RSA-PSS 評価報告書

平成 13 年 1 月 12 日

概要

1996 年 5 月に開催された Eurocrypt '96 において Bellare と Rogaway は確率的署名方式 (Probabilistic Signature Scheme, PSS) と呼ばれる署名方式を提案した [1]。PSS は RSA 署名を施すメッセージに乱数を付加することにより、確率的な挙動をするように RSA 署名に変更が加えられた方式である。1996 年までに提案されていた既存の全域ハッシュ方式と比べて、RSA 署名の安全性とのより緊密な関係を証明可能である。本稿はこの署名方式の概要と安全性について報告した後、日本の電子政府への採用可能性についての提言を行う。

Abstract: Bellare and Rogaway proposed a new signature scheme called probabilistic signature scheme, PSS, at Eurocrypt '96 in May, 1996. PSS is a modified version of the RSA signature scheme where a random seed is used for creating a signature and its operation is probabilistic. In comparison with the full-domain-hash scheme, FDH, proposed in 1993, the security of PSS is much more tight with the RSA signature scheme than the security of FDH is. We report the construction and security of PSS, and give our opinion for the possibility of adopting PSS in electronic goverment of Japan.

1 プリミティブの概要と安全性

1.1 プリミティブ

確率的署名方式 (Probabilistic Signature Scheme, PSS) には、下記のプリミティブが用いられている。

1.1.1 RSA 生成器：$\mathcal{RSA}$

入力 1^k に対して、ランダムに二つの $(k/2)$-ビットの素数を選び、その値を乗じて N とする。また、暗号化する際のベキ $e \in \mathbf{Z}^*_{\varphi(N)}$ をランダムに選び、それに対応する $ed \equiv 1$

$\mathrm{mod}\ \varphi(N)$ を満たす復号化する際のべき d を計算する。φ は Euler 関数を表す。

つまり RSA 生成器は、暗号化関数 f を決定する N, e, d を出力する。関数 $f : \mathbf{Z}_N^* \to \mathbf{Z}_N^*$ と $f^{-1} : \mathbf{Z}_N^* \to \mathbf{Z}_N^*$ は、それぞれ、$f(x) = x^e \bmod N$, $f(x) = x^d \bmod N$ と定義され、これら二つの関数はどちらも $\mathbf{Z}_N^*$ の置換 (全単射) となる。

1.1.2 ハッシュ関数：MD5, SHA-1

PSS では署名生成及び署名検証において、ハッシュ関数 h や乱数生成器 g を利用している。H =MD5 や H =SHA-1 のような暗号理論上のハッシュ関数 H から h を構成する方法は、例えば、

$$H(\mathtt{const}.\langle 0\rangle.x)||H(\mathtt{const}.\langle 1\rangle.x)||H(\mathtt{const}.\langle 2\rangle.x)||\cdots.$$

という文字列から h の出力長分の語頭を切り出し、出力とすることである。ここで、$\langle i\rangle$ は整数 i の 32 ビットのビット列への符号化を表す。$||$ はビット列の連結を表す。定数 const は、関数 h について固有の値であり、異なるハッシュ関数を生成する際には、別の定数を用いる。

乱数生成器 g の場合も、同様の処理により構成可能である。

1.2 安全性及びパラメータ

RSA 暗号の暗号化関数 f に対する逆像を求めるアルゴリズム I は、(N, e, y) を入力として、$f^{-1}(y)$ を求めるアルゴリズムとして定義される。この I の成功確率は、(N, e, d) を $\mathcal{RSA}(1^k)$ により生成し、$\mathbf{Z}_N^*$ からランダムに選ばれた x について $y = f(x)$ としたときに、$f^{-1}(y)$ を出力する確率である。以下、このアルゴリズムを形式的に定義する。

ある計算モデルにおいて、実行時間と表現サイズの和が $t(k)$ によって抑えられるとき、I を *t-inverter* と言う。ここで、$t : N \to N$。また、I が t-inverter でかつ、各 k に対して I の成功確率が少なくとも $\epsilon(k)$ であるとき、I は $\mathcal{RSA}$ を (t, ϵ)-偽造すると言う。ここで、$\epsilon : N \to [0, 1]$。さらに、$\mathcal{RSA}$ を (t, ϵ)-偽造するような inverter が存在しないとき、$\mathcal{RSA}$ は (t, ϵ)-安全であるという。素因数分解問題を解くアルゴリズムは様々あり、現在最も高速だと考えられているものに数体ふるい法 (Number Field Sieve, NFS) がある。NFS では、k-ビットの合成数を分解するのに、約 $e^{1.9k^{1/3}(\log k)^{2/3}}$ 時間かかる [2]。つまり、$\mathcal{RSA}$ は、C をある特別な定数として $t(k)/\epsilon(k) \leq Ce^{k^{1/4}}$ を満足するすべての t, ϵ について、(t, ϵ)-安全であると仮定してよい。文献 [3, 4] には、平成 13 年 1 月現在までに報告されているデータをもとに、素因数分解された整数の長さの年ごとの変遷が示されている。特に、文献 [3] において、平成 11 年 8 月 22 日に 512 ビットの素因数分解に成功したこと、ならびに、この素因数分解に 8400MIPS 年の計算量が必要となったことが報告されている。

2 スキームの概要と安全性

本報告では、RSA 署名を利用した各種の方式の中で、$x = f^{-1}(y)$ の計算自体は変更せず、入力 y の構造を変える方式を主に取り上げる。

2.1 既存の方式

PSS よりも以前に提案された方式として、ハッシュ後復号化方式 (Hash-Then-Decrypt 方式) と全域ハッシュ方式 (Full-Domain-Hash 方式) がある。いずれも、y をハッシュ関数の出力自体としたものであり、メッセージ M からハッシュ関数を計算して y を求める処理には、確率的な振舞いはない。これらは PSS が考案された経緯と密接に関係しているため、概要と安全性をまず概観する。

2.1.1 ハッシュ後復号化方式

RSA 署名において、メッセージ M に対して、まず、あるハッシュ関数により $y = Hash(M)$ と計算したのち、$x = f^{-1}(y) = y^d \bmod N$ により署名を生成するのが、ハッシュ後復号化方式 (Hash-Then-Decrypt 方式) である。検証においては、$f(x) = x^e \bmod N$ を計算し、$Hash(M)$ との一致を確認する。この方式は衝突困難なハッシュ関数を用いており、具体的には H を MD5 のような暗号理論上のハッシュ関数を使って以下のような $Hash_{\mathrm{PKCS}}$ を用いる。

$$Hash_{\mathrm{PKCS}}(M) = \texttt{0x 00 01 FF FF}\cdots\texttt{FF FF 00}||H(M).$$

このハッシュ関数は $\mathbf{Z}_N^*$ の一部の要素のみを出力する関数であるため、攻撃の対象となる可能性が残る。実際、文献 [5] において、一つの偽造方法が示されている。但し、この偽造方法はある特定の法の場合にのみ適用可能で、しかも、素因数分解を実行するよりも効率が悪いため、実質的な脅威ではない。

2.1.2 全域ハッシュ方式

メッセージ M をハッシュする際に、$\mathbf{Z}_N^*$ の定義域全体にハッシュするように改良したのが全域ハッシュ (Full-Domain-Hash 方式, FDH) である。つまり、メッセージをハッシュする関数 $Hash_{FDH}$ は、$Hash_{FDH} : \{0,1\}^* \to \mathbf{Z}_N^*$ となる。この方式の安全性は、$Hash$ が理想的だと仮定すると RSA にのみ依存する。

ここで、RSA と署名方式について、以下のような安全性の概念を定義する。攻撃者が、$\mathbf{Z}_N^*$ からランダムに y を選び $t'(k)$ 時間内に高々 $\epsilon'(k)$ の確率で $f^{-1}(y)$ を求めることに成功するとき、RSA は $(t', \epsilon')-$ 安全 であるという。また、公開鍵を知っている攻撃者が、q_{sig} 個の署名とメッセージの組を与えられ、q_{hash} 回ハッシュ関数にアクセス可能で、

さらに $t'(k)$ 時間内に高々 $\epsilon(k)$ の確率で新しい署名の偽造に成功するとき、その署名方式は $(t, q_{\mathrm{sig}}, q_{\mathrm{hash}}, \epsilon)$-安全 であるという。

FDH の安全性は証明されており、RSA が (t', ϵ')-安全ならば FDH$[k_0, k_1]$ は $(t, q_{\mathrm{sig}}, q_{\mathrm{hash}}, \epsilon)$-安全 となる。ここで、$t = t' - \mathrm{poly}(q_{\mathrm{sig}}, q_{\mathrm{hash}}, k)$, $\epsilon = (q_{\mathrm{sig}} + q_{\mathrm{hash}})\dot{\epsilon}'$ である。

2.2 提案方式

既存の方式と異なり、ハッシュ関数の入力としてメッセージ M に加えて乱数を含めることにより、y を求める処理に確率的な振舞いを持たせる。

前節の FDH は RSA 署名との帰着関係を証明可能であったが、RSA 署名の偽造に成功する確率に比べて、FDH の偽造に成功する確率が非常に大きくなる可能性が残っていた。つまり、RSA 自体の安全性が FDH の安全性に反映されにくい部分があったため、FDH よりもさらに緊密な帰着関係を有する署名方式として PSS が提案された。

2.2.1 確率的署名方式

確率的署名方式 (Probabilistic Signature Scheme, PSS) $PSS[k_0, k_1] = (genPSS, SignPSS, VerifyPSS)$ は、1 と k の間の値で $k_0 + k_1 \leq k - 1$ を満足する k_0, k_1 をパラメータとして取る。具体的なパラメータの値として、例えば、$k = 1024, k_0 = k_1 = 128$ が挙げられている。

鍵生成アルゴリズム $GenPSS$ は、1^k を入力として、$\mathcal{RSA}(1^k)$ により (N, e, d) を得る。つまり、$pk = (N, e), sk = (N, d)$ であるような (pk, sk) を出力する。

署名及び検証アルゴリズムは、二つのハッシュ関数を利用している。一つは、圧縮器と呼ばれる写像 h であり、$h : \{0,1\}^* \to \{0,1\}^{k_1}$ と定義される。二つ目は、乱数生成器と呼ばれる写像 g であり、$g : \{0,1\}^{k_1} \to \{0,1\}^{k-k_1-1}$ と定義される。ここで、g_1 を $w \in \{0,1\}^{k_1}$ を入力として $g(w)$ の先頭 k_0 ビットを出力する関数、g_2 を $w \in \{0,1\}^{k_1}$ を入力として $g(w)$ の残りの $k - k_0 - k_1 - 1$ ビットを出力する関数とする。署名及び検証アルゴリズムは、それぞれ以下のようになる。

- 署名アルゴリズム

  $SignPSS(M)$

  $r \overset{R}{\leftarrow} \{0,1\}^{k_0}$; $w \leftarrow h(M \parallel r)$; $r^* \leftarrow g_1(w) \oplus r$

  $y \leftarrow 0 \parallel w \parallel r^* \parallel g_2(w)$

  return $y^d \bmod N$

  ステップ $r \overset{R}{\leftarrow} \{0,1\}^{k_0}$ は、署名者がランダムに k_0 ビットの 種 r を選ぶことを示している。$\parallel$ は連結である。次に、署名者はこの種とメッセージ M を結合し、圧縮関数 h で k_1 ビットの文字列 w を生成する。一方、乱数生成用関数は、w から k_0 ビットの文字列 r^* と $k - k_0 - k_1 - 1$ ビットの文字列を生成する。最後に、$w \parallel r^*$ の上位に 0 を下位に $g_2(w)$ を結合する。0 を結合することにより、y が必ず RSA 関数の法の値よりも小さくなるように設定している。

なお、各メッセージ毎に新しい種を選ぶ。このため、同一のメッセージであっても、署名者が選ぶ r の値によって、異なる署名が生成される。

- 検証アルゴリズム

$VerifyPSS(M, x)$

$y \leftarrow x^e \bmod N$

Break up y as $b \parallel w \parallel r^* \parallel \gamma$.

$r \leftarrow r^* \oplus g_1(w)$

if $(h(M \parallel r) = w$ and $g_2(w) = \gamma$ and $b = 0)$ **then return 1**

else return 0

署名 (M, x) が与えられたとき、検証者はまず $y = x^e \bmod N$ を計算し、r^*, w, r を復元する。これらの値は、y が正しく構成されたことの検証に利用され、全てのチェックが成功したときのみ、正当な署名として受理する。

2.2.2 安全性

PSS の安全性は、FDH 方式と同様に RSA 暗号の安全性に基づいており以下の定理が示されている。

定理：$\mathcal{RSA}$ が (t', ϵ') 安全だと仮定する。このとき、任意の $q_{\mathrm{sig}}, q_{\mathrm{hash}}$ に対して署名方式 $\mathbf{PSS}[k_0, k_1]$ は $(t, q_{\mathrm{sig}}, q_{\mathrm{hash}}, \epsilon)$-安全である。ここで、

$$t(k) = t'(k) - [q_{\mathrm{sig}}(k) + q_{\mathrm{hash}}(k) + 1] \cdot k_0 \cdot \Theta(k^3)$$

$$\epsilon(k) = \epsilon'(k) - [2(q_{\mathrm{sig}}(k) + q_{\mathrm{hash}}(k))^2 + 1] \cdot (2^{-k_0} + 2^{-k_1})$$

が成り立つ。

ここで、ϵ に関する式は、k_0, k_1 の大きさに対して指数関数的に小さくなる $o(1)$ を用いて、$\epsilon = \epsilon' - o(1)$ と表わせる。よって、PSS における偽造の成功確率は、k_0, k_1 が大きくなるについて、RSA 署名における偽造の成功確率に指数関数的に近付くため、PSS の安全性は RSA の安全性と非常に密接に関係しており、PSS は FDH よりよりも高い安全性を有している。

2.2.3 提案方式の拡張

PSS の変形版として PSS-R が提案されている。これは、PSS にメッセージ回復機能が付加された方式であり、その安全性も PSS とほぼ同等である。

さらに、RSA から PSS を構成したのと同様の手法で、Rabin 署名を確率的な振舞いをするように改良した方式も提案されている。これらは、メッセージ回復機能がない Rab、及び、メッセージ回復機能がある Rab-R に分けられる。Rab 及び Rab-R は、RSA 署名を偽造する問題を仮定せずに、素因数分解問題の仮定のみで構成可能であるため、素因数分解問題が RSA 署名を偽造する問題に帰着することが知られていない現在、PSS よりも、高い安全性を有するといえる。

3 提言

本報告では、RSA-PSSの概要とその安全性について検討した。RSA-PSSに対する攻撃についての論文を探索したが、現在までに知り得る限りにおいて、攻撃に成功した論文は存在しなかった。現状では素因数分解かRSA署名の偽造以外は攻撃の方法が存在しないと考えられる。このように、RSA-PSSはRSA自体の安全性、並びに、素因数分解問題の困難さに大きく依存するため、これらの問題を解くアルゴリズム、及び、計算機能力の進歩には、今後も十分に注意を払っていく必要がある。

最後に、本方式の電子政府への採用については、少くともセキュリティ面では、RSAが電子政府で採用されるかどうかとほぼ同等の問題であると言える。しかし、米国政府が自国の方式(IEEE標準案)としてPSSを採用しないならば、それはRSAないし、素因数分解問題を信用していない証拠と考えられるため、日本政府も注意が必要である。そもそも、米国の方式を日本政府が使うのは、PSSの安全性に何らかの重大な問題が発生して、行政的な責任問題が発生したときに、国内で責任関係が完結しないので、解決が困難になる可能性がある。

参考文献

[1] M. Bellare and P. Rogaway, "The Exact Security of Digital Signatures - How to Sign with RSA and Rabin," U. Maurer (Ed.), Advances in Cryptology - EUROCRYPT '96, Lecture Notes in Computer Science vol.1070, Springer-Verlag, pp.399-416, 1996.

[2] J.P.Buhler, H.W. Lenstra, Jr. and C. Pomerance, "Factoring integers with the number field sieve," A. Lenstra and H. Lenstra (eds.), The development of the number field sieve, Lecture Notes in Mathematics, vol.1554, pp.50-94, Springer-Verlag, 1993.

[3] S. Cavallar, B, Dodson, A.K. Lenstra, W. Lioen, P.L. Montgomery, B. Murphy, H. t. Riele, K. Aardal, J. Gilchrist, G. Guillerm, P. Leyland, J. Marchand, F. Morain, A. Muffett, C. Putnam, C. Putman and P. Zimmermann, "Factorization of a 512-Bit RSA Modulus," B. Preneel (Ed.), Advances in Cryptology - EUROCRYPT 2000, Lecture Notes in Computer Science vol.1807, Springer-Verlag, pp.1-18, 2000.

[4] H. IMAI and J. SHIKATA, "Development of Cryptology in the Nineties," IEICE Trans. on Fundamentals, VOL.E84-A, NO.1, PP.61-67, 2001.

[5] J.-S. Coron, D. Naccache and J.P. Stern, "On the Security of RSA Padding," M. Wiener (Ed.), Advances in Cryptology - CRYPTO 2000, Lecture Notes in Computer Science vol.1666, Springer-Verlag, pp.1-18, 2000.