RSA署名に対する新しい攻撃法の提案について
― Coron-Naccache-Sternの攻撃法 ―

宇根 正志（うね まさし）

要　旨

RSA方式は、素因数分解問題の困難性に依拠した公開鍵暗号方式であり、データ守秘（RSA暗号）とデジタル署名（RSA署名）の両方の機能を有している。RSA方式のアルゴリズム自体に対しては、これまで現実的な脅威となる攻撃法が提案されていないことから、安全性の高い公開鍵暗号方式として、金融分野をはじめとする幅広い分野で利用されている。

Gemplus社（フランス）のCoron、Naccache、ルーバン・カトリック大学（ベルギー）のSternは、1999年4月、RSA署名に対する新しい攻撃法を発表した。本攻撃法は、一定条件を満足するメッセージの署名を利用して別のメッセージの署名を偽造するというものであり、公開鍵を直接素因数分解するよりも少ない計算量で署名偽造が可能となる。

Coron-Naccache-Sternの攻撃法で注目されるのは、本攻撃法がRSA署名を利用したデジタル署名方式の国際標準ISO/IEC 9796-2に適用できるという点である。本国際標準は、主にICカード上での実装を想定して策定されており、署名からメッセージを復元できる仕組みとなっている。本攻撃法は、こうしたデジタル署名方式の特徴点を巧みに利用したものであり、RSA署名のアルゴリズムを直接攻撃するものではない。Coronらの研究成果は、暗号アルゴリズム自体の安全性だけでなく、その利用方法を含めた総合的な評価の必要性を強く示唆するものである。

本稿では、まず、これまでのRSA署名の安全性に関する主な研究成果を整理し、ISO/IEC 9796の概要を説明する。その上で、Coron-Naccache-Sternの攻撃法について説明し、情報セキュリティ技術の標準化を担当するISO/IEC JTC1/SC27の対応状況やICカード等の標準規格への影響について説明する。

キーワード：公開鍵暗号、RSA署名、デジタル署名、ISO、国際標準

本稿の作成にあたっては、櫻井幸一助教授（九州大学大学院システム情報科学研究科）から、本テーマでの執筆を示唆して頂いたほか、田中初一教授（神戸大学工学部）、清水秀夫研究員（放送・通信機構横浜リサーチセンター）から、本攻撃法の位置付けや技術的な内容について貴重なコメントを頂戴した。ここに記して感謝したい。もっとも、本稿のあり得べき誤りはすべて筆者に属することは言うまでもない。

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

1．はじめに

RSA方式は、データ守秘（RSA暗号）とデジタル署名（RSA署名）の両方の機能を有する公開鍵暗号方式である。RSA方式の安全性は素因数分解問題の困難性に依拠しており、これまで現実的な脅威となる攻撃法は提案されていない。

こうした中、1999年4月、フランスのICカード製造業者であるGemplus社のCoron、Naccache、ベルギーのルーバン・カトリック大学のSternはRSA署名に対する新しい攻撃法を発表した。Coron-Naccache-Sternの攻撃法（以下、CNS攻撃法）は、署名変換の対象となるデータが小さな素数の積となるメッセージを生成し、そのメッセージの署名を利用して、他のメッセージの署名を偽造するというものである。本攻撃法では、公開鍵を直接素因数分解する攻撃法に比べ、素因数分解の対象となる合成数を小さくすることができるため、攻撃に必要な計算量を大幅に削減できる。本攻撃法は、直ちに現実的な脅威となる訳ではないが、「攻撃者が自分にとって都合のよいメッセージの署名を入手できる」という条件が成立する実装環境においては、その安全性に問題点を生じさせる可能性もある。

CNS攻撃法において留意すべき点は、RSA署名を利用したデジタル署名方式の国際標準ISO/IEC 9796-2に対して本攻撃法が有効である点である。ISO/IEC 9796-2は、計算能力が比較的限定されているICカード上での実装を想定して策定されており、処理対象となるデータ量が少なく、署名からメッセージを復元可能なデジタル署名方式が採用されている。本攻撃法は、こうしたデジタル署名方式の特徴を巧みに利用したものであり、RSA署名のアルゴリズムを直接攻撃するものではない。情報セキュリティ技術の標準化を担当しているISO/IEC JTC1/SC27は、CNS攻撃法への対応方法について現在検討を進めている。ISO/IEC 9796-2は、EMV'96をはじめとするICカードの業界標準に採用されていることから、SC27の対応次第では、これらの業界標準へも影響が及ぶ可能性がある。

CNS攻撃法は、暗号方式の評価を行う際には、その利用方法や実装環境を十分考慮した上で総合的な判断を下すことが重要であることを強く示唆するものである。金融分野を含めて、暗号技術を利用した製品やシステムのセキュリティを検討する際には、こうした研究成果を考慮していく必要があるものと思われる。

本稿では、まず、これまでのRSA署名の安全性に関する研究結果を整理し、ISO/IEC 9796に規定されているデジタル署名方式について説明する。その上で、CNS攻撃法の概要やISO/IEC 9796-2への適用方法について説明し、これまでのISO/IEC JTC1/SC27におけるCNS攻撃法への対応状況やCNS攻撃法がICカードの業界標準に及ぼす影響について説明する。

2．RSA方式とそのアルゴリズム

RSA方式は、1978年に、Rivest、Shamir、Adlemanによって考案された最初の本格的な公開鍵暗号であり、データ守秘（RSA暗号）およびデジタル署名（RSA署名）の両方に利用可能である。RSA方式は、大きな2つの素数からなる合成数を素因数分解する問題を解くことが計算量的に困難である[1]ことに依拠している（Rivest, Shamir and Adleman［1978］）。RSA方式の安全性に関する証明（例えば、RSA方式の安全性と素因数分解問題の困難性との等価性）はこれまで示されていないものの、現在まで素因数分解よりも現実的に有効な攻撃法は報告されていない。

このため、RSA方式は、最も信頼性の高い公開鍵暗号方式として金融分野をはじめとする幅広い分野において実用化されており、ISO等の数多くの国際標準に採用されている[2]。

RSA方式における鍵生成、暗号化・復号手順は以下の通り。

＜RSA方式のアルゴリズム＞

【鍵生成】 2つの大きな素数pとqを選び、これらの積$n=pq$を計算する。$p-1$と$q-1$の最小公倍数$L = LCM(p-1, q-1)$を計算する。最大公約数$GCD(e, L)=1$を満足する自然数eを選び、$ed = 1 \pmod L$を満たすdを求める。

【秘密鍵】 d

【公開鍵】 (e, n)

【暗号化】 $C = M^e \bmod n$（ただし、Mは平文、Cは暗号文）

【復号】 $M = C^d \bmod n$

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

2 RSA方式を利用している標準規格としては、ISO/IEC 9796のほか、署名対象メッセージに署名を添付するタイプのデジタル署名方式を規定する国際標準ISO/IEC 14888（Information technology ― Security techniques ― Digital signature schemes with appendix、ISO/IEC［1998b］）や、金融業務における公開鍵暗号を利用した鍵管理方法の国際標準ISO 11166-2（Banking ― Key management by means of asymmetric algorithms ―Approved algorithms using the RSA cryptosystem、ISO［1994］）が挙げられる。また、米国政府内で利用されるデジタル署名方式を規定するFIPS 186-1（NIST［1999］）や、米国銀行業界における公開鍵暗号方式に関する標準規格ANSI X9.31（ANSI［1998］）においてもRSA署名が採用されている。

3．これまでのRSA署名の安全性に関する研究

3.1　RSA署名の分類

RSA署名を利用したデジタル署名方式は、「付録型」と「メッセージ復元型」の２つに分類することができる[3]。各方式の内容は以下の通り。

（1）付録型

付録型のデジタル署名方式は、署名生成者が署名をメッセージに添付して送信し、署名検証者は送付されたメッセージを利用して署名の正当性を検証する方式である。

付録型におけるナイーブな方式は、メッセージのハッシュ値を生成し、そのハッシュ値をRSA署名の秘密鍵dで変換することで署名を生成するというものである。ナイーブな方式における署名生成、検証手順は以下の通り。

【署名生成】メッセージMのハッシュ値 $m=H(M)$ を生成した後（Hはハッシュ関数）、以下の計算によってMに対する署名Sを生成する。

$$S = m^d \bmod n$$

SはMに添付されて受信者に送付される。

【署名検証】受信者は、以下の等式が成立するか否かを検証する。

$$H(M) = S^e \bmod n$$

また、RSA署名による署名変換の対象となるデータ（署名変換対象データ）として、単にメッセージのハッシュ値を利用するのではなく、メッセージになんらかの変換を加えたデータ、乱数、予め決められたパディングデータ等を結合させて署名変換対象データを生成する方式も提案されている。こうした方式としては、PKCS #1やPSS署名等が挙げられる。

（2）メッセージ復元型

メッセージ復元型は、署名からメッセージ全体または一部を復元可能とする方式であり、署名検証の際にはメッセージを必要とせず、署名変換対象データのフォーマットの正当性によって検証を行うものである。

RSA署名を利用したメッセージ復元型の署名方式としては、ISO/IEC 9796-1,2,3やPSS-R署名等が挙げられる。

3 本分類方法については、Menezes, Oorschot and Vanstone［1997］p. 427を参照。

3.2 RSA署名に対する安全性評価結果

デジタル署名方式の安全性を評価する場合には、どのような攻撃に対する安全性を考えるかを明確にする必要がある。攻撃法を大きく２つに分類すると、以下の表１の通り。

表１ デジタル署名方式に対する攻撃法の分類

	攻撃法の内容
受動的攻撃	公開鍵と、攻撃者が選択できないメッセージに対する署名を利用した攻撃法
能動的攻撃	公開鍵と、攻撃者が選択したメッセージに対する署名を利用した攻撃法

これらの攻撃法は、攻撃者が入手可能な署名に対するメッセージを、攻撃者自ら選択することが可能か否かによって分類される。受動的攻撃は、実現可能性が高いものの、攻撃者が利用できる情報は制限されている。一方、能動的攻撃は、攻撃者が自分の選んだメッセージに対する署名を入手できるものの、受動的攻撃と比べて実現可能性は低い。

デジタル署名方式のタイプと攻撃のタイプに沿って、RSA署名を利用した主なデジタル署名方式を分類し、これまでの安全性評価を整理すると表２の通り[4]。現時点で安全性が証明されている方式は、PSS署名とPSS-R署名だけである。以下では、安全性が証明されていない方式について説明する（PSS署名やPSS-R署名については６.において説明）。

表２ RSA署名を利用した主なデジタル署名方式の安全性評価結果

デジタル署名方式		受動的攻撃に対する安全性	能動的攻撃に対する安全性
付録型	ナイーブなRSA署名	△	Desmedt-Odlyzkoの攻撃法
	PKCS #1	△	△
	PSS署名	○	○
メッセージ復元型	ISO/IEC 9796-1	△	△[5]
	ISO/IEC 9796-2	△	CNS攻撃法
	ISO/IEC 9796-3	△	Misarskyの攻撃法
	PSS-R署名	○	○

（注）○：安全性が証明されている。
△：安全性は証明されていないものの、有効な攻撃法が提案されていない。

4 RSA署名の安全性に関するこれまでの研究に関しては、Kaliski and Robshaw［1995］やMisarsky［1998］を参照。

5 Coronらは、ISO/IEC 9796-1を一部変更したデジタル署名方式に対して有効な攻撃法を提案している（Coron et al.［1999］）。

(1) 受動的攻撃に対する安全性

受動的攻撃が可能な環境を前提にする場合、いずれのタイプのデジタル署名方式に対しても、現在最も有効な攻撃法は法nを素因数分解して秘密鍵dを計算するというものである。RSA署名の安全性は、nの素因数分解に必要となる計算量によって左右される。これまでに提案されている主要な素因数分解アルゴリズムを整理すると、以下の表３の通り[6]。

表３ 合成数nの性質と最も効率的な素因数分解アルゴリズム

合成数$n(=p \cdot q)$の性質	最も効率的な解法
$p \pm 1$または$q \pm 1$が小さな素数の積となる場合	$P+1$法、$P-1$法
$\lvert p-q \rvert$が小さな素数の積となる場合	Fermat法
上記2つの条件をいずれも満足せず、pとqのサイズが同一の場合	Adleman-Lenstra版数体ふるい法

表３は、合成数nが２つの素数pとqの積になっている場合、pとqがどのような条件の場合にどの解法が最も高速かを整理したものである。pとqのサイズが同一であり、pとqにそれぞれ特殊な性質が存在しない場合、現時点でAdleman-Lenstra版の数体ふるい法が最高速の解法である。

― RSA社は、２つの素数の積の素因数分解がどれだけ短時間で成功するかを競うコンテストを実施しており、1999年2月には、数体ふるい法を用いることにより、約2か月間を費して465 bitの合成数が素因数分解されている（300MHzのパソコンに換算して約60台を使用、計算量は約2000 MIPS年[7]）。こうしたことから、現時点では、付録型のデジタル署名方式に関する国際標準ISO/IEC 14888-3（ISO/IEC［1998b］）や金融機関において利用される情報セキュリティ技術に関するガイドラインISO/TR 13569（ISO［1997］and［1998］）等において、RSA方式等、素因数分解問題を利用する公開鍵暗号方式では、1024 bit以上の鍵長を利用することが推奨されている。

(2) 能動的攻撃に対する安全性

①ナイーブなRSA署名

ナイーブなRSA署名の場合、Desmedt-Odlyzkoの攻撃法によって、ある一定の条件を満足するメッセージの署名を偽造可能であることが示されている（Desmedt and Odlyzko［1986］）。

6 素因数分解アルゴリズムについては、宇根・岡本［1999b］を参照。

7 MIPS年：1 MIPSは1秒間に100万回の命令を実行できる計算能力を表しており、2000 MIPS年は、1 MIPSの計算能力を有する計算機によって2000年間で実行される計算量を表す。

Desmedt-Odlyzkoの攻撃法は、メッセージのハッシュ値が小さな素数の積となっている場合に有効となる。本攻撃法に必要な計算量はハッシュ値のサイズに依存し、法nのサイズには依存しない。本攻撃法の手順は以下の通り。

（ｉ）攻撃者は、メッセージのハッシュ値が比較的小さな素数に素因数分解可能となるメッセージMを選択する。ハッシュ関数をHとすると、例えば、Mのハッシュ値 $H(M)$ が $H(M)=p_1 \cdot p_2 \cdots p_{k-1} \cdot p_k$となる場合を考える（$p_i$は素数、$k$は自然数）。

（ii）攻撃者は、上記の条件を満たすMを大量に生成し、署名者からMに対する署名Sを入手する。署名Sは以下の通り。

$$S = H(M)^d \bmod n = p_1^d \cdot p_2^d \cdots p_{k-1}^d \cdot p_k^d \bmod n$$

（iii）攻撃者は、各$H(M)$が $p_1 \cdot p_2 \cdots p_{k-1} \cdot p_k$ という小さな素数の積となっていることを知っており、入手した複数の署名を使って、各素数p_iをd乗して$\bmod n$を計算した値（$p_i^d \bmod n$）を得る。

（iv）攻撃者は、入手した$p_i^d \bmod n$の値を蓄積し、それらを組み合わせることによって、ハッシュ値が小さな素数の積となる任意のメッセージに対する署名を偽造する。

本攻撃法によって、ナイーブなRSA署名は能動的攻撃に対して安全性を確保することができないことが示された。このため、能動的攻撃への対応方法として、署名変換対象データの生成にメッセージのハッシュ値を利用するという方法と、署名変換対象データに冗長性をもたせるという方法が提案された（Misarsky［1998］）。これらの対応方法を取り入れて設計されたデジタル署名方式が、後述するPKCS #1やISO/IEC 9796-1,2,3等の方式である。

②PKCS #1

PKCS #1[8]は、RSA方式を利用したデータ守秘方式およびデジタル署名方式の利用方法に関する技術仕様である。PKCS #1は、Netscape社が提唱する暗号通信、認証等のセキュリティ機能が付加されたHTTPプロトコルであるSSL等に採用されている。現在利用されているPKCS#1 Version 2.0（RSA Laboratories［1998］）[9]の署名生成方法は以下の通り。

8 PKCS（Public-Key Cryptosystem Standard）は、RSA社が策定する公開鍵暗号に関する技術仕様であり、現在PKCS #1をはじめとして12の仕様が定められている。PKCSシリーズでは、RSA方式を利用する際のデータ変換の方法から、守秘、署名、鍵管理等についてルールを設けている。ただし、PKCSはRSA社が独自に制定する技術仕様であって、国際的な標準化検討委員会等によって定められた標準ではないことに注意が必要である。

9 PKCS#1 Version 2.0の前バージョンVersion 1.5（RSA Laboratories［1993］）では、データ守秘方式としてOAEP（Optimal Asymmetric Encryption Padding, Bellare and Rogaway［1995］）とは別の方式が採用されていた。しかし、1998年6月、本方式に対して能動的攻撃が適用可能であることがBleichenbacherによって示されたほか、PKCS#1を実装しているSSL Version 3において実際に能動的攻撃が可能になることが実証された（Bleichenbacher［1998］）。このため、RSA社は、1998年9月にOAEPをデータ守秘方式とするPKCS#1 Version 2.0を発表した。OAEPは、一定条件の下で安全性が証明されている方式である。

＜PKCS #1 Version 2.0の署名生成方法＞

【公開鍵】 (e,n)（nはk byte）

【秘密鍵】 d

【署名生成手順】

メッセージをM、ハッシュ関数を$Hash$（ハッシュ値のサイズをh byte）、SRを署名変換対象データとする。

〈1〉メッセージMのハッシュ値 $H=Hash(M)$ を計算する。

〈2〉ハッシュ関数のID情報とハッシュ値Hを含むデータT（サイズはt byte）を生成する。

〈3〉パディングデータをPSとすると、PSのサイズは（$k–t–3$）byteとなり、各bitの値をすべて“１”とする。

〈4〉この結果、SRは以下のフォーマットとなり、サイズは公開鍵と同じk byteとなる[10]。

$$SR = [\,00\ 01_{16} \| PS \| 00_{16} \| T\,] = [00\ 01_{16} \| FF...FF_{16} \| 00_{16} \| T]$$

〈5〉署名生成者は、上記のSRを秘密鍵dで変換して署名Sを生成する。

$$S = SR^{d} \bmod n$$

本デジタル署名方式の安全性については証明が示されていないものの、素因数分解よりも有効な攻撃法はこれまで提案されていない。

③ISO/IEC 9796-1,2,3

ISO/IEC 9796（Information technology—Security techniques—Digital signature schemes giving message recovery）は、メッセージ復元型のデジタル署名方式の国際標準であり、メッセージを署名に埋め込むことによって交信されるデータ量を削減することが可能となっており、計算能力が比較的限定されるICカード上での実装に適した方式である。

ISO/IEC 9796は、以下の４つのパートから構成されている。

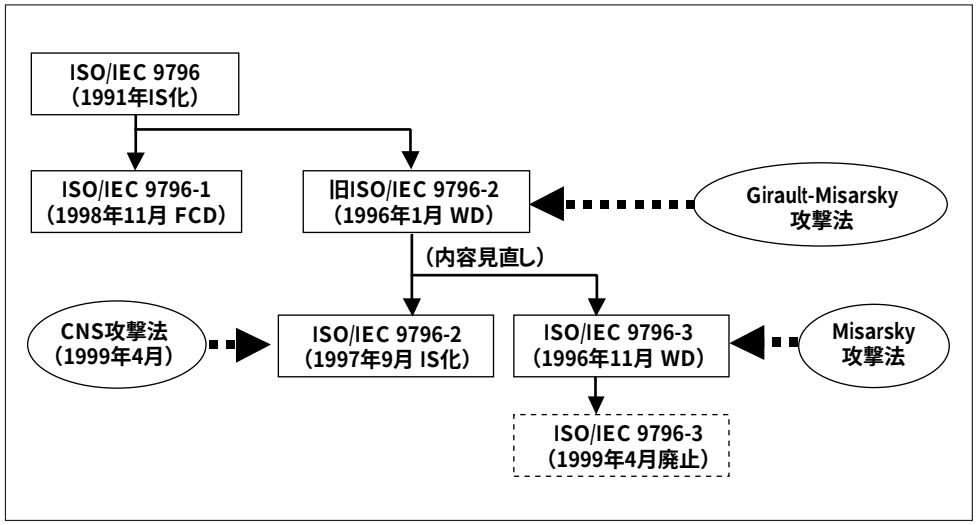
- Part 1：冗長性を利用した方式
- Part 2：ハッシュ関数を利用した方式
- Part 3：検証関数を利用した方式
- Part 4：離散対数問題を利用した方式[11]

10 以下の表記では、「01_{16}」の「16」の添え字はbyte表示であることを表し、添え字がない場合はbit表示であることを表す。すなわち、「01_{16}」はbit表示では「0000 0001」となる。

11 Part 4は、Part 1,2,3とは異なる経緯の中で標準化が進められているものであり、本稿では取り上げない。なお、ISO/IEC JTC1/SC27/WG2は、Part 3として標準化が進められていた国際標準案の廃止（後述）を決定しており、Part 4を新たにPart 3として標準化する方向で検討を進めている。

このうち、Part 1〜3のデジタル署名方式にRSA署名が利用されており、これらのデジタル署名方式の標準化の流れは以下の図１の通り（Girault and Misarsky［1997］, Misarsky［1997］and［1998］）。

図１ ISO/IEC 9796-1,2,3の標準化プロセスと攻撃法の提案



（a）ISO/IEC 9796-1のデジタル署名方式

ISO/IEC 9796-1（Information technology—Security techniques—Digital signature schemes giving message recovery—Part 1: Mechanisms using redundancy）は、1991年に国際標準となったISO/IEC 9796（Information technology—Security techniques—Digital signature schemes giving message recovery）のデジタル署名方式のリバイス版として提案されている国際標準案であり、署名生成・検証方法は同一のものである（ISO/IEC［1991］、Coron et al［1999］）[12]。

本国際標準案に規定されているデジタル署名方式は、ハッシュ関数を利用しない方式であり、署名変換対象データにメッセージ全体を埋め込み、署名からメッセージ全体を復元可能となるように設計されている（署名変換対象データのフォーマットについては補論１参照）。公開鍵のサイズが1024 bitの場合、256 bit以下の比較的短いメッセージに対してのみ利用可能となっている。

12 ISO/IEC 9796のEditorであるフランスのGuillouらは、公開鍵nよりも非常に小さな署名変換対象データを利用することによっていくつかの攻撃法が適用可能となることを指摘するとともに、ISO/IEC 9796が、それらの攻撃法に対して十分な安全性を確保できるように設計されていることを示した（Guillou et al.［1991］）。

（b）ISO/IEC 9796-2のデジタル署名方式

ISO/IEC 9796-2（Information technology—Security techniques—Digital signature schemes giving message recovery—Part 2: Mechanisms using a hash-function）は、署名からメッセージ全体または一部を復元可能なデジタル署名方式の国際標準であり、1996年1月にSC27において標準化が開始され、1997年9月に完了している（ISO/IEC［1997］）。

標準化が開始された当初、WD[13]として提案されたISO/IEC 9796-2（以下、旧ISO/IEC 9796-2と呼ぶ）には、署名検証のために剰余演算を利用した関数（$H(M) = 2(M \bmod (2^{79}+1))$）が用いられていたが、この関数の性質を利用した攻撃法がフランステレコムのGiraultとMisarskyによって発表された（Girault and Misarsky［1997］）。このため、旧ISO/IEC 9796-2で利用されていた剰余演算の関数の代わりにハッシュ関数が採用され、新しいISO/IEC 9796-2の標準化が進められた。

ISO/IEC 9796-2では、ハッシュ関数を利用することにより、ISO/IEC 9796-1と比べて利用可能なメッセージサイズを拡大することができる。公開鍵のサイズが1024 bitの場合、署名から復元可能なメッセージは最大848 bitとなる（署名変換対象データのフォーマットについては補論１参照）。

（c）ISO/IEC 9796-3のデジタル署名方式

ISO/IEC 9796-3（Information technology—Security techniques—Digital signature schemes giving message recovery—Part 3: Mechanisms using a check-function）は、旧ISO/IEC 9796-2に対する攻撃法として発表されたGirault-Misarskyの攻撃法に対して安全性を確保できるように設計されたデジタル署名方式の国際標準案である（Misarsky［1998］）。本国際標準案は、1996年11月にWDとしてSC27に提案された。

本デジタル署名方式における署名変換対象データの生成方法の特徴は、$C=M \bmod (2^{79}+1)$という検証式によって生成される検証コードCを複数の比較的小さなデータに分割し、署名変換対象データの特定の位置に埋め込む点である（署名変換対象データのフォーマットについては補論１参照）。検証コードの埋め込みによって、署名変換対象データが複数の部分に分割され、Girault-Misarskyの攻撃法に対する安全性が高まるとされている。

しかし、Misarskyは、本デジタル署名方式に対する攻撃法を発表し、一定の条件[14]の下で、公開鍵の素因数分解よりも効率的に署名を偽造可能であることを示した（Misarsky［1997］）。本攻撃法のアイデアは以下の通り。

13 WD（Working Draft）：新しい国際標準の提案（New Work Item Proposal）がISOの専門委員会または分科委員会において行われ、それが投票によって承認された後、作業グループが作成した国際標準規格の作業原案。ISOにおける国際標準化プロセスの詳細については、岩下・谷田部［1999］を参照。

14 Misarskyの攻撃法をISO/IEC 9796-3に適用するための条件は、公開鍵のサイズをN、メッセージのサイズをmとする場合、$(N/2) - 80 < m$が成立することである。例えば、公開鍵のサイズが1024 bitの場合には、メッセージのサイズが433 bit以上のときに本攻撃法が適用可能となる。

（ｉ）メッセージMに対する署名変換対象データを$U(M)$とする。攻撃者は、署名偽造の対象であるメッセージMに対して、$U(M)\cdot U(X) = U(Y) \pmod n$を満足するデータ$X$と$Y$を見つける。

（ii）攻撃者は、XとYを正当な署名者に送付し、署名者からXとYに対する署名$U(X)^d \bmod n$と$U(Y)^d \bmod n$を入手する。

（iii）$U(M)\cdot U(X) = U(Y) \pmod n$を変形すると、$U(M)^d \cdot U(X)^d = U(Y)^d \pmod n$となり、$U(M)^d = (U(Y)^d / U(X)^d) \pmod n$が成立する。このため、攻撃者は入手した２つの署名を上記等式の右辺に代入し、メッセージMに対する署名$U(M)^d \bmod n$を偽造する。

Misarskyは、本攻撃法をISO/IEC 9796-3に適用した場合（公開鍵のサイズを640 bit、メッセージのサイズを512 bitに設定）、Pentium 166MHzのパソコンを利用して約30分で署名の偽造に成功したとの実験結果を発表している（Misarsky［1997］）。

本攻撃法が発表されたことを受けて、ISO/IEC JTC1/SC27は、1999年4月にISO/IEC 9796-3の標準化作業を中止し、ISO/IEC 9796-3を廃止することを決定している。

４．CNS攻撃法の内容と特徴

4.1　攻撃法の基本的な手順

CNS攻撃法は、署名変換対象データが小さな素数の積となるメッセージを大量に集め、それらのメッセージを正当な署名者に送信して署名を生成してもらい、入手した署名から別のメッセージの署名を偽造するというものである（Coron et al.［1999］）[15]。

Coronらは、本攻撃法をISO/IEC 9796-2に適用可能であり、公開鍵が1024 bitの場合には、必要となる署名を入手した上で、176 bitの合成数の素因数分解を実行することによって、別のメッセージの署名を偽造できることを示している。

本攻撃法のアイデアを説明すると以下の通り（詳しい説明は補論２を参照）。

15 べき乗剰余演算によって変換されたデータを解読ないしは偽造する際に、そのデータが比較的小さな素数の積となっていることを利用するアイデアとしては、1988年に神戸大学の田中教授が発表したID情報に基づく鍵配送方式に対する攻撃法が挙げられる（田中［1988］）。田中教授は、同論文の中で、本攻撃法がRSA公開鍵暗号系にも適用可能であることを指摘しているが、このアイデアに基づいた具体的な攻撃法はこれまで提案されていなかった。

＜以下の説明に利用される記号の意味＞

・U：メッセージMから署名変換対象データ$U(M)$を生成する関数。$U(M)$のサイズは公開鍵のサイズと同一となる。
・L：攻撃の際に素因数分解が必要となる数値のサイズ（bit数）。ISO/IEC 9796-2への攻撃の場合には176 bitとなる。
・p_k：最小の素数2から順番に数えてk番目に大きな素数
・(n, e)：RSA署名の公開鍵（ただし$n = p \cdot q$、pとqは素数）
・d：RSA署名の秘密鍵

①攻撃者はp_kを設定した上で、署名変換対象データ$U(M)$がp_k-smooth[16]となるメッセージMを大量に集める。
②攻撃者は選択したメッセージMを正当な署名者に送信し、各Mに対する署名$U(M)^d \bmod n$を返信してもらう。
③攻撃者は入手した$U(M)^d \bmod n$を利用して、メッセージM'に対する署名$U(M')^d \bmod n$を生成する。ただし、M'は、$U(M')$がp_k-smoothとなるようなメッセージである。

4.2 攻撃法の特徴点

CNS攻撃法の主な特徴点をまとめると、以下の４点になる。

① CNS攻撃法は能動的攻撃である。

― 本攻撃法は、「攻撃者が自分の都合のよいメッセージに対する署名を入手できる」という実装環境において有効であり、本攻撃法が現実的な脅威となり得るか否かは攻撃対象となるデジタル署名方式の実装環境に依存する。

②署名を偽造することができるメッセージには制約が存在し、攻撃者が署名を偽造するメッセージを自由に選択することはできない。

― 本攻撃法によって署名を偽造可能なメッセージMは、Mの署名変換対象データ$U(M)$がp_k-smoothとなる必要があるため、偽造対象のメッセージを自由に選択することはできない。ただし、本攻撃法によって意味のあるメッセージ

16 p_k-smooth：全ての素因数がp_k以下となる合成数を「p_k-smooth」という。p_k-smoothな合成数uは、

$$u = \prod_{j=1}^{k} p_j^{v_j}$$

と表される。ただし、$p_j \leqq p_k$（$j = 1,...,k$）である。

に対する署名を偽造できる可能性は残されていることから、本攻撃法が現実的な脅威となる可能性もある。

③攻撃に必要な計算量のオーダーは、

$$O\left(\frac{Lk(\log e)\sqrt{k\ln k}}{\rho(L/\log_2(k\ln k))}\right)$$

となる。また、必要なメッセージ数のオーダーは$O(k(\log e))$となる。

— 攻撃に必要な計算量は、L、k、公開鍵eに依存する一方、公開鍵nのサイズには依存しない。このため、法nのサイズを大きくしても、本攻撃法に対する安全性は向上しない。ISO/IEC 9796-2に本攻撃法を適用する場合、SHA-1の利用を前提としたときには176 bitの合成数の素因数分解によって攻撃可能となり、1024 bitの公開鍵を素因数分解する攻撃法に比べて大幅に計算量を削減することができる。
— 攻撃に必要なメッセージ数もkとeのみに依存することから、公開鍵のサイズに依存しない。なお、最も単純なケースとしてeが素数の場合、攻撃に必要なメッセージ数は$k+1$個となる。

④予め設定されたkとeに対して、攻撃に必要な数の署名を入手することができれば、$U(M)$がp_k-*smooth*となるメッセージMに対する署名を、複数偽造することができる。

— ただし、ISO/IEC 9796-2に本攻撃法を適用する場合には、攻撃に利用するMに公開鍵nの一部が含まれているため、nが変更されると、それまでに入手していたMに対する署名を使って、追加的に別の新たな署名を偽造することはできなくなる。

4.3 攻撃法の適用例

Coronらは、本攻撃法がISO/IEC 9796-2に適用可能であることを示したほか、署名変換対象データの生成方法に一部修正を加えたISO/IEC 9796-1（詳細な説明については6.を参照）や、PKCS #1 Version 2.0、ANSI X9.31に対して、本攻撃法を適用した場合の必要計算量やメッセージ数を試算している。

ただし、署名変換対象データの生成方法に何ら変更を加えないISO/IEC 9796-1に対しては、本攻撃法を適用することができない（Coron et al.［1999］）。

また、PKCS #1 Version 2.0やANSI X9.31に対して本攻撃法を適用できるのは公開鍵nが特殊な形態をしている場合のみであり、たとえ適用できたとしても、攻撃に必要となる計算量は素因数分解よりも多くなる。このため、これらのデジタル署名

方式は、本攻撃法に対して十分な安全性を有している（Silverman and Naccache［1999］）。

― Coronらは、論文（Coron et al.［1999］）の中で、「PKCS #1 Version 2.0やANSI X9.31のデジタル署名方式に対しては、本攻撃法は、公開鍵nが$n=2^k \pm c$となる特殊なケースにおいてのみ適用できる。したがって、これらのデジタル署名方式を実装する際に、$n=2^k \pm c$となる公開鍵を利用しないようにすることで本攻撃法に対して安全性を確保できる。なお、ANSI X9.31には、既に$n=2^k \pm c$となる特殊な公開鍵の利用を避けるように明記されている」と指摘している。

5．ISO/IEC 9796-2に対する攻撃法

以下では、公開鍵およびメッセージのサイズが1024 bit、ハッシュ関数としてSHA-1（ハッシュ値が160 bit）を利用するケースを例に、ISO/IEC 9796-2に対する攻撃法を説明する（Coron et al.［1999］）。

5.1　攻撃の手順

（1）公開鍵nの形態

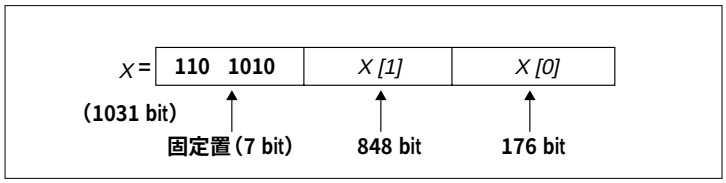
本攻撃法のアイデアは、「公開鍵nと署名変換対象データ$U(M)$の関数$F(n, U(M))$を用いて、$U(M)$から素因数分解が比較的容易な小さな合成数を生成し、その合成数がp_k-*smooth*となるメッセージMを利用する」というものである。

ISO/IEC 9796-2の署名変換対象データ$U(M)$は、左から7 bit分が“110 1010”となる。仮に公開鍵nの左から7 bit分が“110 1010”となる場合（確率は$1/2^7$）、Mの左から848 bit分を一定の値に定めた上で、関数$F(n, U(M)) = n-2U(M)$を利用することによって、素因数分解の対象となるデータ$n-2U(M)$のサイズをハッシュ値のサイズと同程度の169 bitにすることができる。

また、nの左7 bit分が“110 1010”となっていない場合には、適当な数値aを選択してnをa倍することにより、左から7 bit分が“110 1010”となる1031 bit（= 1024 bit +7 bit）のデータ$X(=a \cdot n)$を生成することができる。その上で、Mの左から848 bit分を一定の値に定め、関数$F(n, U(M)) = a \cdot n - 2^8 \cdot U(M) = X - 2^8 \cdot U(M)$を利用することによって、素因数分解の対象となるデータ$X - 2^8 \cdot U(M)$のサイズを176 bitにすることができる。

Xのサイズは1031 bitであり、そのフォーマットは以下の図２の通り。Xの右から176 bitの部分を$X[0]$とし、左7 bit“110 1010”と$X[0]$の間の部分（848 bit）を$X[1]$とする。

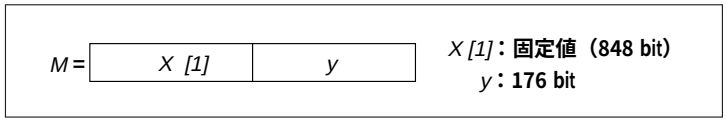
図2　Xのフォーマット



(2) 署名者に送付するメッセージの生成

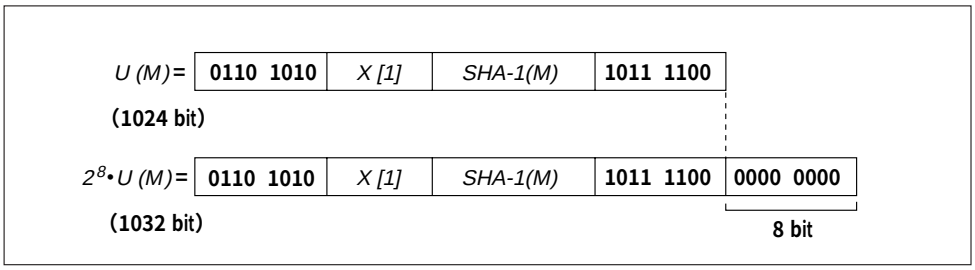
攻撃に利用するメッセージMは、左848 bitの部分が$X[1]$と等しくなるように設定される。残りの176 bitをyとする。Mの取り得る値の総数は2^{176}個となる（図3参照）。

図3　メッセージの形態



このようなMの署名変換対象データ$U(M)$を計算し、さらに$2^8 \cdot U(M)$を計算すると図4の通り（詳細については補論1参照）。$2^8 \cdot U(M)$は$U(M)$の各bitが8つ左にシフトし、右から8 bitだけ“0”が付け加えられる。$2^8 \cdot U(M)$のサイズは1032 bitとなる。

図4　$U(M)$と$2^8 \cdot U(M)$の値

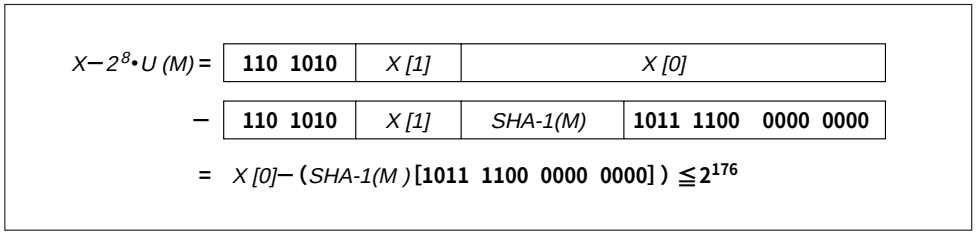


〈$2^8 \cdot U(M)$の構成〉

- ［Header, More-data bit, Padding field］＝［0110 1010］(8 bit)
- ［Data field］＝$X[1]$（848 bit）
- ［Hash field］＝$SHA\text{-}1(M)$（160 bit）
- ［Trailer］＝［1011 1100］(8 bit)
- 追加されるデータ：［0000 0000］(8 bit)

次に　$(X - 2^8 \cdot U(M))$を計算する。$2^8 \cdot U(M)$の左1 bitは“0”であることから、最初の855 bitのデータ（［110 1010］$X[1]$）がキャンセルアウトされるため、計算結果である（$X[0] - (SHA\text{-}1(M)$［1011 1100 0000 0000］)）は2^{176}以下となる（図5参照）。

図５　$X-2^8 \cdot U(M)$ の値



こうして得た（$X-2^8 \cdot U(M)$）が p_k-*smooth*となるようにyを選ぶ。

$$X - 2^8 \cdot U(M) = \left(\prod_{j=1}^{k} p_j^{v_j} \right)$$

選択したyに対して（$X-2^8 \cdot U(M)$）がp_k-*smooth*となっていることを確認するためには、（$X-2^8 \cdot U(M)$）を実際に素因数分解する必要があるが、（$X-2^8 \cdot U(M)$）のサイズ（Lの値）は176 bit以下であるため、公開鍵のサイズ1024 bitに比べて非常に少ない計算量で素因数分解を行うことができる。

こうして入手したMを署名者に送付し、Mに対応する署名$U(M)^d \bmod n$を入手する。

（3）署名の偽造

入手した署名$U(M)^d \bmod n$を利用して、別のメッセージM'に対応する署名$U(M')^d \bmod n$を偽造する（詳細は補論２参照）。

5.2　攻撃に必要となる計算量とメッセージ数

本攻撃法において必要な計算量やメッセージ数は、kおよびeに依存する[17]。公開鍵nのサイズを変化させたとしても、（$X-2^8 \cdot U(M)$）を計算することによって、素因数分解が必要となる合成数（$X-2^8 \cdot U(M)$）のサイズは常に176 bit以下となることから、本攻撃法に必要となる計算量は公開鍵のサイズに依存しない。

ISO/IEC 9796-2への攻撃は、Mの中に公開鍵の一部が含まれていることから、公開鍵が変更されると、それ以前に生成していたMを別の署名の偽造に利用することができなくなる。このため、本攻撃法に対する運用面での対応策として、公開鍵を頻繁に変更するという方法が有効であると考えられる。

Coronらは、本攻撃法をISO/IEC 9796-2のデジタル署名方式に適用する場合に必要となる計算量とメッセージ数を試算している。その結果、SHA-1を利用した場合、2^{61}のオーダーの計算量と2^{40}のオーダーのメッセージが必要となることが示されている（表４参照）。

17　Lの値は、ハッシュ値が160 bitであることから、176 bitで固定されている。

表４　CNS攻撃法をISO/IEC 9796-2に適用した場合の必要計算量の試算

ハッシュ長（bit）	計算量が最小となるkのサイズ（bit）	必要計算量のオーダー	必要メッセージ数のオーダー
64	15	2^{47}	2^{30}
80	17	2^{51}	2^{34}
128	18	2^{54}	2^{36}
160	20	2^{61}	2^{40}

（出典）Coron et al.［1999］

6. ISO/IEC 9796-1に変更を加えたデジタル署名に対する攻撃法

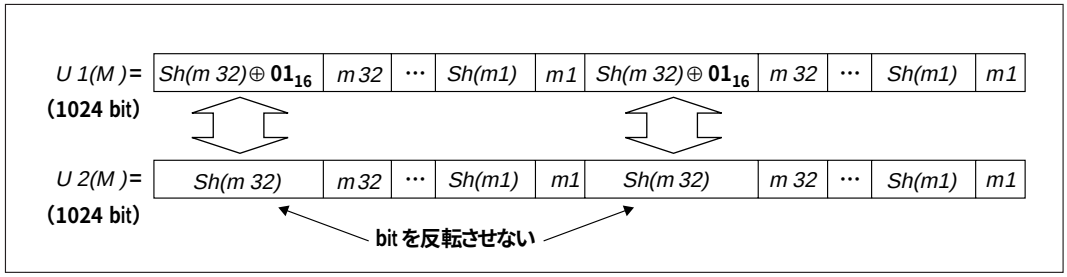
Coronらは、CNS攻撃法が、ISO/IEC 9796-1に変更を加えたデジタル署名方式（以下、変更版ISO/IEC 9796-1と呼ぶ）に対しても適用可能であることを示している（Coron et al.［1999］）。変更版ISO/IEC 9796-1に対する攻撃法は、ISO/IEC 9796-1の安全性を評価する上で有用と考えられるため、以下にその概要を説明する。

6.1　ISO/IEC 9796-1と変更版ISO/IEC 9796-1との差異

ISO/IEC 9796-1では、署名変換対象データを生成する際に、一定のルールに基づいて特定のbitの値を反転させる（１との排他的論理和を計算する）演算が利用されている（詳細については補論１を参照）。Coronらの攻撃は、このbit反転を行わないようにした変更版ISO/IEC 9796-1に対してのものである。

例えば、公開鍵のサイズが1024 bit、メッセージのサイズが256 bitの場合、この変更により、署名変換対象データの2 bit分の値に差異が発生することになる（図６参照）。

図６　2つの署名変換対象データのフォーマットの差異



― 図６の例では、メッセージMは$M = [m32, m31, ..., m2, m1]$となっており、各$mi$（$i=1,...,32$）は8 bitのデータである。また、$U1(M)$がISO/IEC 9796-1における署名変換対象データ、$U2(M)$が変更版ISO/IEC 9796-1の署名変換対象デー

タである。なお、図６において明示していないが、$U1(M)$および$U2(M)$ともに、左から1 bitが“１”に固定されるほか、右から4 bitが“0110”に固定される。*Sh*は8 bit入出力の換字変換である。

6.2　攻撃の手順

以下では、公開鍵のサイズが1024 bit、メッセージのサイズが256 bitの場合について説明する。

（1）攻撃法のアイデア

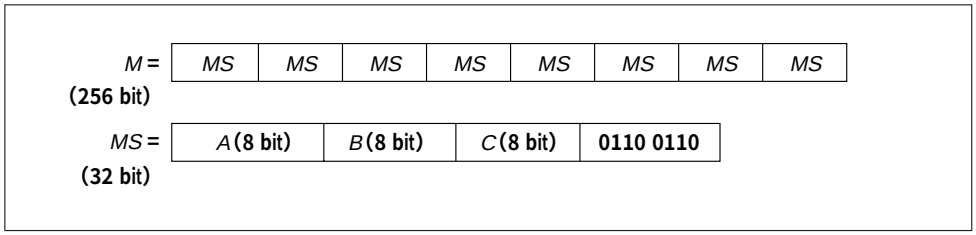
変更版ISO/IEC 9796-1にCNS攻撃法を適用する場合には、署名変換対象データのbitパターンが、ほぼ同一のbitパターンの繰り返しとなる点を利用する。具体的には以下の手順で攻撃を実行する。

（a）以下の２つの条件を満足するようにメッセージMのフォーマットを決める。
　（i）署名変換対象データ$U(M)$のbitパターンが、あるデータmのbitパターンの繰り返しとなる（すなわち、$U(M)=[m,m,...,m]$となる）。
　（ii）繰り返されるデータmのサイズは、mの素因数分解が容易に実行できるように小さくなる。
（b）mがp_k-*smooth*となるMの値を見つける（mのサイズは素因数分解可能なほど小さいため、mがp_k-*smooth*となっていることを確認することは容易）。Mを署名者に送付してその署名を入手し、別のデータの署名偽造に利用する。

（2）署名者に送付するメッセージの生成

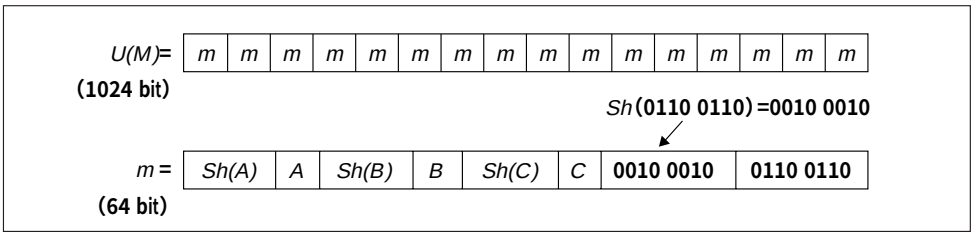
攻撃に利用するメッセージMのフォーマット（図７参照）は、上記（a）の２つの条件を満足するように、32 bitデータMS（$=[A, B, C,$ 0110 0110$]$）を８個結合したものとする（A、B、Cは各々8 bitのデータであり、攻撃者が選択可能）。ただし、Aを選ぶ際には、「$Sh(A)$の左から1 bitが常に“１”となる」という条件が加わる。

図７　攻撃に利用するメッセージのフォーマット



この結果、署名変換対象データ$U(M)$は、データmが16個結合する構造となる（図８参照、詳細については補論１参照）。mは64 bitであり、$U(M)$は1024 bitである。

図8　署名変換対象データフォーマット



この結果、$U(M)$はmによって、$U(M)=m\times\Gamma$と表される。ただし、

$$\Gamma=2^{15\times 64}+2^{14\times 64}+\cdots+2^{2\times 64}+2^{64}+1=\sum_{i=0}^{15}2^{64i}$$

次に、mがp_k-*smooth*となるようにA、B、Cの値を定める。mがp_k-*smooth*となっていることを確認するためにはmの素因数分解を行う必要があるが、mのサイズ（Lの値）は64 bitであり、公開鍵のサイズ1024 bitに比べて非常に小さいため、mの素因数分解は比較的容易に実行できる。

mがp_k-*smooth*となるようなMを見つけると、$U(M)$は以下の等式を満足する。

$$U(M)=\left(\prod_{j=1}^{k}p_j^{v_j}\right)\times\Gamma$$

攻撃者は、こうして入手したメッセージMを署名者に送付し、署名$U(M)^d \bmod n$を入手する。

(3) 署名の偽造

署名$U(M)^d \bmod n$を利用して、別のメッセージM'に対応する署名を偽造する（補論2参照）。ただし、M'は$U(M')$が以下の形式を満足するものに限定される。

$$U(M')=\left(\prod_{j=1}^{k}p_j^{v_j}\right)\times\Gamma$$

6.3　攻撃に必要となる計算量とメッセージ数

本攻撃法では、素因数分解が必要なデータのサイズ（Lの値）が64 bitとなり、必要となる計算量やメッセージ数はkやeによって変化する。公開鍵nのサイズが変化したとしても、$U(M)$に含まれるmの数が変化するだけであり、mのサイズは64 bitで変化しない。このため、公開鍵のサイズは本攻撃法に必要となる計算量に影響を与えない。

Coronらは、本攻撃法による署名を実際に偽造する実験を行っている。その結果、Pentium 200MHzを搭載したパソコンを約１日費して、$U(M)$がp_k-*smooth*となるメッセージMを15830個のみつけ、12829個のメッセージに対する署名の偽造に成功している（表５参照）[18]。

表５　CNS攻撃法による署名偽造実験の結果

kの値	攻撃に利用したメッセージ数	偽造した署名数
345	346	1
500	799	298
1000	3203	2202
1500	6198	4697
2000	9344	7343
2500	12555	10054
3000	15830	12829

（出典）Coron et al.［1999］

７．RSA署名を利用した安全性証明付きのデジタル署名方式

ISO/IEC 9796-2に規定されているデジタル署名方式は安全性が証明されておらず、実際にCNS攻撃法が適用可能となった。しかし、近年、RSA署名を利用したデジタル署名方式の中でも、安全性が証明されている方式がいくつか提案されている[19]。付録型のデジタル署名方式としてPSS署名が提案されているほか、メッセージ復元型としてPSS-R署名が提案されている。

7.1　PSS署名

PSS（Probabilistic Signature Scheme）署名は、BellareとRogawayによって提案された方式である（Bellare and Rogaway［1996］）。PSS署名は、RSA署名を利用した付録型の方式であり、以下の２つの仮定の下で、「能動的攻撃が可能な環境であったとしても、いかなるメッセージに対する署名も偽造することができない」ことが証明されている。

18　なお、Coronらの論文（Coron et al.［1999］）には、本実験において利用された公開鍵の一部eについて詳細な情報が記載されていないものの、$k=345$の場合、$346\ (=k+1)$個のメッセージによって1つの署名を偽造することが可能となっていることから（表５参照）、必要なメッセージ数は$k+1$となっていることがわかる。

19　安全性が証明されている公開鍵暗号方式については、宇根・岡本［1999a］を参照。

仮定１：入力値に対してランダムな数値を出力する理想的なランダム関数が利用できる（ランダムオラクルモデル）[20]

仮定２：RSA暗号関数が一方向性を有している[21]

PSS署名では、メッセージや乱数を理想的なランダム関数によってランダム化した上で結合し、さらに検証値として“０”を1 bitだけ左から結合して署名変換対象データを生成する仕組みとなっている（図９参照）。このように、署名変換対象データを左1 bitを残して完全にランダム化することによって、能動的攻撃に対して攻撃の手掛かりを与えないように設計されている。

また、PSS署名の署名生成・検証に必要な計算量は、RSA署名の計算量に、１回のデータ圧縮変換と２回のデータ拡張変換が追加されるのみであり、RSA署名と同程度の実用性を有している。

＜PSS署名の署名生成・検証方法＞

【鍵生成】秘密鍵と公開鍵の生成方法はRSA方式と同じ。

- h ：ランダム関数（出力値k_1 bit）
- g_1：ランダム関数（入力値k_1 bit、出力値k_0 bit）
- g_2：ランダム関数（入力値k_1 bit、出力値$k-k_0-k_1-1$ bit）

【署名生成】平文をMとし、k_0 bitの乱数rを生成して以下の計算①～③によってデジタル署名Sを生成。

①$w = h(M \| r)$、②$R = g_1(w) \oplus r$

③$S = X^d \bmod n$（ただし、$X = (0 \| w \| R \| g_2(w))$）

【署名検証】 $X = S^e \bmod n$を計算し、$X = (b \| y \| z \| \alpha)$とする（$b$：1 bit、$y$：$k_1$ bit、z：k_0 bit、α：残りのbit）。

$R' = g_1(y) \oplus z$を計算し、次の等式がすべて成立するかを検証。

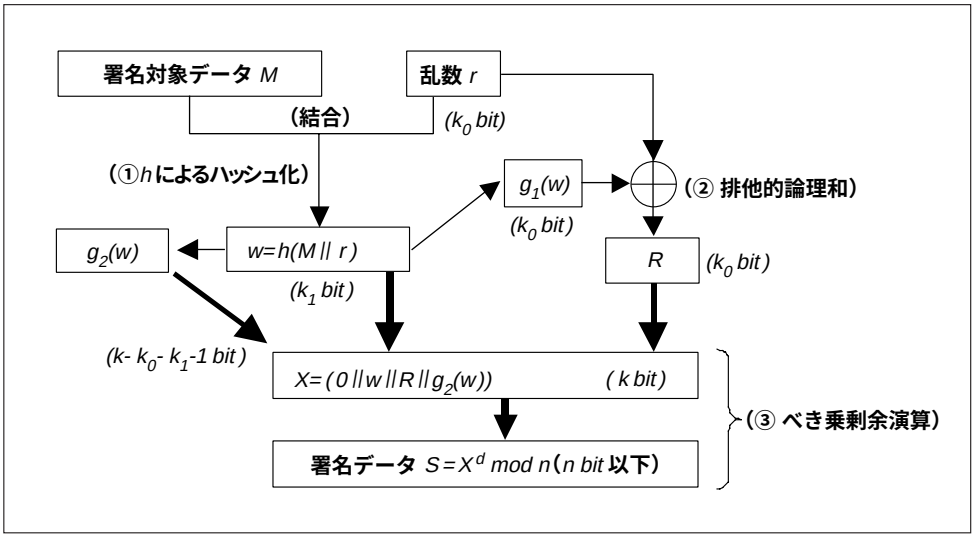
$h(M \| R') = y$、$g_2(y) = \alpha$、$b = 0$

成立すれば署名が真正であることが確認される。そうでない場合には署名を受け付けない。

20 ランダム関数は仮想的な仮定であり、ランダム関数をハッシュ関数で代用した場合には証明が成立しなくなるという問題が残されている。

21 RSA暗号の公開鍵（e, n）および$n-1$以下の自然数Yが与えられた場合、$Y = x^e \bmod n$を満足するxを求めることが計算量的に困難であるとき、「RSA暗号関数は一方向性を有する」と呼ばれる。本命題が正しいか否かについては、現時点では証明されていない。

図９　PSS署名の署名生成手順の概念図



7.2　PSS-R署名

PSS-R署名は、PSS署名のアルゴリズムを、署名からメッセージを復元可能なように改良した方式である（Bellare and Rogaway［1996］）。PSS-R署名の安全性はPSS署名と同様であり、前述の２つの条件の下で能動的攻撃に対して安全であることが証明されている。

PSS-R署名がPSS署名と異なる点は、署名変換対象データXのうち、$g_2(w)$の代わりに$g_2(w) \oplus M$が利用されている点である。署名検証の際に、$g_2(w) \oplus M$に$g_2(w)$の排他的論理和を計算することで、Mを復元することができる。

署名生成および検証に必要となる計算量については、PSS署名に１回の排他的論理和演算が追加されるだけであるが、メッセージのサイズを大きくすると署名変換対象データXが拡大し、べき乗剰余演算の計算量が増加する。このため、実際の処理速度はメッセージのサイズにも依存する。

８．CNS攻撃法による標準化活動等への影響

ISO/IEC 9796-1,2の標準化を担当しているISO/IEC JTC1/SC27/WG2では、現在、CNS攻撃法への対応方法を検討し始めている。CNS攻撃法を発表したCoron、Naccache、SternもSC27/WG2の活動に協力するとしており[22]、SC27/WG2においてどのような判断が下されるのか、注目されるところである。

22　Silverman and Naccache［1999］を参照。

また、CNS攻撃法によって影響を受ける可能性のある業界標準として、Europay、MasterCard、Visaが共同で作成したICカードに関する標準規格EMV'96が挙げられる（Europay et al.［1996］）。EMV'96では、ANNEX Eにおいて、ISO/IEC 9796-2における署名生成・検証方法がそのまま記述されている。EMV'96に準拠したICカードにおけるデジタル署名においては、本手順が利用されることとなっている。したがって、仮に、今後ISO/IEC 9796-2の改訂が行われることになれば、EMV'96を利用するシステムへの影響が生じる可能性がある。

9．おわりに

ISO/IEC 9796-2のデジタル署名方式は、主にICカードにおける実装を想定して設計されており、メッセージ復元機能を有する特殊な署名変換対象データのフォーマットが採用されていた。CNS攻撃法は、こうした署名変換対象データのフォーマットの特徴を利用した攻撃法であり、暗号アルゴリズム自体が高い信頼性を有していたとしても、利用方式によって安全性が損われる可能性があることを改めて示唆するものである。

デジタル署名をはじめとする暗号技術の実装方法の多様化が進む中、1998年6月にSSLに対して能動的攻撃が可能となることがBleichenbacherによって示されたように[23]、CNS攻撃法が現実化する可能性は否定できない。このため、デジタル署名方式を実装する際には、その実装環境を十分に考慮した上で安全性を評価する必要がある。

また、能動的攻撃に対する安全性を高める方法として、PSS署名やPSS-R署名といった安全性が証明されているデジタル署名方式を採用することも検討する必要がある。安全性が証明されているデータ守秘方式としてはOAEPやEPOC[24]が提案されているほか、デジタル署名方式としてはTSH-ESIGN[25]が提案されている。OAEPは一部の技術仕様に既に採用されており、デジタル署名方式の分野においても、安全性が証明されている方式を採用する動きが広がる可能性がある。ただし、こうした暗号方式は、一定の仮定の下での安全性を証明するものであり、実装環境次第でそうした仮定が満足されなくなる可能性もある点には留意が必要である。

このように、実装環境を含めた暗号方式の安全性に関する評価は非常に重要である。今後も、安全性証明等の理論研究と実装技術に関する研究の両者の動向について注視していく必要があろう。

23 詳細については、Bleichenbacher［1998］、宇根・岡本［1999a］を参照。
24 EPOCについては、Okamoto, Uchiyama and Fujisaki［1998］、宇根［1998］を参照。
25 TSH-ESIGNについては、Okamoto, Fujisaki and Morita［1998］を参照。

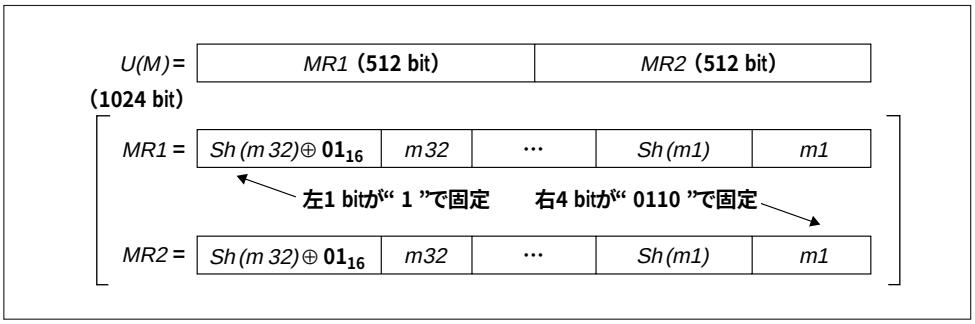
補論1：ISO/IEC 9796-1,2,3の署名変換対象データのフォーマット

1. ISO/IEC 9796-1の署名変換対象データ

ISO/IEC 9796-1では、メッセージ全体を署名から復元可能にするため、比較的短いメッセージのみ利用可能となっている。公開鍵のサイズを1024 bitとする場合、利用可能なメッセージのサイズは256 bit以下となる。以下では、メッセージのサイズが256 bit、公開鍵のサイズが1024 bitのケースを想定して説明する（ISO/IEC［1991］）。

署名変換対象データ*U*（*M*）のサイズは公開鍵のサイズと同一の1024 bitとなり、図10に表されるフォーマットとなる。ただし、メッセージ*M*を8 bitごとに分割し、右から*m1, m2, ..., m31, m32*とする。

図10 ISO/IEC 9796-1のデータフォーマット



このように、署名変換対象データ*U*（*M*）は*MR1*と*MR2*（いずれも512 bit）によって構成されている。

*MR1*と*MR2*には、8 bitごとに分割されたメッセージ*mi*（*i* =1,...,32）と、*mi*を換字変換*Sh*（入出力値はともに8 bit）によって変換したデータ*Sh*（*mi*）が含まれており、これら２種類のデータが交互に配置されている。ただし、*m32*に対しては、換字変換したデータ*Sh*（*m32*）と“ 0000 0001 ”との排他的論理和が計算される。

最後に、*MR1*のデータについては、左から1 bitが“ 1 ”に固定されるほか、*MR2*のデータについては、右から4 bitが“ 0110 ”に固定される。

このように生成された*U*（*M*）に対して、秘密鍵*d*による以下の変換によって署名*S*が生成される。

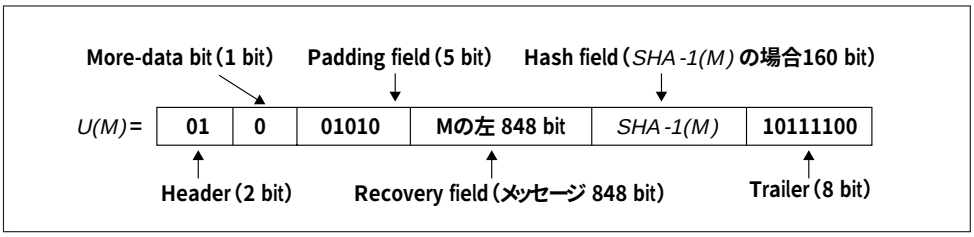
$$S = U(M)^{d} \bmod n$$

2. ISO/IEC 9796-2の署名変換対象データ

ISO/IEC 9796-2では、署名変換対象データを生成する際にハッシュ関数[26]が利用されており、ISO/IEC 9796-1よりも大きなサイズのメッセージの署名を生成可能である。例えば、公開鍵が1024 bitの場合、848 bitまででであればメッセージ全体を復元することが可能である。これ以上のサイズのメッセージに対する署名を生成する場合、メッセージのうち左から848 bit分のデータに対する署名を生成し、残りは署名に添付して送信するという方法が採用されている。

以下では、公開鍵およびメッセージが1024 bit、ハッシュ関数にSHA-1（ハッシュ値は160 bit）を利用する場合を想定し、署名変換対象データ$U(M)$のフォーマットを説明する。署名変換対象データは公開鍵のサイズと同一で1024 bitとなり、フォーマットは図11の通り（ISO/IEC［1997］）。

図11　ISO/IEC 9796-2のデータフォーマット



$U(M)$は以下の６つのパートから構成される。

（ｉ） Header：$U(M)$の左から2 bit分のデータであり、常に“ 01 ”となる。
（ii） More-data bit：Headerに続く1 bitのデータ。この例では、メッセージ全体が復元できないため、“ 1 ”となる。
（iii） Padding field：$U(M)$を1024 bitにするためのパディングデータ。この例では5 bit（=1024 bit–2 bit［Header］–1 bit［More-data bit］– 848 bit［Recovery field］–160 bit［Hash field］– 8 bit［Trailer］）となり、“ 01010 ”となる。
（iv） Recovery field：メッセージMのうち左から848 bit分のデータが入る。残りの176 bit（=1024 bit – 848 bit）分のデータは署名とともに送付される。
（v） Hash field：メッセージMに対するハッシュ値$SHA\text{-}1(M)$が入る。ハッシュ関数としてSHA-1を想定しているため、ハッシュ値のサイズは160 bitとなる。
（vi） Trailer：ハッシュ関数の属性を表すデータ。SHA-1を利用することを想定しているため、8 bitのデータ“ 1011 1100 ”となる。

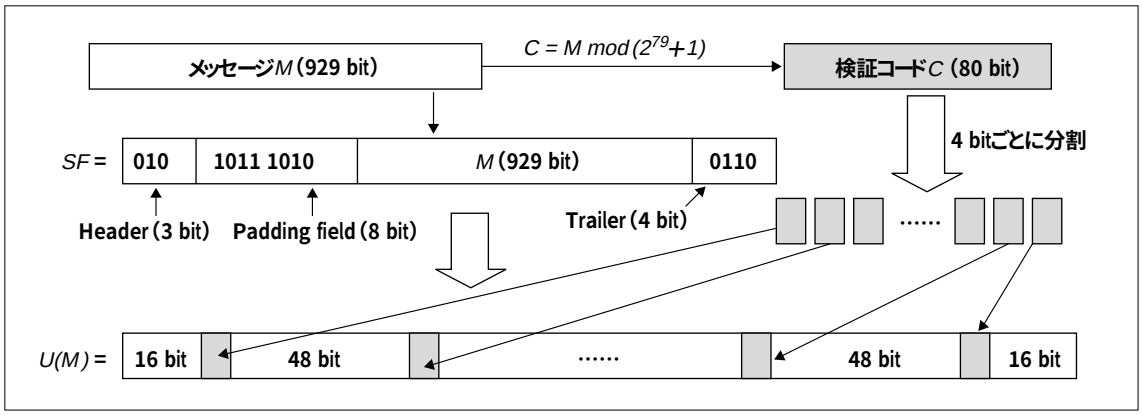
26 本標準規格において利用されるハッシュ関数としては、ISO/IEC 10118（Information technology — Security techniques — Hash-functions、ISO/IEC［1994］,［1998b］）に規定される方式を利用するよう記載されている。具体的には、Matyas-Meyer-Oseas方式、MDC-2、RIPEMD-160、SHA-1等の方式が挙げられている。

こうして生成された$U(M)$に対して、秘密鍵による署名生成変換によって署名$S=U(M)^d \bmod n$が生成される。

3. ISO/IEC 9796-3の署名変換対象データ

ISO/IEC 9796-3のデジタル署名方式には、ハッシュ関数が利用されず、$C = M \bmod (2^{79}+1)$という検証式によって生成される80 bitの検証コードCが利用されている（Misarsky［1998］）。以下では、公開鍵のサイズを1024 bit、メッセージのサイズが929 bitの場合を例として、署名変換対象データのフォーマットを説明する（図12参照）。

図12 ISO/IEC 9796-3のデータフォーマット



署名変換対象データの生成は以下の３段階で行われる。

（ｉ） 中間データSF（944 bit）が生成される。SFは４つの部分から構成されており、左から、3 bitのHeader、8 bitのPadding field、929 bitのメッセージ、4 bitのTrailerとなっている。

（ii） 80 bitの検証コードCを生成する。検証コードは$C = M \bmod (2^{79}+1)$によって生成される。

（iii） 検証コードCを20個の4 bitデータに分割し、中間データSFの特定の位置に埋め込まれる。具体的には、SFの左から16 bit目に１個挿入された後、48 bitごとに１個ずつ挿入される。この結果生成される1024 bit（= 944 bit + 80 bit）のデータが署名変換対象データ$U(M)$となる。

こうして生成された$U(M)$に対して、秘密鍵による署名生成変換によって署名$S=U(M)^d \bmod n$が生成される。

補論2：CNS攻撃法の概要

＜以下の説明に利用される記号の意味＞

・U：メッセージMから署名変換対象データ$U(M)$を生成する関数。$U(M)$のサイズは公開鍵のサイズと同一となる。
・L：攻撃の際に素因数分解が必要となる数値のサイズ（bit数）。
・p_k：最小の素数２から順番に数えてk番目に大きな素数
・(n, e)：RSA署名の公開鍵
・d：RSA署名の秘密鍵

2.1　攻撃に利用されるメッセージMの選択

（1）メッセージMの条件

攻撃者に利用可能なメッセージMの条件は、その署名変換対象データ$U(M)$がp_k-*smooth*となることである。すなわち、Mは

$$U(M) = \prod_{j=1}^{k} p_j^{v_j} \qquad (1)$$

を満足する（ただし、p_jはp_k以下の素数）。

（2）攻撃に必要な素因数分解の計算量を削減するための工夫

(1) 式を満足するMを1つ見つける際に、(1) 式を満足する$U(M)$を直接素因数分解してp_k-*smooth*となっているかどうかを検証するという方法では、$U(M)$のサイズが公開鍵のサイズと同一であることから、必要な計算量は公開鍵の素因数分解と同程度となり、計算量を削減することは困難である。

このため、$U(M)$を直接素因数分解する方法の代わりに、以下の方法を採用する。

（a）公開鍵nの形態や署名変換対象データのフォーマットに着目して、$a \cdot n - b \cdot U(M)$のサイズ（L *bit*）が、公開鍵のサイズに比べて十分小さくなるようにa、bおよびMの一部を定める[27]。ただし、bの値はp_k-*smooth*となるように選択される。

27　ISO/IEC 9796-1に本攻撃法を適用する場合（公開鍵のサイズを1024 bit、メッセージのサイズを256 bitに設定）、$a = 0$かつ$b = (-1/\Gamma)$となるほか、メッセージのうちの64 bit分が一定の値に定められる。このとき、Lの値は64 bitとなる。一方、ISO/IEC 9796-2に本攻撃法を適用する場合（公開鍵、メッセージともに1024 bitに設定）、$b = 2$または2^8となるほか、aの値はnの形態に応じて決定される。メッセージについては、左から848 bit分がnの値に依存して決定される。このとき、Lの値は高々176 bitとなる。

（b） $a\cdot n-b\cdot U(M)$がp_k-smoothとなるようにMの残りの部分を定める。

上記の（a）、（b）の手順に沿ってa、b、Mを定めることにより、以下の等式が成立する。

$$a\cdot n-b\cdot U(M)=\prod_{j=1}^{k}p_j^{v_j} \qquad (1)'$$

（1）'式が成立するMによって$U(M)^d \bmod n$がp_k-smoothとなることを以下の手順で確かめることができる。

まず、$a\cdot n-b\cdot U(M)$をd乗して展開し、$\bmod n$を計算する。

$$(a\cdot n-b\cdot U(M))^d \bmod n=\{(a\cdot n)^d+d(a\cdot n)^{d-1}(-b\cdot U(M))+\cdots+(-b\cdot U(M))^d\} \bmod n$$

右辺括弧内の項のうち、nがかかっている項の$\bmod n$を計算すると0となることから、

$$(a\cdot n-b\cdot U(M))^d \bmod n=(-b\cdot U(M))^d \bmod n=(-b)^d U(M)^d \bmod n$$

上記の式を$U(M)^d \bmod n$について整理すると、

$$U(M)^d \bmod n=(-b)^{-d}(a\cdot n-b\cdot U(M))^d \bmod n$$

これに（1）'式を代入すると、

$$U(M)^d \bmod n=(-b)^{-d}\left(\prod_{j=1}^{k}p_j^{v_j}\right)^d \bmod n$$

となる。bの値がp_k-smoothとなることから、$U(M)^d \bmod n$は必ずp_k-smoothとなる。

（3）メッセージMの選択に必要な計算量

攻撃に利用するメッセージを１つ見つけるために必要な計算量は、攻撃者が定めたp_kに対して、$a\cdot n-b\cdot U(M)$の取り得る値の中にp_k-smoothな値が存在する確率Xと、$a\cdot n-b\cdot U(M)$がp_k-smoothであるか否かを検証するために必要な計算量Yを求め、$X\cdot Y$を計算する必要がある。

（a） $a\cdot n-b\cdot U(M)$の取り得る値の中にp_k-smoothな値が存在する確率

$a\cdot n-b\cdot U(M)$の取り得る値の数は、$a\cdot n-b\cdot U(M)$のサイズがL bitであることから2^Lとなる。L bitのデータの中でp_k-smoothとなる合成数の割合は、p_kのサイズ（$\log_2(p_k)$ bit）とLの値に依存しており、以下の関数$\rho(t)$によって与えられる。ただし、$t=L/\log_2(p_k)$である。

$$\rho(t) \cong (2\pi t)^{-1/2} exp\left(\gamma - t\zeta + \int_0^{\zeta} \frac{e^s - 1}{s} ds\right) \quad (e^{\zeta} - 1 = t\zeta、\gamma：正定数)$$

このように、(1)'式を満足するMを1つ見つけるためには、平均$(1/\rho(L/log_2(p_k)))$個の数をランダムに選択する必要がある。

— 例えば、Lが160 bit、p_kのサイズが24 bitの場合（t=160/24の場合）には、$\rho(t) \approx 2^{-19}$となる。これは、2^{24}-*smooth*となる160 bitの数を見つけるためには平均2^{19}個の自然数を選択する必要があることを意味する。

(b) $a \cdot n - b \cdot U(M)$がp_k-*smooth*であるか否かを検証するための計算量

選択したMが(1)'式を満足するか否かを検証するための計算量のオーダーは、$a \cdot n - b \cdot U(M)$のサイズがL *bit*であることから$O(L\sqrt{k \ln k})$となる[28]。

以上の(a)と(b)より、(1)'式を満足するMを1つ見つけるために必要となる平均的な計算量$C_{L,k}$は、

$$C_{L,k} = O\left(\frac{L\sqrt{k \ln k}}{\rho(L/log_2(k \ln k))}\right)$$

となる。攻撃者は、$C_{L,k}$が最小となるようにp_kの値を設定する。

2.2 署名の偽造方法

(1) 選択したMに対する署名入手

攻撃者は、(1)'式を満足するr個のメッセージ（$M_1, M_2, ..., M_{r-1}, M_r$）を入手し、正当な署名者に送信する。署名者は、自分の秘密鍵を利用して(2)式で表される署名$U(M_i)^d \bmod n$を生成し、攻撃者に返信する。

$$U(M_i)^d = \prod_{j=1}^{k} p_j^{(v_{i,j})d} (\bmod n) \qquad (2)$$

署名偽造の対象となるメッセージをM_{r+1}とすると、M_{r+1}も(1)式を満足する必要がある。

(2) 署名の偽造方法

攻撃者は、入手した署名$U(M_i)^d \bmod n$からM_{r+1}に対する署名を偽造するために、$U(M_i)^d \bmod n$を説明変数、$U(M_{r+1})^d \bmod n$を従属変数とする方程式を以下の手順で導出する。

28 ここで利用されている素因数分解アルゴリズムはPollard-Brentのアルゴリズムである（Coron et al.［1999］）。Coronらは、Pollard-Brentのアルゴリズムが比較的小さな素数からなる合成数の素因数分解を行う際に有効なアルゴリズムであるとして、CNS攻撃法に利用している。

〈1〉 $U(M_i)$ を、以下のように表されるk次元ベクトルV_iに対応させる。ただし、eはRSA署名方式の公開鍵の一部である。

$$U(M_i) \rightarrow V_i = \{ v_{i,1} \bmod e, v_{i,2} \bmod e, \ldots, v_{i,k} \bmod e \}$$

V_iを要素とするベクトルの集合の要素はe^k個存在し、基底となるベクトルは$\{p_1, p_2, p_3, \ldots, p_k\}$である。

〈2〉 偽造対象メッセージM_{r+1}にも、$U(M_{r+1})$に対応するk次元ベクトルV_{r+1}が存在し、V_{r+1}は他のr本のベクトルV_iの線形結合で表される。すなわち、V_{r+1}は適当な定数b_i（$i=1,\ldots,r$、$0 \leqq b_i \leqq e-1$）によって、

$$V_{r+1} = \sum_{i=1}^{r} b_i V_i \bmod e = \{ v_{r+1,1} \bmod e, v_{r+1,2} \bmod e, \ldots, v_{r+1,k} \bmod e \} \quad (3)$$

と表される。

(3)式に$V_i = \{v_{i,1} \bmod e, v_{i,2} \bmod e, \ldots, v_{i,k} \bmod e\}$を代入すると、

$$\sum_{i=1}^{r} b_i V_i \bmod e = \sum_{i=1}^{r} b_i \{ v_{i,1} \bmod e, v_{i,2} \bmod e, \ldots, v_{i,k} \bmod e \} \bmod e$$

$$= \left\{ \sum_{i=1}^{r} b_i v_{i,1} \bmod e, \sum_{i=1}^{r} b_i v_{i,2} \bmod e, \ldots, \sum_{i=1}^{r} b_i v_{i,k} \bmod e \right\} \quad (4)$$

となる。

〈3〉 (3)式と(4)式の右辺を比較すると、$v_{r+1,j}$について以下の等式が得られる。

$$v_{r+1,j} = \sum_{i=1}^{r} b_i v_{i,j} \bmod e$$

さらに、適当なa_jを選び、$v_{r+1,j}$を(5)式のように表す。

$$v_{r+1,j} = \sum_{i=1}^{r} b_i v_{i,j} - a_j e \quad (5)$$

〈4〉 偽造対象メッセージM_{r+1}に対する署名$U(M_{r+1})^d \bmod n$は、以下のように表される。

$$U(M_{r+1})^d = \prod_{j=1}^{k} p_j^{(v_{r+1,j})^d} \bmod n \quad (6)$$

(6)式に(5)式を代入し、以下のように変形する。

$$U(M_{r+1})^d = \prod_{j=1}^{k} p_j \left(\sum_{i=1}^{r} b_i v_{i,j} \; a - j e \right)^d \bmod n$$

$$= \prod_{j=1}^{k} p_j \left(\sum_{i=1}^{r} b_i v_{i,j} \right)^d \times \prod_{j=1}^{k} p_j^{(-a_j ed)} \bmod n \quad (7)$$

RSA署名方式の演算の性質 $e \times d \bmod (p-1)(q-1) = 1$を、(7)式の右辺第２項に適用すると、

$$U(M_{r+1})^d = \prod_{j=1}^{k} p_j^{\left(\sum_{i=1}^{r} b_i v_{i,j}\right)d} \times \prod_{j=1}^{k} p_j^{(-a_j)} \bmod n$$

さらに、式を展開すると、

$$U(M_{r+1})^d = \prod_{j=1}^{k} p_j^{(b_1 v_{1,j} + b_2 v_{2,j} + \cdots + b_r v_{r,j})d} \times \prod_{j=1}^{k} p_j^{-a_j} \bmod n$$

$$U(M_{r+1})^d = \prod_{j=1}^{k} \prod_{i=1}^{r} p_j^{(v_{i,j})d \times b_i} \times \prod_{j=1}^{k} p_j^{-a_j} \bmod n$$

$$= \prod_{i=1}^{r} \left(\prod_{j=1}^{k} p_j^{(v_{i,j})d} \right)^{b_i} \times \prod_{j=1}^{k} p_j^{-a_j} \bmod n \qquad (8)$$

(8)式に(2)式を代入すると、

$$U(M_{r+1})^d = \prod_{i=1}^{r} (U(M_i)^d)^{b_i} \times \prod_{j=1}^{k} p_j^{-a_j} \bmod n \qquad (9)$$

(9)式が、M_{r+1}の署名を偽造するための方程式である。

〈5〉(9)式に入手した署名 $U(M_i)^d \bmod n$を代入することによって、$U(M_{r+1})^d \bmod n$を生成することができる。

計算に必要となるM_iの個数はkとeの値に依存し、$O(k \log e)$と表される。必要となる計算量は$C_{L,k}$に$O(k \log e)$ を乗じた値となり、

$$O\left(\frac{Lk(\log e)\sqrt{k \ln k}}{\rho(L/\log_2(k \ln k))} \right)$$

と表される。

参考文献

岩下直行・谷田部充子、「金融分野における情報セキュリティ技術の国際標準化動向」、『金融研究』第18巻第２号、日本銀行金融研究所、1999年、33-56頁

宇根正志、「公開鍵暗号方式EPOCについて」、*IMES Discussion Paper* 98-J-19、日本銀行金融研究所、1998年

――――・岡本龍明、「安全性証明付きの公開鍵暗号方式を巡る動向について」、『1999年暗号と情報セキュリティシンポジウム予稿集』、電子情報通信学会、1999a年、881-886頁

――――・――――、「公開鍵暗号の理論研究における最近の動向」、『金融研究』第18巻第２号、日本銀行金融研究所、1999b年、195-251頁

田中初一、「ID情報に基づく岡本鍵配送方式の解析」、『電子情報通信学会技術報告』ISEC88-7、電子情報通信学会、1988年5月

American National Standards Institute, “X 9.31 Digital Signatures Using Reversible Public Key Cryptography (rDSA),” 1998.

Bellare, M., and P. Rogaway, “Optimal asymmetric encryption,” *Advances in Cryptology — Proceedings of EUROCRYPT ’94*, Lecture Notes in Computer Science, Vol. 950, Springer-Verlag, 1995, pp.92-111.

――――, and ――― , “The Exact Security of Digital Signatures—How to Sign with RSA and Rabin,” *Advances in Cryptology — Proceedings of EUROCRYPT ’96*, Lecture Notes in Computer Science, Vol. 1070, Springer-Verlag, 1996, pp.399-416.

Bleichenbacher, D., “Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1,” *Advances in Cryptology—Proceedings of CRYPTO’98*, Lecture Notes in Computer Science, Vol. 1462, Springer-Verlag, 1998, pp.1-12.

Coron, J. S., D. Naccache, and P. Stern, “On the Security of RSA Padding,” to appear CRYPTO ’99, 1999.

Desmedt, Y. G., and A. M. Odlyzko, “A chosen text attack on the RSA cyptosystem and some discrete logarithm schemes,” *Advances in Cryptology—Proceedings of CRYPTO’85*, Lecture Notes in Computer Science, Vol. 218, Springer-Verlag, 1986, pp.516-522.

Europay International S.A., MasterCard International Incorporated, and Visa International Service Association, “EMV’96: Integrated Circuit Card Specification for Payment Systems,” July 1996 (http://www.visa.com/nt/chip /download.html) .

Girault, M., and J. F. Misarsky, “Selective Forgery of RSA Signatures Using Redundancy,” *Advances in Cryptology— Proceedings of EUROCRYPT ’97*, Lecture Notes in Computer Science, Vol. 1233, Springer-Verlag, 1997, pp. 495-507.

Guillou, L. C., J.-J. Quisquater, M. Walker, P. Landrock, and C. Shaer, “Precautions taken against various potential attack in ISO/IEC DIS 9796 “Digital signature scheme giving message recovery,” *Advances in Cryptology— Proceedings of EUROCRYPT ’90*, Lecture Notes in Computer Science, Vol. 473, Springer-Verlag, 1991, pp. 465-473.

International Organization for Standardization, “ISO 11166-2 Banking — Key management by means of asymmetric algorithms— Part 2: Approved algorithms using the RSA cryptosystem,” 1994.

————, “ISO/TR 13569 Banking and related financial services —Information security guidelines,” October 1997.

————, “ISO/TR 13569 Banking and related financial services—Information security guidelines, Amendment 1,” December 1998.

————, and International Electrotechnical Commission, “ISO/IEC 9796 Information technology—Security techniques—Digital signature scheme giving message,” 1991.

————, and ————, “ISO/IEC 10118-2 Information technology—Information techniques—Hash-functions—Part 2: Hash-functions using an n-bit block cipher algorithm,” 1994.

————, and ————, “ISO/IEC 9796-2 Information technology—Security techniques—Digital signature scheme giving message recovery—Part 2: Mechanisms using a hash-function,” 1997.

————, and ————, “ISO/IEC 10118-3 Information technology—Information techniques—Hash-functions—Part 3: Dedicated hash-functions,” 1998a.

————, and ————, “ISO/IEC 14888-3 Information technology —Security techniques—Digital signature with appendix—Part 3: Certificate-based mechanism,” 1998b.

Kaliski, B., and M. Robshaw, “The Secure Use of RSA,” *RSA Laboratories CryptoBytes*, Vol. 1, No. 3, 1995.

Menezes, A. J., P. C. Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.

Misarsky, J. F., “A Multiplicative Attack Using LLL Algorithm on RSA Signatures with Redundancy,” *Advances in Cryptology —Proceedings of CRYPTO’97*, Lecture Notes in Computer Science, Vol. 1294, Springer-Verlag, 1997, pp. 221-234.

————, “How (not) to design RSA signature schemes,” *Proceedings of Public Key Cryptography ’98*, Lecture Notes in Computer Science, Vol. 1431, Springer-Verlag, 1998, pp.14-28.

National Institute for Standards and Technology, “Specifications for a digital signature standard,” *Federal Information Processing Standard Publication* (*FIPS PUB*) 186-1, 1999.

Okamoto, T., E. Fujisaki, and H. Morita, “TSH-ESIGN: Efficient Digital Signature Scheme Using Trisection Size Hash,” Submission to IEEE P1363a, November 1998.

————, S. Uchiyama, and E. Fujisaki, “EPOC: Efficient Probabilistic Public-Key Encryption,” Submission to IEEE P1363a, November 1998.

Rivest, R. L., A. Shamir, and L. Adleman, “A method of obtaining digital signatures and public key cryptosystems,” *Communications of the ACM*, Vol. 21, No. 2, 1978, pp.120-126.

RSA Laboratories, “PKCS #1: RSA Encryption Standard Version 1.5,” November 1993.

————, “PKCS #1: RSA Cryptography Specifications Version 2.0,” September 1998 (ftp://ftp.rsa.com/pub/pkcs/ascii/pkcs-1v2.asc).

Silverman, R. D., and D. Naccache, “Recent Results on Signature Forgery,” April 1999 (http://www.rsa.com/rsalabs/html/sigforge.html).