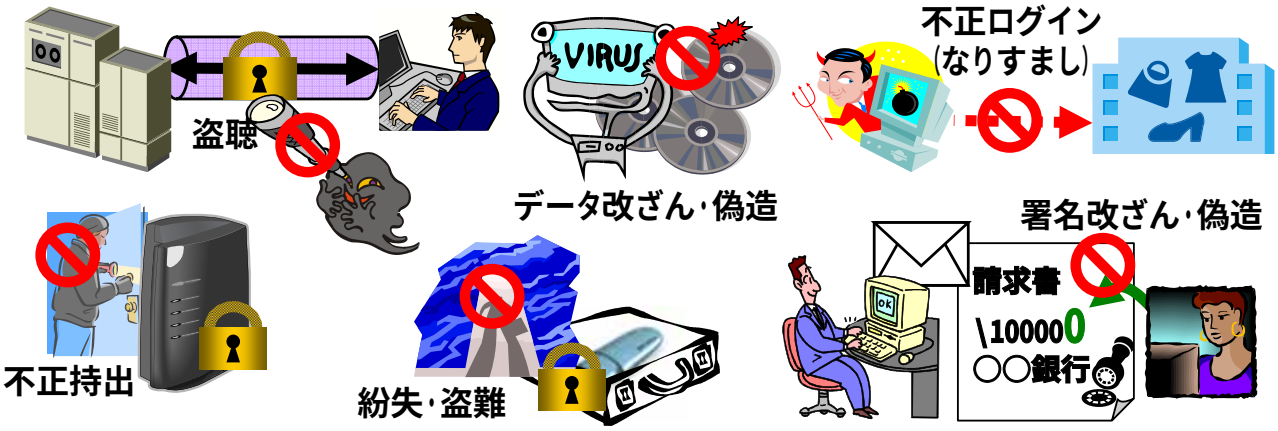
２００８年１１月２７日
日本電信電話株式会社
情報流通プラットフォーム研究所
神田　雅透

どんなところで暗号が使われているか

「使っていることさえ分からない（意識しない）」のも多い

キーワード：機能目的から見た分類

暗号化	**無権限者から情報を秘匿**
鍵配送(情報 = 秘密鍵)	受信者に秘密鍵を配送
鍵共有(情報 = 秘密鍵)	送信者と受信者が秘密鍵を共有
メッセージ認証	**情報の原本性を確認**
ユーザ認証	**権限を有するユーザであることを確認**
(電子)署名	**署名者が情報の原本性を保証**(メッセージ認証+ユーザ認証)



キーワード：暗号方式から見た分類

共通鍵暗号	**暗号化と復号とで同じ秘密鍵を利用する方式**	
ブロック暗号	データを一定の長さに区切って暗号化	AES, Camellia, Triple DES
ストリーム暗号	擬似乱数生成器を利用して暗号化	RC4
公開鍵暗号	**暗号化と復号とで異なる鍵を利用する方式**	
公開鍵暗号	データの暗号化に利用	RSA
ハイブリッド暗号	公開鍵暗号で秘密鍵の配送を行い、共通鍵暗号でデータの暗号化を行う方式	RSA+RC4 (SSL/TLS)
デジタル署名	電子署名を実現する一形態。署名者の認証とデータの完全性を検証を同時に行う方式	RSA, DSA, ECDSA
ハッシュ関数	**任意のデータを一定長のデータに圧縮する方式**	MD5, SHA-1, SHA-2, AHS (SHA-3)

	特長	処理速度	管理すべき秘密鍵	鍵配送	コスト	その他
共通鍵暗号	**導入が比較的容易** **大容量のデータ暗号化に向く**	**非常に速い**	ユーザは秘密鍵すべてを管理 最大n(n-1)/2個	必要	**安価**	**グループ設定が容易**
公開鍵暗号	**鍵管理が比較的容易** **不特定多数間での暗号通信に向く**	遅い (1/1000以下)	**ユーザは1つの秘密鍵を管理** **全体でn個**	**不要**	一般に高価	公開鍵証明書の確認手段が必要

「2010年問題」「世代交代」、一言でいえば・・・・

多くのシステムで利用されている
様々な暗号の安全性低下が無視できなくなってきた

- 暗号解読技術の進展

  脆弱性が見つかれば
  急激に安全性が低下する恐れあり

  - 個別アルゴリズムがもつ脆弱性を利用した手法で解読
  - 解読成功は「**設計要件を満たさない**」の意味であって、実害が生じるか否かは問わない

- 計算機環境の進展

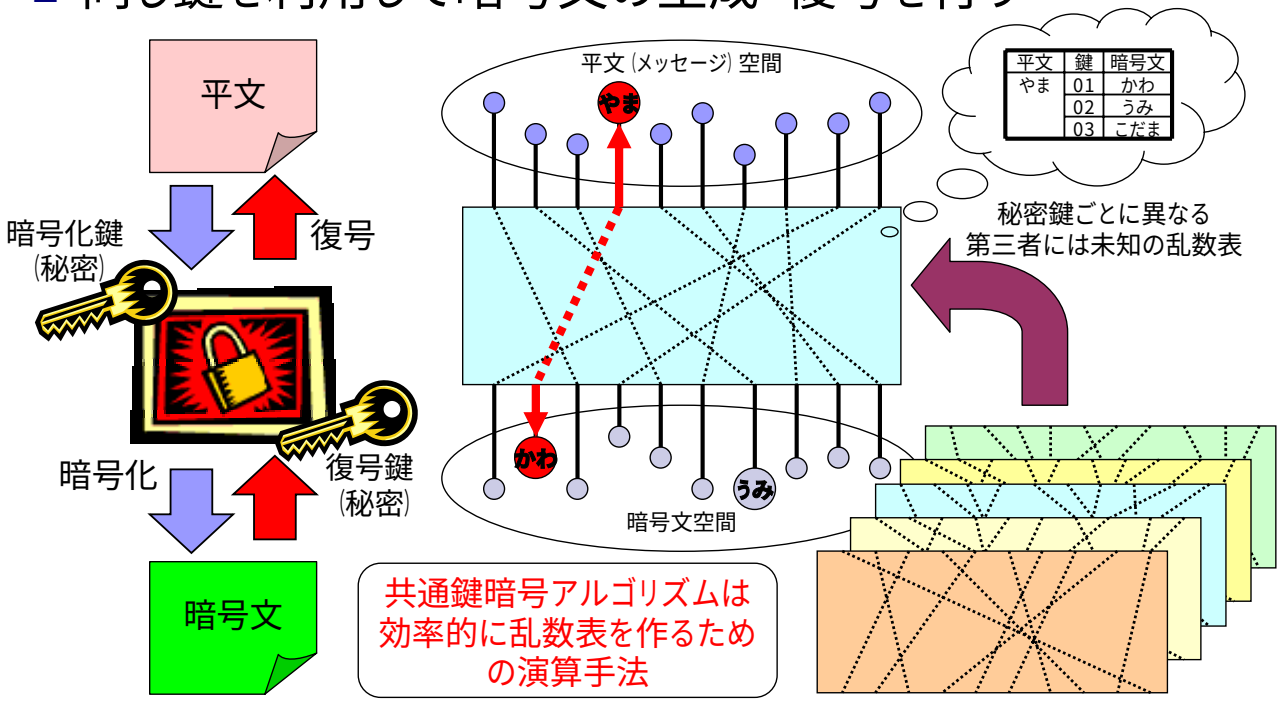
  計算機能力の向上に伴い
  徐々にだが確実に安全性低下

  - 暗号解読技術で解読
    ⇒ 解読手法が有効な証明
  - 鍵全数探索的な手法で解読
    ⇒ 設計方法によらず解読可能

現実的な前提条件での解読成功は
「実害が生じる恐れがあり得る」の意味をもつ

共通鍵暗号 #1

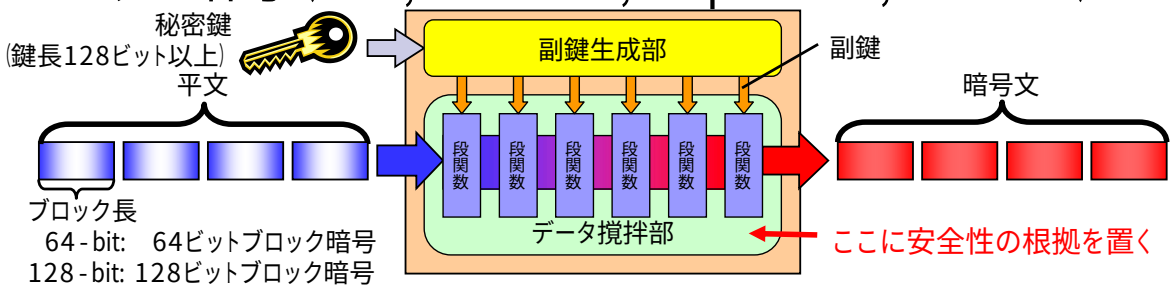
- 同じ鍵を利用して暗号文の生成･復号を行う



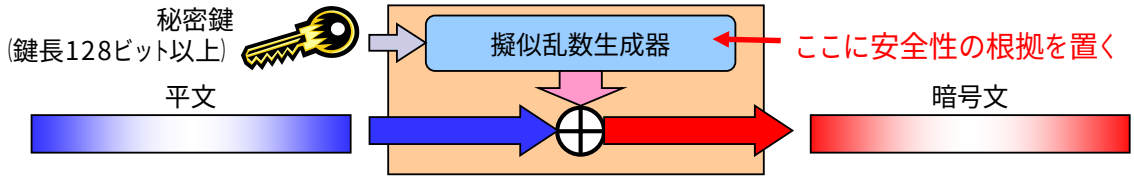
共通鍵暗号アルゴリズムは
効率的に乱数表を作るための演算手法

共通鍵暗号　#2

- ブロック暗号（AES, Camellia, Triple DES, MISTY1）



- ストリーム暗号（RC4）



安全な共通鍵暗号

「現在知られているいかなる攻撃法でも、鍵全数探索法よりも効率よく秘密鍵を求められない」と検証

共通鍵暗号の安全性

もはや「Triple DES/RC4」の時代ではないでしょ・・・

- 暗号解読技術の進展
  - 1981年 Triple DESに対する解読手法(中間一致攻撃)発見
  - 1990年 DESに対する初めての解読手法(差分解読法)発見
  - 1994年 DESに対する初めての解読実験(線形解読法)成功
  - 2001年 WEP (128-bit RC4)に対する初めての解読実験成功
  - 2007年 WEPを1分以内で解読
  - **2008年 WEPに対する受動攻撃での解読実験成功**
    - **データ量20MB**
    - **解読時間10秒**

- 計算機能力向上による鍵全数探索攻撃の脅威

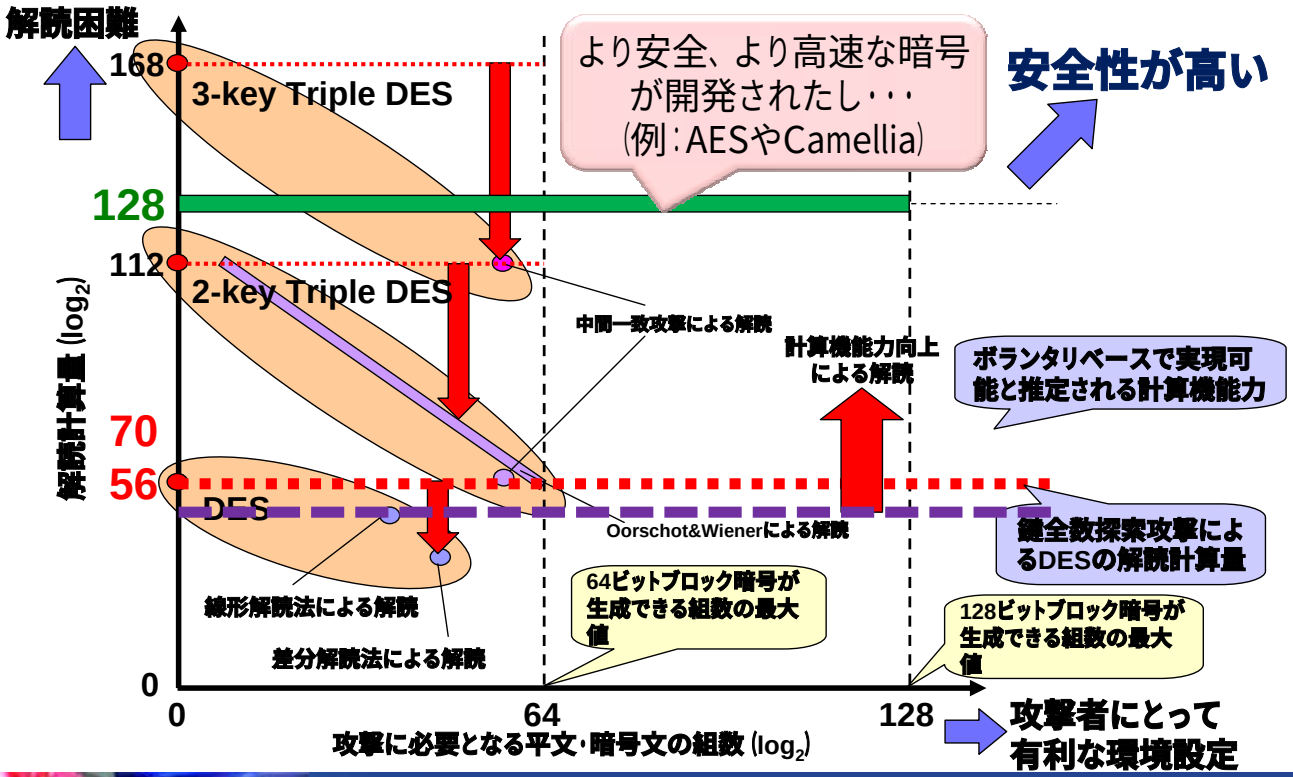
DES Challenge (56ビット全数探索)

1997	約140日	約1万台相当のPC
1998 – I	約40日	約4万台相当のPC
1998 – II	約56時間	DES Cracker (900億/秒)
1999	**22時間15分**	**DES Cracker +約10万台相当のPC (2400億/秒)**

Symmetric-key Challenge

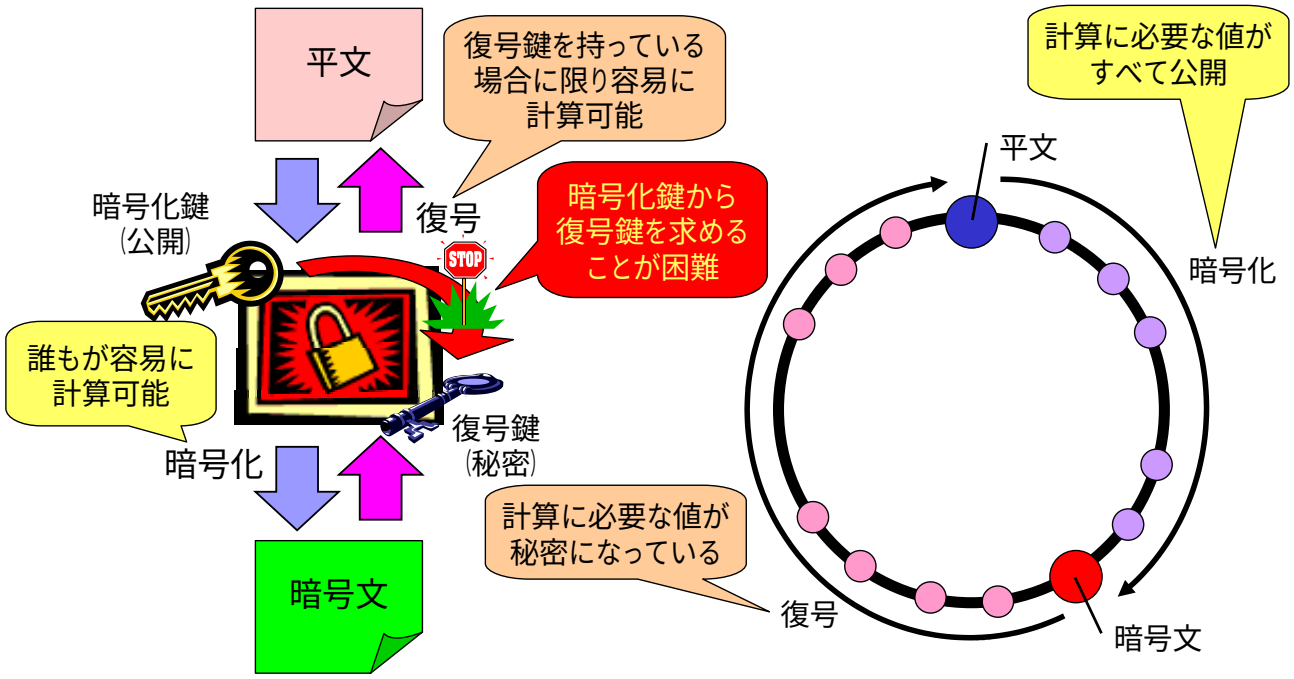
40-bit RC5	3.5時間 (1997/1)
48-bit RC5	313時間 (1997/2)
56-bit RC5	265日 (1997/10)
64-bit RC5	**1757日 (2002/7)**

共通鍵暗号の安全性



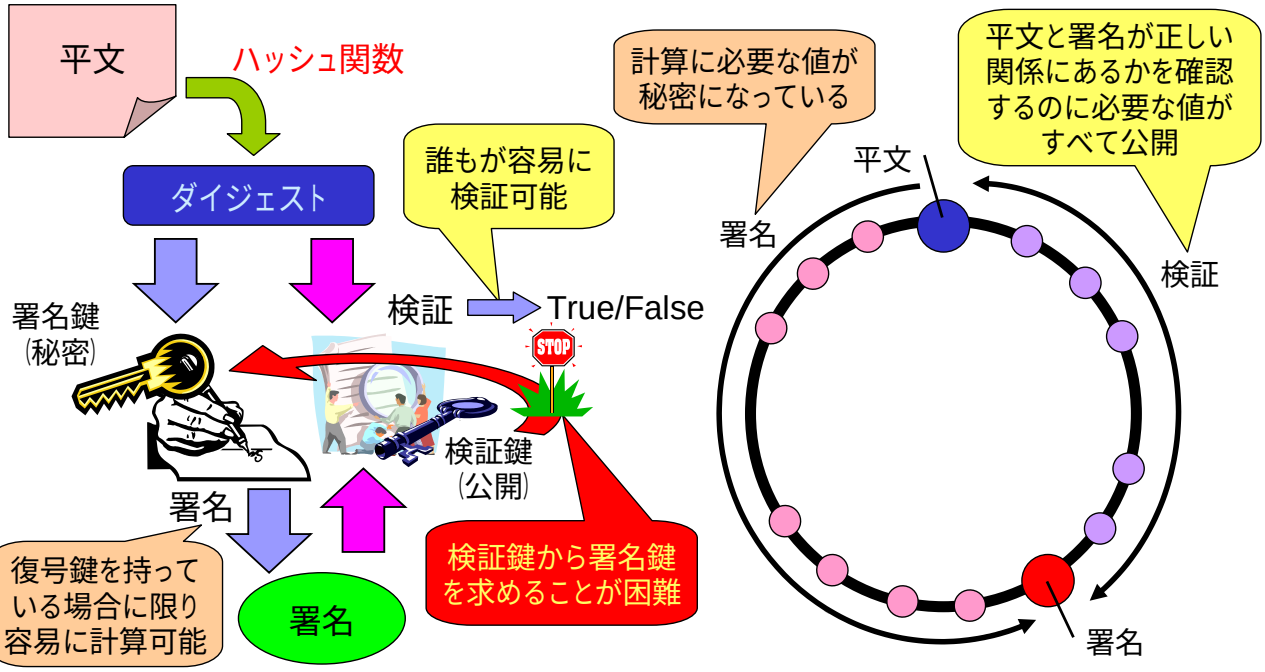
公開鍵暗号

- 異なる鍵を利用して暗号文の生成・復号を行う



デジタル署名

- 異なる鍵を利用してメッセージと署名者の正当性を保証



公開鍵暗号とデジタル署名の安全性

- 公開鍵から秘密鍵を算出することは困難

  ➡ **数学的未解決問題に安全性の根拠**

  - **素因数分解問題困難性** (RSA)

    $n = pq$　　n ✂ (p, q) : **素数**

  - **有限体上の離散対数問題困難性** (ElGamal, DSA)

    $z = g^x \pmod{p}$　　(z, g, p) ✂ x

  - **楕円曲線上の離散対数問題困難性** (PSEC-KEM, ECDSA)

    $Q = [k]P \quad (on \quad E)$　　(Q, P, E) ✂ k

(定性的な意味での) 安全な公開鍵暗号

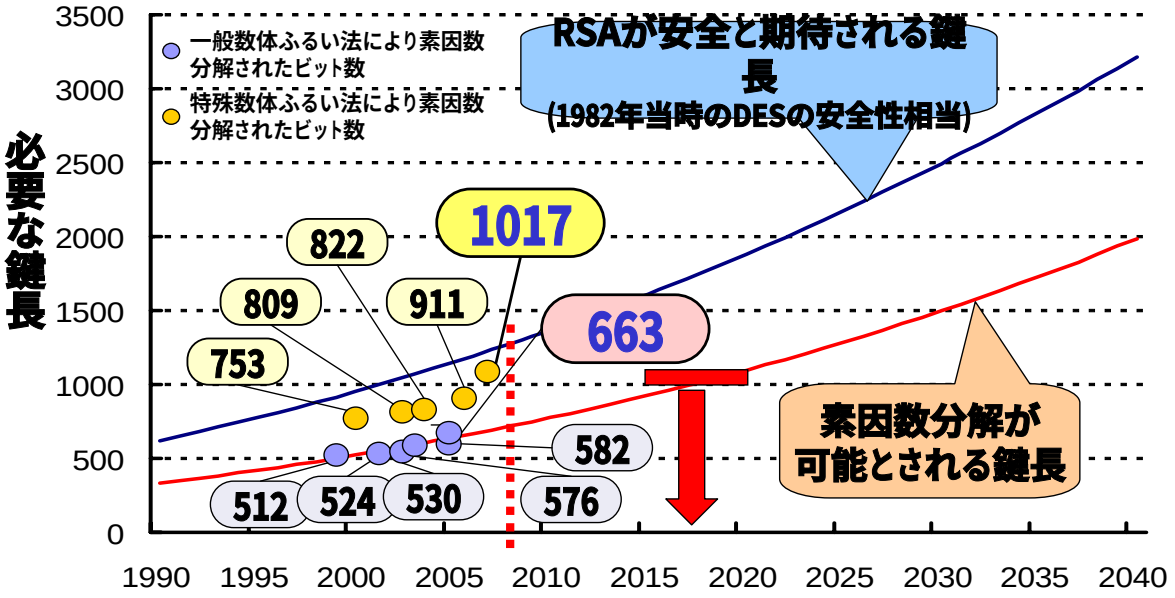
「ある手法で公開鍵暗号を解けたならば数学的未解決問題が解ける」と証明

⇔ 暗号解読が数学的未解決問題と同じ難しさ

公開鍵暗号・デジタル署名の安全性

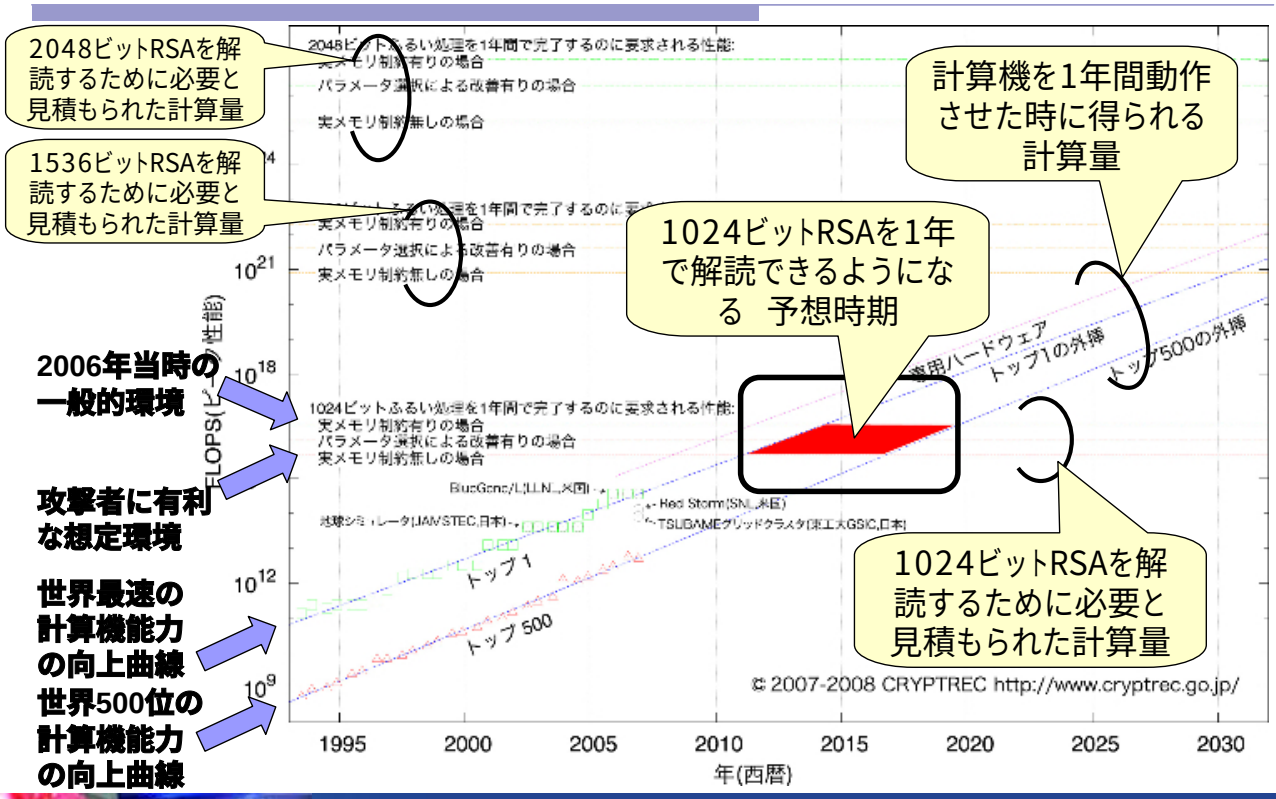
計算機能力向上による数学的未解決問題解法の脅威

例えば、RSA暗号は「素因数分解問題」が解けると秘密鍵が求まる



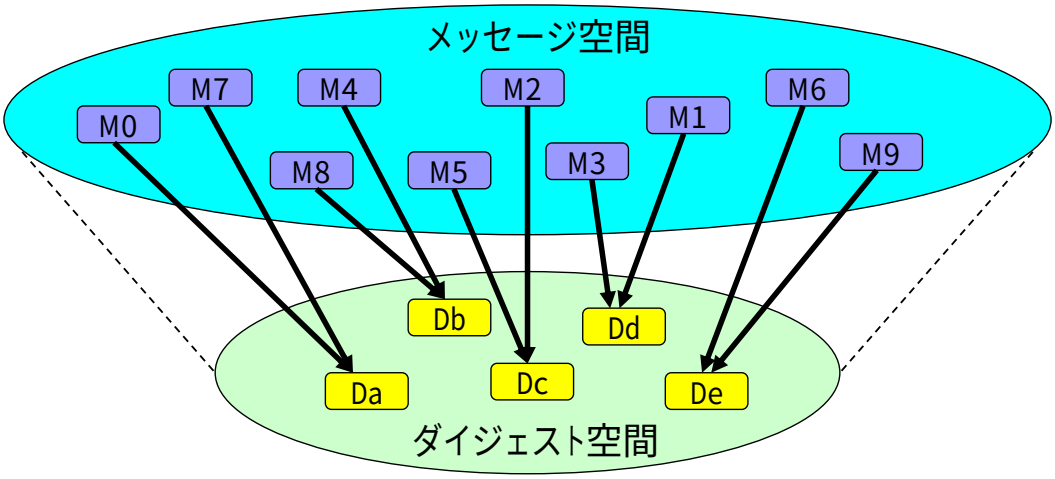
参考文献: A.K.Lenstra,E.Verhaul "Selecting cryptographic key sizes" Proc. PKC 2000, R.P.Brent "Recent progress and prospects for integer factorization algorithm" ,Proc. COCOON 2000

RSA暗号・RSA署名の安全性



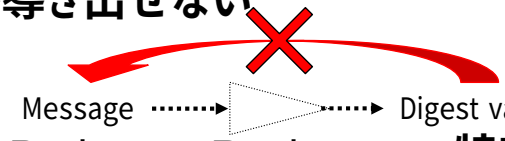
ハッシュ関数

- 任意長のメッセージを一定長のダイジェストに写像
  - **メッセージ空間よりもダイジェスト空間のほうが小さいので、理論上必ず衝突が起こる（この点が暗号と異なる）**
    SHA-1：**メッセージ空間** 2^{512} ⇒ **ダイジェスト空間** 2^{160}



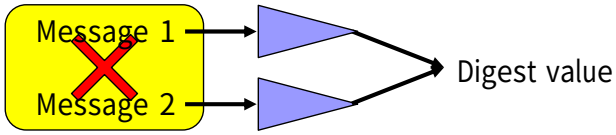
ハッシュ関数の安全性

- 一方向性（Onewayness）
  - Preimage Resistance: **ダイジェストからもとのメッセージ（の候補）が導き出せない**



  - Second Preimage Resistance: **特定のダイジェストに一致する別メッセージを作り出せない**

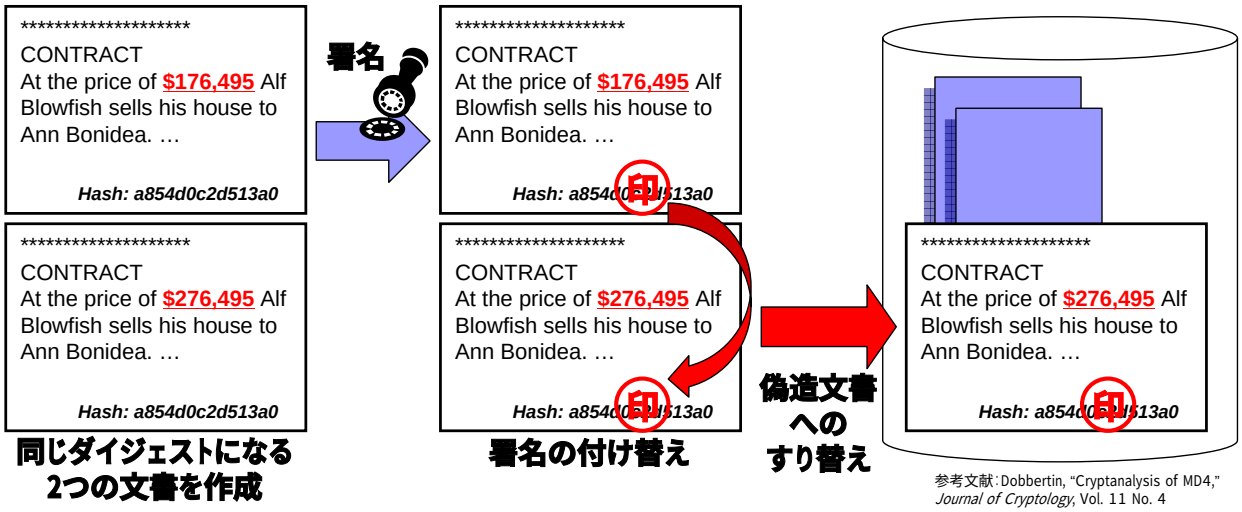
- 衝突困難性・非衝突性（Collision-free / resistance）
  - **同じダイジェストになる異なるメッセージ組を作り出せない**



ハッシュ関数に問題が起きることの影響 #1

例えば、デジタル署名のすり替え（改ざん）が可能になる

- 改ざん検知が不可能となる文書をあらかじめ用意可能
  - X.509の偽造証明書もこのケース

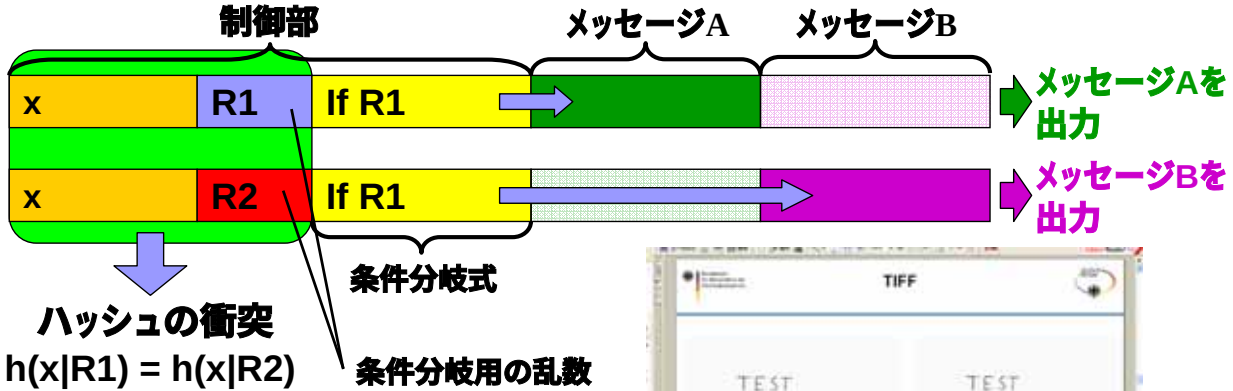


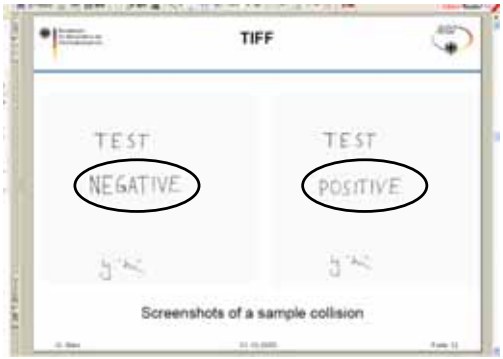
参考文献：Dobbertin, "Cryptanalysis of MD4," *Journal of Cryptology*, Vol. 11 No. 4

ハッシュ関数に問題が起きることの影響 #2

例えば、異なる任意のメッセージの出力が可能になる

- 制御部に出力先を決める乱数をあらかじめ組み込み



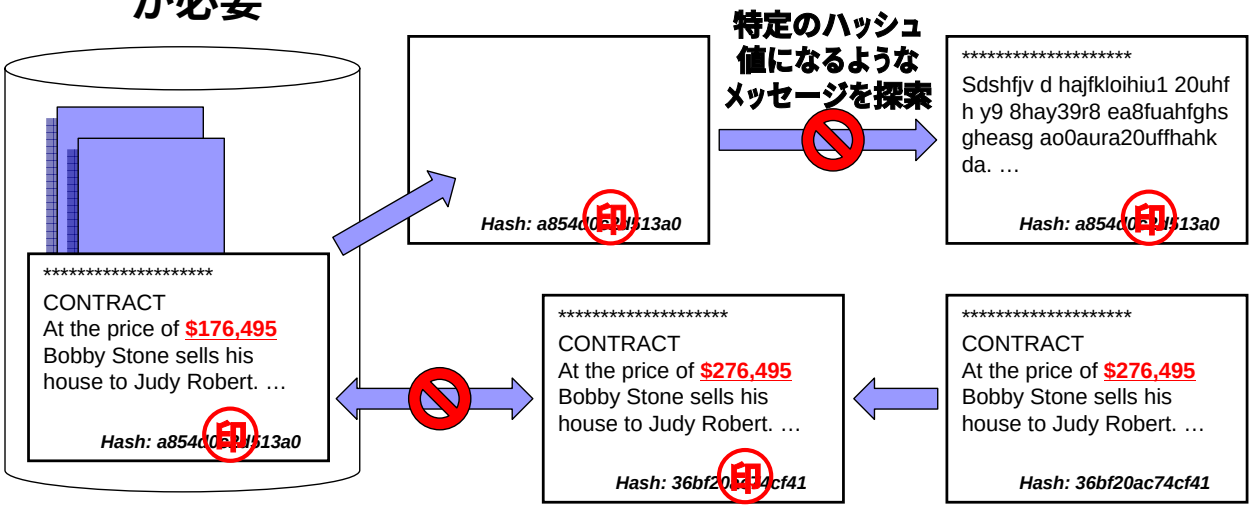


参考文献:
Daum and Lucks, "Attacking Hash Functions by Poisoned Messages "The Story of Alice and her Boss"," Presented at the rump session of EUROCRYPT '05, 2005
Max Gebhardt, Georg Illies, and Werner Schindler, "A Note on the Practical Value of Single Hash Collisions for Special File Formats," Proceedings of Cryptographic Hash Workshop

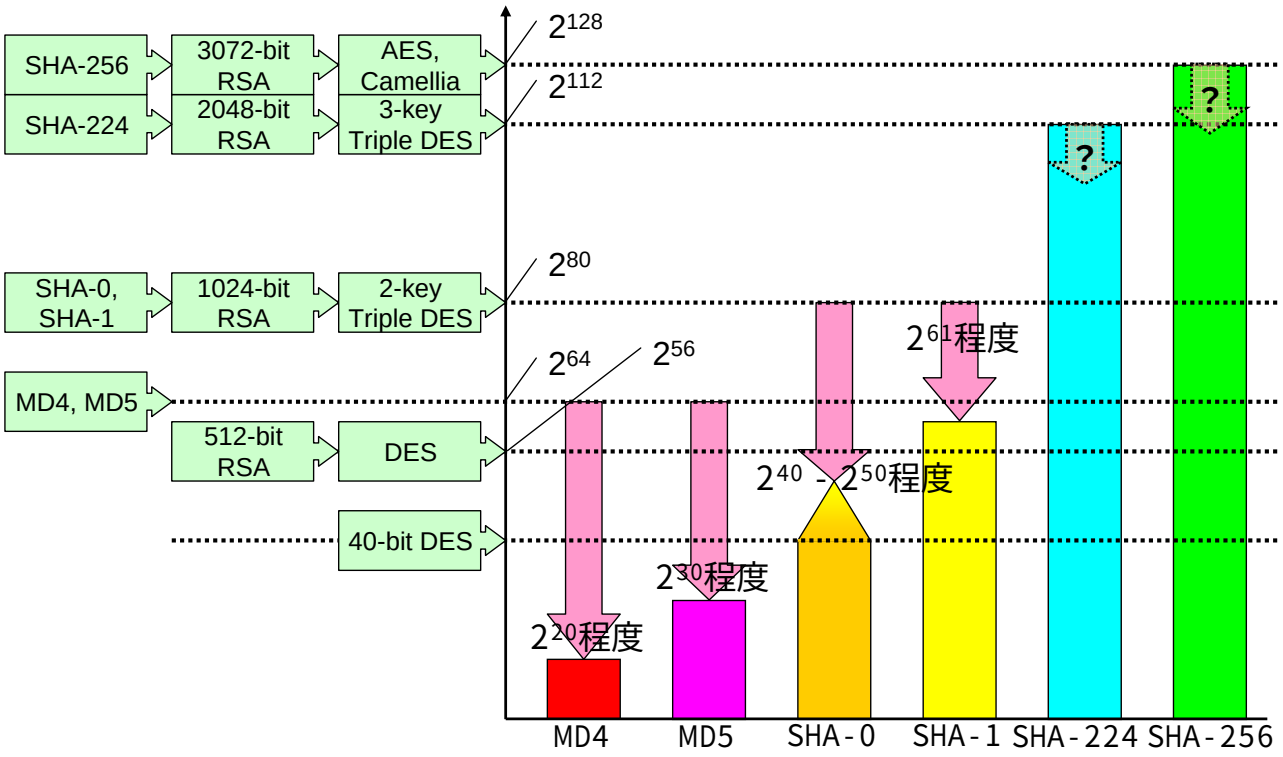
ハッシュ関数に問題が起きることの影響 #3

ただし、以下のようなケースが可能になるわけではない

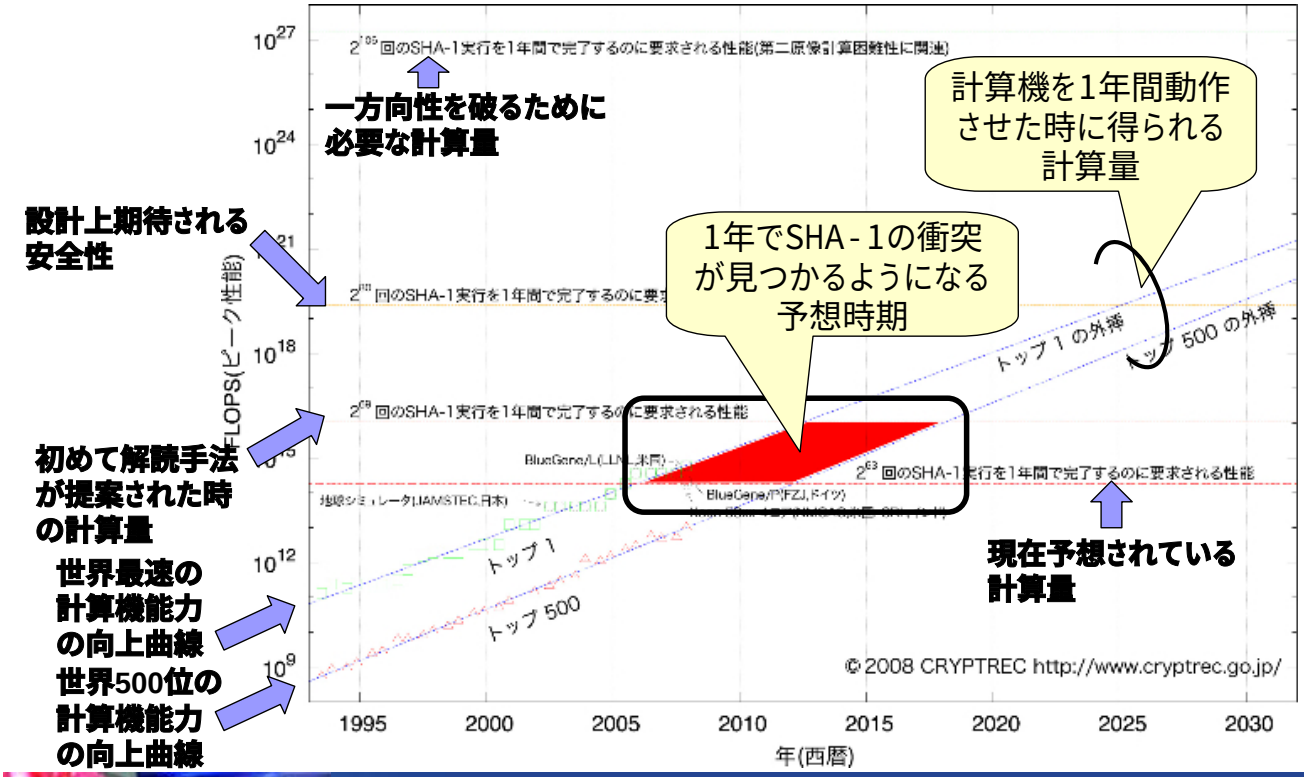
- 改ざん検知が不可能となる文書に後から置き換える
  - **このケースが可能になるためには「一方向性」が破られることが必要**



代表的なハッシュ関数の安全性



SHA-1の安全性

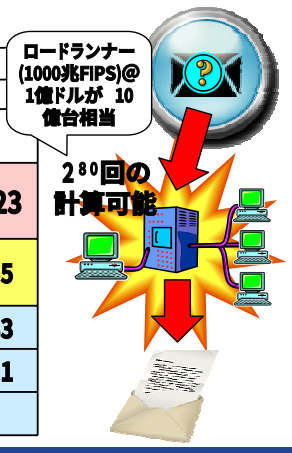


「暗号が破れた」とは

現在主流の暗号は長期利用に耐えられそうもない

- 今までは、「個別アルゴリズムが仕様上の安全性上限を満たさない」ことが問題
- 2010年問題は、「カテゴリの仕様上の安全性上限設定が妥当ではなくなってきている」ことが問題

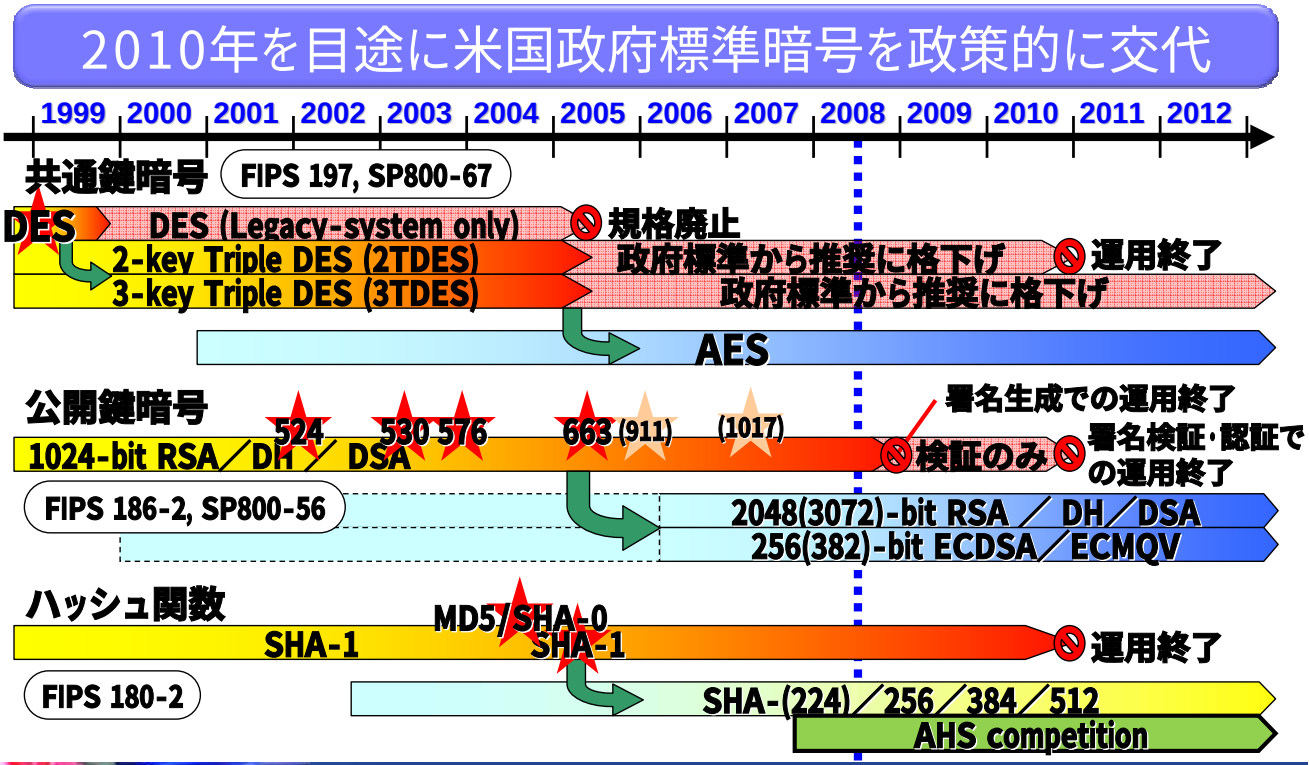
等価安全性強度 / 最良の攻撃法	共通鍵暗号 鍵全数探索 (64ビット)	ハッシュ関数 誕生日攻撃	公開鍵暗号 素因数分解 数体ふるい法 (663－700ビット)	公開鍵暗号 離散対数 指数計算法 (607ビット)	公開鍵暗号 楕円曲線 ρ法 (109ビット)
80ビット	2-key Triple DES	SHA-1	K=1024	L=1024, N=160	F=160-223
112ビット	3-key Triple DES	SHA-224	K=2048	L=2048, N=224	F=224-255
128ビット	AES-128	SHA-256	K=3072	L=3072, N=256	F=256-383
192ビット	AES-192	SHA-384	K=7680	L=7680, N=384	F=384-511
256ビット	AES-256	SHA-512	K=15360	L=15360, N=512	F=512



技術面からの次世代暗号移行について

- 共通鍵暗号は「より安全で、より処理性能が高い新しいアルゴリズムへ」
  - 64ビットブロック暗号 (Triple DES, MISTY1など) から128ビットブロック暗号 (AES, Camelliaなど) への移行で問題なし
- 公開鍵暗号は「より安全にするために鍵長をより長く」
  - 処理性能が低下する可能性が高いことが問題になるかも
    - PKIインフラが必要な場所ではRSAの優位性は揺るがず
    - 楕円暗号は個別システムから導入が進むと予想
- ハッシュ関数は「緊急避難的にとりあえずSHA-2へ」
  - 2006/3/15　SHA-1取扱いについて公式アナウンス
    - デジタル署名／タイムスタンプサービス等で利用：速やかに移行
    - 2010年以降もSHA-1が利用可能：鍵導出・HMAC・擬似乱数生成
  - SHA-3が市場に出てくるのは早くても10年ぐらい先の話・・・

米国政府の次世代暗号移行政策 ('05.8公表)



日本政府の次世代暗号移行政策

詳しくは・・・・

繁富さんにバトンタッチ！

SSLサーバの現状調査結果について

SSL/TLSは暗号化技術？

- 「縁の下の力持ち」ゆえ、どんな暗号技術を使っているかさえ認識されていないことも多い

共通鍵暗号	ブロック暗号	RC2(40), DES (40,56), Triple DES, IDEA, AES, Camellia, SEED
	ストリーム暗号	RC4(40, 128)
公開鍵暗号		RSA, ECDH, DH
デジタル署名		RSA, DSA, ECDSA
ハッシュ関数		SHA-1, MD5

128 bit SSLによる暗号化って何？

個人情報の保護

(1) 通信データの暗号化
SSLという事実上世界標準の暗号化技術を利用しています。

インターネットバンキングは、128ビットSSL(Secure Sokets Layer)暗号化通信方式を採用
・米国ベリサイン社による最新の暗号化技術を採用して、情報の盗聴・情報の書換えを防いでおります。
※本方式で暗号化されたお客様の情報は、2の128乗通りの符号を解読しなければ見ることができないため、現在、最もセキュリティ強度が高い暗号化技術といわれています。

「秘匿性」の確保
[redacted]は、オンライン取引に求められる高いセキュリティを確保しています。
128bitSSLによる世界最高水準の暗号化技術の導入によって、個人金融資産に関わるデリケートな情報を、お客さま以外の第三者に盗み見されたり、データを改ざんされたりすることを防止します。

128 bit SSL(Secure Sockets Layer)暗号化技術の採用
[redacted]では、インターネット通信時に128 bit SSL(Secure Sockets Layer)という強力な暗号化技術を採用し、お客さまの重要な情報が盗まれたり、故意に書き換えられたりされないように保護しています。

[redacted]では128ビットRC4や168ビットTriple-DESなどの非常に強力なものを含め、SSL3で規定されているすべての暗号化に対応していますので、それらに対応しているブラウザをお持ちなら、通信内容を強力に保護することができます。

SSL/TLSは暗号化技術？

セキュリティに関することで「下位互換」を重視すべきか？

例えば、SSL2.0でもつなげられることはユーザの利益か

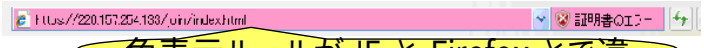
「SSL 2.0には既知の脆弱性があるため、HTTPSプロトコルなどの SSLで保護された通信が解読され、重要な情報が漏洩する可能性があります。(IPAセキュリティセンタ)」

SSLサーバ証明書はどこまで信用していいものか？

実在証明無しの証明書はドメインさえ所有していれば誰にでも発行

EV SSL証明書の登場

運営組織の実在性確認
プロセスの厳格化



色表示ルールが IE と Firefox とで違う！

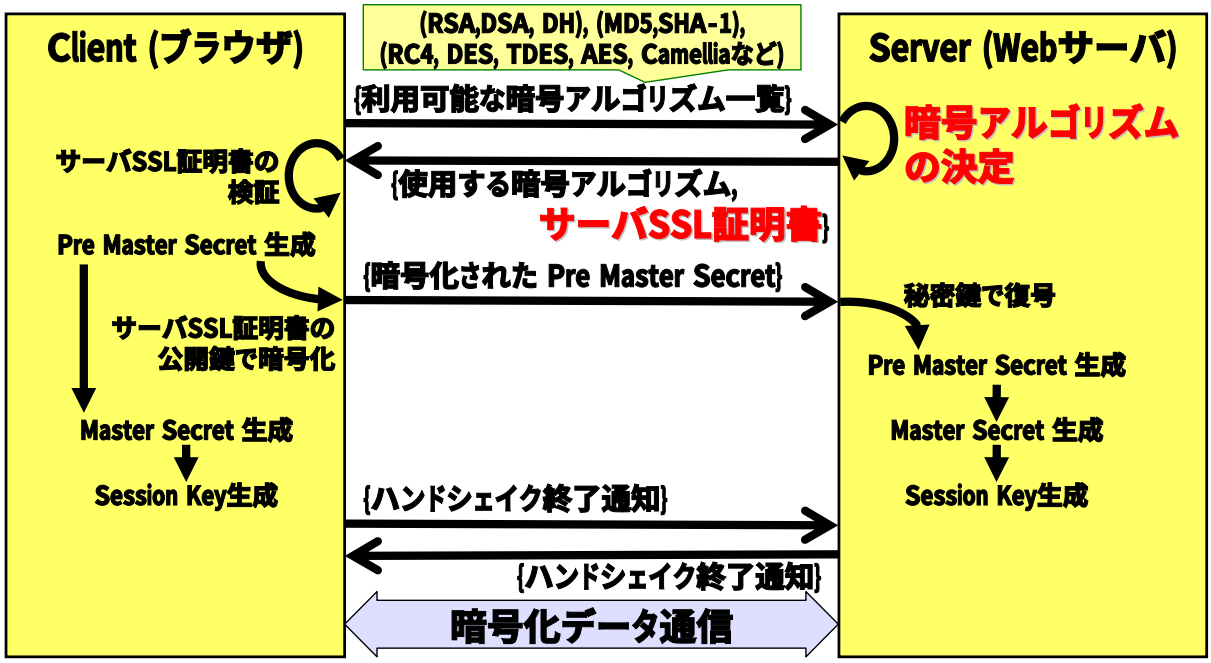
ITpro > セキュリティ
ニュース
"本物"のSSL証明書を持つフィッシング・サイト出現
記事一覧へ
米SANS Institute や米Websense は現地時間2月13日，実在するサイトに思わせるようなドメイン名を持ち，なおかつ，そのドメイン名に対して発行されたSSL用サーバー証明書(デジタル証明書)を持つ偽サイトが確認されたとして注意を呼びかけた(関連記事)。

出典：ITPro>セキュリティより

Camellia A 128-bit Block Cipher Suitable for Multiple Platforms
20081127 Internet Week 2008資料
(c) 日本電信電話株式会社 情報流通プラットフォーム研究所

SSL/TLSの概要

- ハンドシェイクで使用する暗号アルゴリズムを決定



期待した設定どおりの動作をしていますか

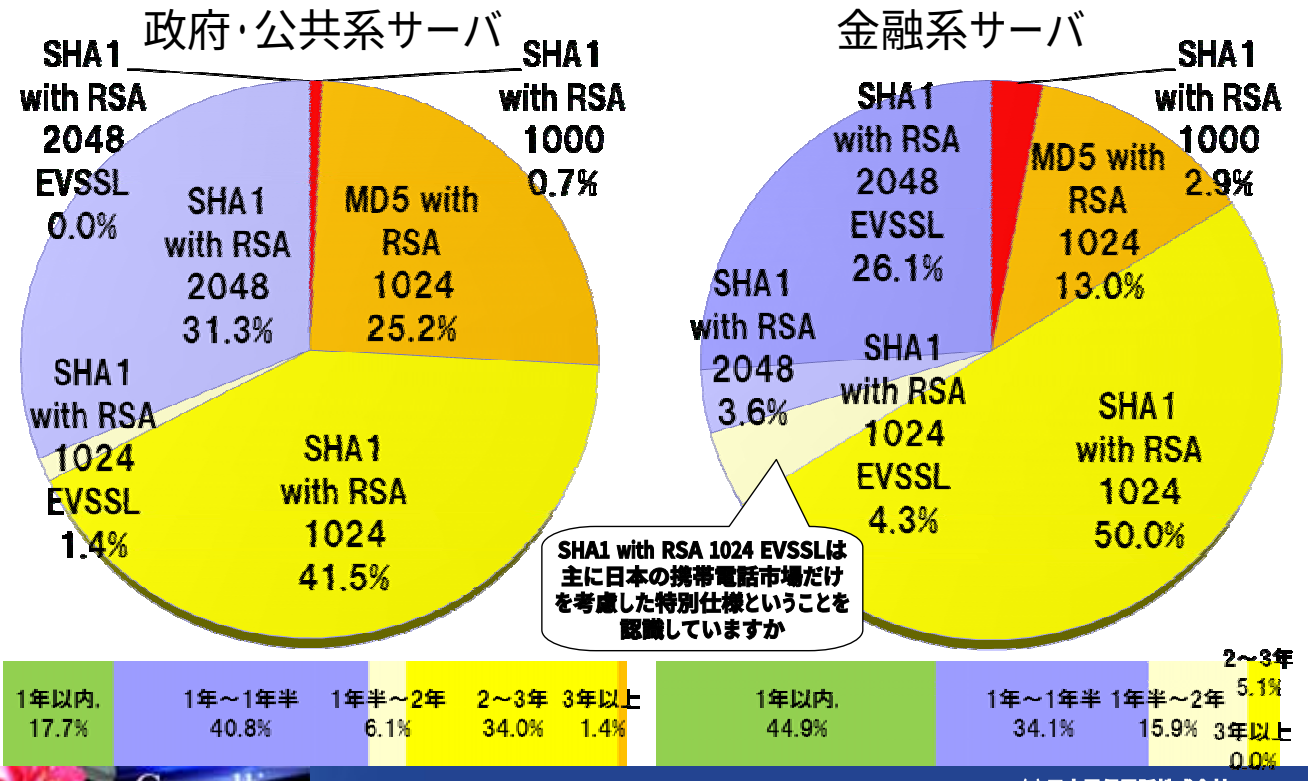
- 暗号アルゴリズム設定がベンダ任せになっていませんか
  - 「AES256-SHA」**が使えるはずなのに**「RC4-SHA」**を選択**
  - 「**鍵交換は**2048**ビット**RSA」**なのに**「**署名は**1024**ビット**RSA」
  - 「**暗号化は**AES 256**ビット**」**なのに**「**鍵交換は**512**ビット**RSA」
  - RC4**で接続といってもSHA1を使っているとは限らない**
  - **どの暗号アルゴリズムが選択されたかブラウザ(クライアント)からは確認できない**
  - 128**ビットSSLといいながら、弱い暗号でつなごうと思えば接続できてしまう設定をしている**
  - **同じ企業内のサーバであっても設定内容に一貫性がない**
  - SSL**サーバ証明書は自動更新？**

実際にどうなっているのか検証してみました

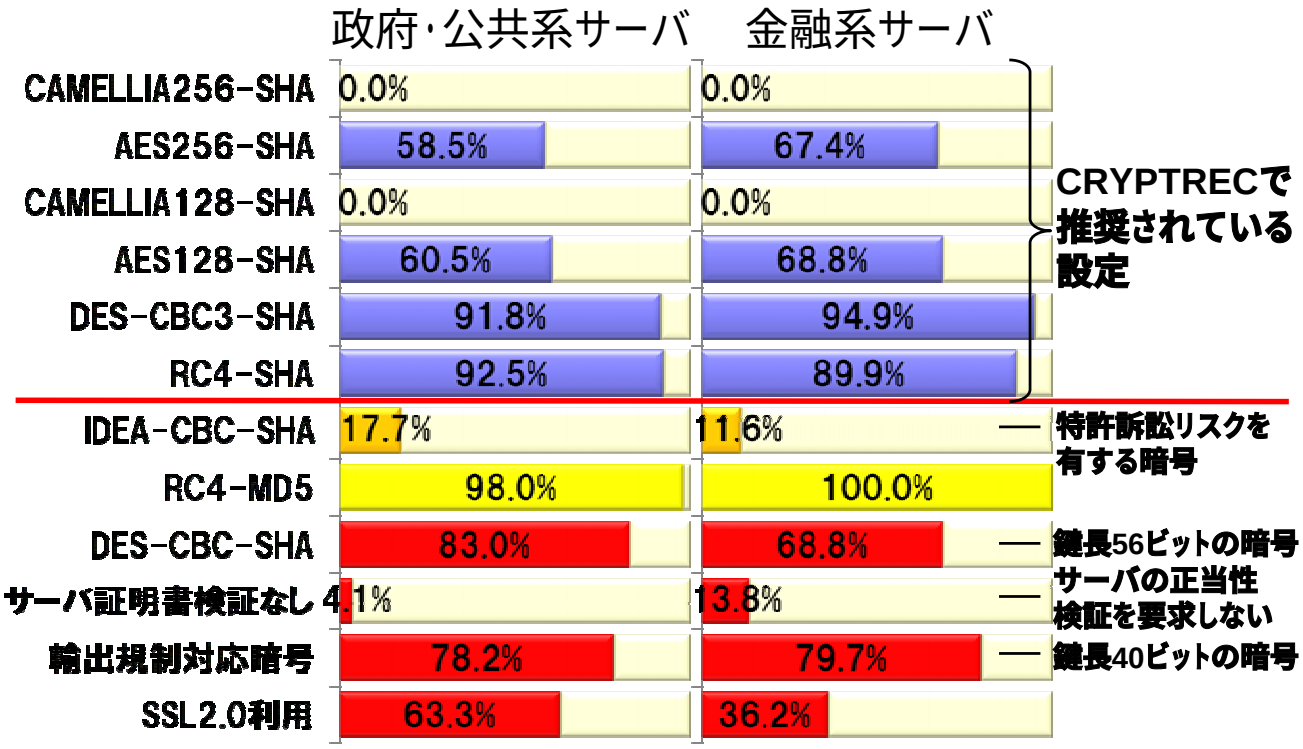
- 調査対象：
  政府・公共系サイト及び金融系サイトの各トップページからたどることができるSSLサーバ
- 調査期間：2008年10月～11月
- 調査内容：
  - **サーバ証明書の状況**（**有効期限、アルゴリズム、鍵長等**）
  - **暗号選択設定の状況**（**接続可能なアルゴリズム**）
  - **ブラウザでの実際の接続状況**（IE6, IE7, Firefox3）

中間調査結果を少々・・・（重複あり）
政府・公共系：147サーバ、金融系：138サーバ

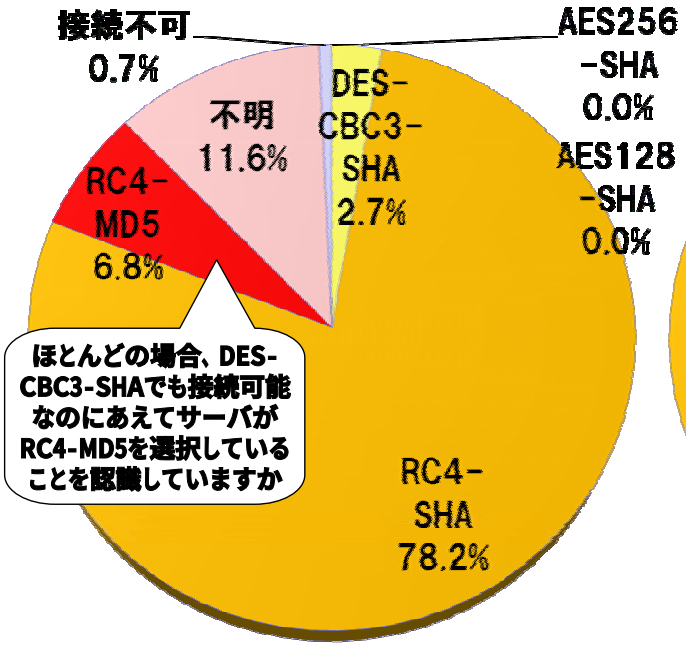
サーバ証明書（アルゴリズムと有効期限）

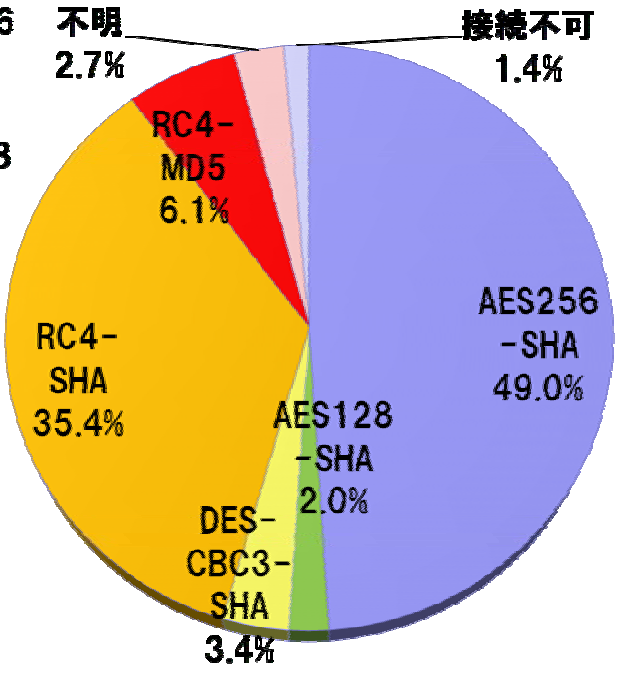


サーバ受け入れ可能な主な暗号アルゴリズム候補



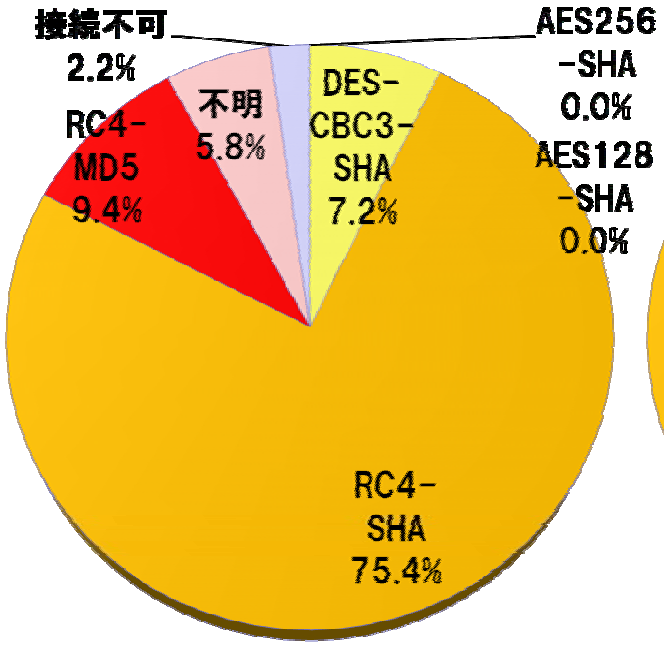
政府系サーバでの接続アルゴリズム



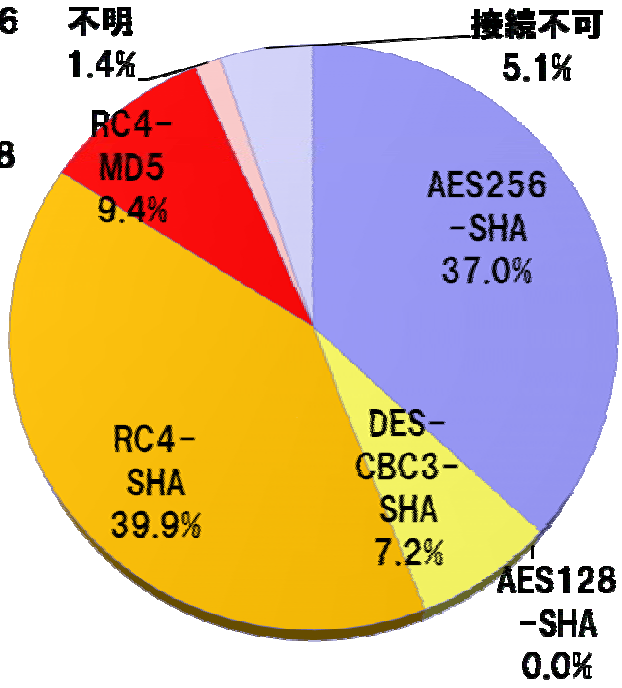


金融系サーバでの接続アルゴリズム

Internet Explorer 6（XP SP2）



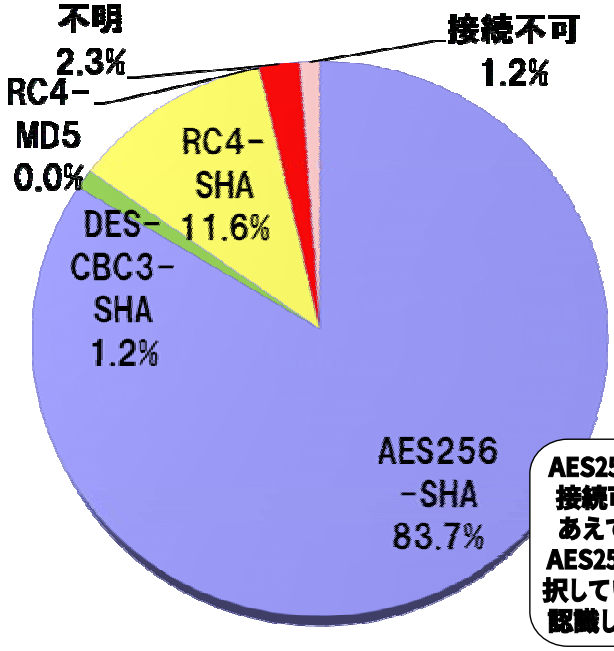
Firefox 3（XP SP2）



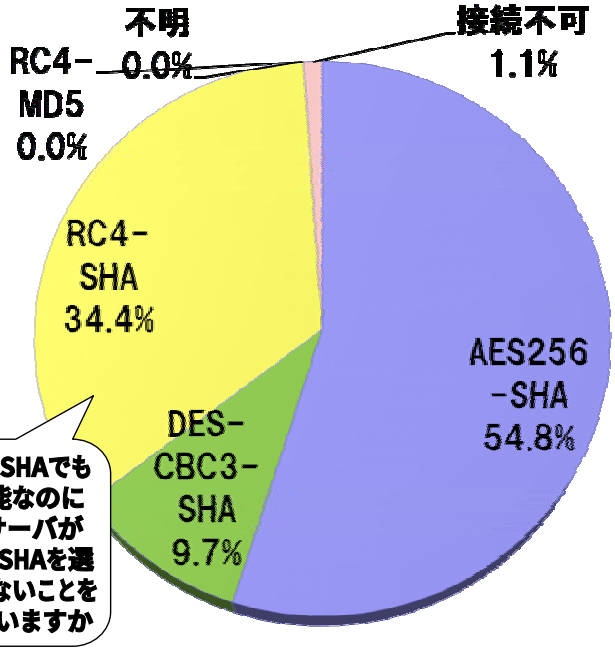
AESが利用される設定になっているか

Firefox 3（XP SP2）（AESが接続可能なサーバに限定）

政府・公共系サーバ



金融系サーバ

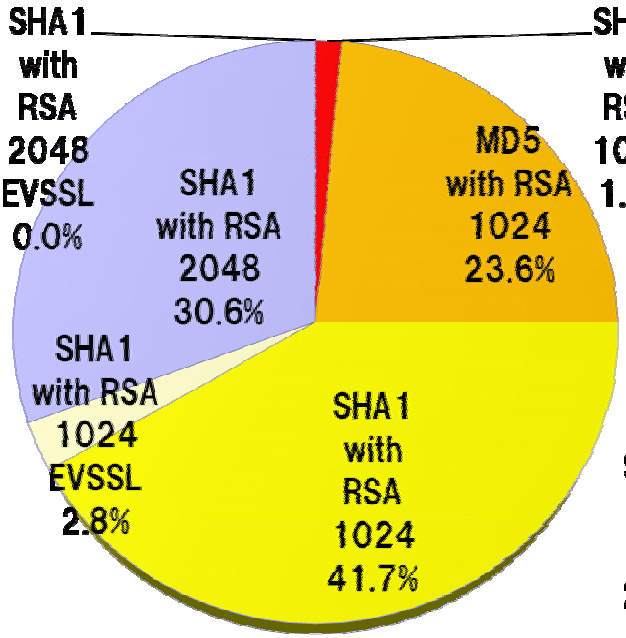


AES256-SHAでも接続可能なのにあえてサーバがAES256-SHAを選択していないことを認識していますか

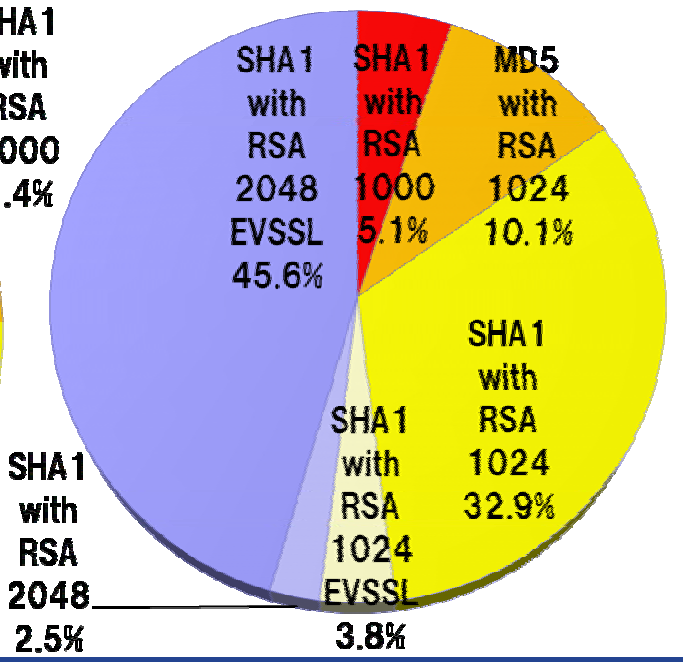
SSLサーバ証明書と選択される暗号アルゴリズム

Firefox 3（XP SP2）（AESで接続しているサーバに限定）

政府・公共系サーバ



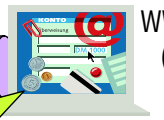
金融系サーバ



国産暗号によるSSL暗号通信が初めて可能に

WWWサーバ（例：ECサイト）

SSL/TLS暗号通信 by Camellia



WWWブラウザ（例：顧客）

Sever: Apache+OpenSSL
日本・欧州ではCamelliaを利用可能にすることを推奨

Browser: Firefox 3
「史上最速・最軽量」と
「日本発の技術を搭載」がキーワード

引用：Apache Lounge，http://www.apachelounge.com/forum/viewtopic.php?t=1992

【参考】 Camelliaを利用可能にするためのガイダンス資料を公開しています。また、以下のOSに同梱されているOpenSSLではCamelliaが使えるバージョンのものがすでに組み込まれています。

Fedora Core 9以降、OpenSUSE 10.3以降、Gentoo Linux 2008.0以降、FreeBSD 7.0以降、FreeBSD ports 2007/6/12以降

引用：Mozilla Firefox 3 日本語版レビューアーズガイド