ESIGN 署名方式詳細評価報告書

2001 年 12 月 14 日

ESIGN 署名方式詳細評価報告書

2001.12.14

概要

本報告書は、CRYPTREC からの依頼に基づき、ESIGN 署名の安全性評価を行ったものである。本報告書の構成は次の通りである。第 2 章で ESIGN 署名提案者自身による自己評価の概要をまとめる。第 3 章で e 乗根近似問題について、第 4 章で素因数分解問題について考察する。第 5 章で ESIGN 署名の補助関数であるハッシュ関数について述べる。第 6 章で数多くある ESIGN 署名のバージョンの違いについて整理する。最後に、第 7 章において ESIGN 署名の安全性評価を総括する。

1 はじめに

本報告書は、CRYPTRECからの依頼に基づき、ESIGN署名の安全性評価を行ったものである。CRYPTRECからの評価依頼書には、次の5つの安全性評価項目がリストされている。

- e 乗根近似問題
- 電子署名および認証業務に関する法律に基づく特定認証業務の認定に係る指針案における推奨パラメータ: $e \geq 8$、$|n| \geq 1024$ の安全性評価
- 昨年度CRYPTRECへの応募における応募者による推奨パラメータ: $e \geq 8$、$|p| = |q| \geq 320$、$|n| \geq 960$ の安全性評価
- 昨年度CRYPTRECにおけるソフトウエア実装評価時の応募者による採用パラメータ: $e = 2^{10}$、$|p| = |q| = 384$、$|n| \geq 1152$ の安全性評価
- Vallèe らの攻撃 [VGT88a] の拡張可能性について

本稿の以下の章では、これらの評価項目を以下の3つのカテゴリーに分類して評価を行う。

- e 乗根近似仮定に関して
- 素因数分解仮定に関して
- その他に関して

2 自己評価書の概要

ESIGN署名提案者自身による自己評価 [NTT01b] の概要をまとめる。

2.1 設計方針

提案者により示されている設計方針の大きな柱は2つあり、それらは次の通りである。

1. 適切な仮定のもとで、安全性に関して高い信頼性を保持すること
2. 代表的署名方式であるRSA署名よりも、実装性能において優れていること

2.1.1 安全性に関して

ESIGN署名には、いくつかのバージョンが存在し、IEEE P1361a[IEEE] に採録されたバージョンでは、ディジタル署名における最強の意味での安全性、すなわち、“ある仮定のもとで適応的選択文書攻撃に対して存在的偽造不可” であることの証明がついている。しかし、今年度CRYPTRECに投稿したバージョンは、IEEEバージョンとは異なる仕様で、現時点では前述のような安全性の証明はついていない。

安全性の証明がついていないことに関する提案者の主張は次の通りである。

- 今年度 CRYPTREC 投稿バージョンは 15 年以上の使用実績があり、現時点では最強の意味での安全性に関する攻撃は発見されておらず、heuristic な設計での安全性の証明はないが、現在知られている攻撃法の観点では安全である

2.1.2 速度性能に関して

提案者自身の実装では、Celeron 800MHz において、ESIGN 署名の法 n が 1152 ビット、安全性パラメータ e が $e = 2^{10}$ の時、鍵生成生成 610ms、署名生成 1.04ms、署名検証 0.70ms とのことである。RSA 署名や楕円曲線 DSA 署名と比べて、速度的に優秀と提案者は主張している。

3 e 乗根近似問題

定義 1 (AER 問題) $\mathcal{G}$ を *ESIGN* の鍵生成とする。e 乗根近似問題 *(AER* 問題*)* とは、$pk := \{n, e\} \leftarrow Gen(1^k)$ と $y \leftarrow_R \{0,1\}^{k-1}$ が与えられたとき、$0||y = [x^e \ mod \ n]^k$ となるような $x \in (\mathbf{Z}/n\mathbf{Z}) \setminus p\mathbf{Z}$ を見つける問題である。

定義 2 (AER 仮定) どのような確率的多項式時間アルゴリズム Adv に対しても、全ての定数 c、十分大きな値 k に対して、

$$Pr[Adv(k, n, e, y) \rightarrow x] < 1/k^c$$

が成立するとき、e 乗根近似問題は難しいという。ここで、$0||y = [x^e \ mod \ n]^k$ であり、確率は $\mathcal{G}$ と Adv の確率空間上で取られる。e 乗根近似問題が難しいという仮定は、e 乗根近似仮定 *(AER* 仮定*)* と呼ばれる。

AER 問題は、$e = 2$ と $e = 3$ の場合に lattice basis reduction 法によって解かれている [BD85]。$e \geq 4$ の場合は、現時点では効率的解法は知られておらず、知られている唯一の解法においては、法 n を素因数分解することが必要となる。

3.1 $e > 2^{30}$ について

CRYPTREC からの評価依頼項目の 1 つである、

"ESIGN 署名が安全であるためには $e > 2^{30}$ であることが必要か?"

に関して考察する。これは、2000 年度評価報告書 (の 1 つ)[Cr00] の中で記述されている事項に関連している。

3.1.1 2000 年度報告書の概要

2000 年度詳細評価書 [Cr00] の中で、Vallèe らの攻撃の拡張可能性に関する指摘がなされている。その指摘の概要は次の通りである。

1. Brickel の方法は $e \geq 4$ に適用できないことが分かっているが、Vallèe らの方法を $e \geq 4$ に適用できるかどうかは不明であり、Vallèe らの方法を拡張できる可能性が残されている。
2. Vallèe らの方法の理論的限界はわかっていないが、Vallèe らの方法は LLL アルゴリズムを使うため、LLL アルゴリズムの限界を考えれば Vallèe らの方法の適用限界を推定できる。
3. Vallèe らの方法の時間計算量は $O(e^3)$ である。
4. Vallèe らの方法の空間計算量は $O(e^2)$ である。
5. Vallèe らの方法を並列実行できた場合、時間計算量は $O(e^2)$ になる可能性がある。
6. 時間計算量 $O(2^{128})$ 程度の安全性を保証するためには、$e > 2^{64}$ 程度にする必要がある。
7. $O(2^{60})$ 程度の記憶域を用意することは物理的に困難であると仮定すると、$e > 2^{30}$ 程度で安全となる。
8. Vallèe らの方法を拡張できるかどうかは知られていないが、もし拡張できたとしても、$e > 2^{30}$ 程度であれば、e 乗根近似問題は解けないことになる。

3.1.2 2000 年度報告書に関する考察

前節の "$e > 2^{30}$" に関して考察する。

まず初めに注意しなければならないことは、$e > 2^{30}$ という数値の算出の根拠には多くの仮定が置かれていることである。LLL アルゴリズムの時間計算量として漸近評価式を用い、また、空間計算量としても漸近評価式を用いている。これらの漸近評価式から、$e > 2^{30}$ という specific な数値がはじき出されている。つまり、オーダの比例係数は、実際に計算機実験をすれば数倍変わることもあり、それに伴って 2^{30} という数値も変化する。

したがって、$e > 2^{30}$ という数値には正当な根拠はあまりないと言える。しかしこのことは、e のいくつかの推奨パラメータ (8、2^{10} など) に対しても言えることである。

実際の実装による ESIGN 署名解読実験は、いくつかの例 [AA97, AA99] があるものの、素因数分解実験や、楕円離散対数計算実験ほどには多くの例がない。次章の素因数分解問題においても指摘するように、LLL アルゴリズムを実際に実装し、オーダによる漸近評価だけでなく、計算機実験を行うことが有益であると思われる。日本においてはディジタル署名の法的お墨付きに ESIGN 署名が使われる可能性があることから、実際の解読可能性を把握することは重要である。素因数分解や楕円離散対数のように、解読コンテストを実施することも有効であろう。

4 素因数分解問題

素因数分解問題とは、与えられた整数 n からその素因数を求める問題であるが、ESIGN 署名の場合、$n = p^2q$(n は ESIGN の法)、p と q はビット長が同じ素数、という設定に関する問題となる。素因数分解問題を解く 2 つの代表的アルゴリズム、数体ふるい法 [LL93] と楕円曲線法 [Le87, Mo87] に対する ESIGN 署名の素因数分解問題に関して考察する。

計算量評価には、次式のような漸近的計算量評価式がしばしば用いられる。

$$L_x[u, v] = e^{v(\log x)^u (\log\log x)^{1-u}} \tag{1}$$

あるアルゴリズムが $L_x[0, v]$ という実行時間であれば、それは多項式時間であると言い、計算

量は、アルゴリズムへの入力の大きさに関するある多項式で抑えられる。あるアルゴリズムが $L_x[1, v]$ という実行時間であれば、それは指数時間であると言い、計算量はアルゴリズムへの入力の大きさに関するある指数関数で抑えられる。$0 < u < 1$ の時、準指数時間と呼ばれる。

数体ふるい法の計算量は、素因数分解したい整数の大きさに関する準指数時間で、次の計算量をもつ。

$$L_n[1/3, 1.9292 + o(1)] \tag{2}$$

一般化数体ふるい法によって実際に素因数分解された数の記録は 512 ビットである。

楕円曲線法の計算量は、素因数分解したい整数 n のもっとも小さい素因数を p とすれば、

$$L_p[1/2, \sqrt{1/2}] \tag{3}$$

となる。楕円曲線法によって分解されたもっとも大きな素因数は、$(6^{43} - 1)^{42} + 1$ という数の 54 decimal digit の素数である。

また、Boneh らが大きい r に対する $n = p^r q$ の素因数分解問題を解くアルゴリズムを示している [BDG99]。そこでは、LLL アルゴリズムが利用されている。Boneh らのアルゴリズムは、r が大きい場合 $(|r| = (\log p)^{1/2})$ に有効であり、$n = p^2 q$ に対する素因数分解は、数体ふるい法や楕円曲線法よりも効率が悪い。

現時点では、ESIGN 署名の法 $n = p^2 q$ の大きさが 1024 ビット以上であれば、数体ふるい法に対しても楕円曲線法に対しても安全、すなわち現実的時間、記憶域量で素因数分解することは不可能であると思われる。しかしながら、アルゴリズムの計算量評価は前記の式のような漸近的評価であり、実際に素因数分解を実行した場合、どの程度の実時間、実記憶域量を要するかを見積もることは容易ではない。ESIGN 署名のような $n = p^2 q$ の形の合成数に対する素因数分解が、どの程度のビット長の時、数体ふるい法と楕円曲線法とどちらが有効に働くかを、実際に素因数分解実験を行って確かめることも有益であると思われる。

5 ハッシュ関数

ESIGN 署名は、他の多くのディジタル署名スキームと同様、スキームの補助関数としてハッシュ関数が必要である。ハッシュ関数の ESIGN 署名提案者による推奨は、SHA-1 である [NTT01a]。

最近 NIST において、SHA-1 に代わる新しいハッシュ関数の標準化が進められている。すなわち、SHA-1 のハッシュ値は 160 ビットであるが、より大きいハッシュサイズのハッシュ関数が作られている。その関数は SHA-256、SHA-384、SHA-512 と呼ばれ、それぞれハッシュサイズは 256 ビット、384 ビット、512 ビットである。

コリジョンサーチによるハッシュ関数の攻撃の計算量は、ハッシュサイズの平方根となる。ハッシュ関数へのコリジョンサーチ攻撃と、素因数分解攻撃とのバランスを考えると、数体ふるい法による素因数分解攻撃の計算量評価式が式 (2) であるとすれば、ハッシュ値 160 ビットと法 1024 ビットとがだいたいバランスすることになる。

したがって、ディジタル署名のセキュリティレベルを 1024 ビット以上に求めるならば、ESIGN 署名の法サイズを 1024 ビット以上にするとともに、ハッシュ関数も SHA-1 の代わりに、SHA-256、SHA-384、SHA-512 にすることも検討しなければならない。

6 各標準化機関における ESIGN 署名の仕様

ESIGN 署名は、CRYPTREC 以外にも様々な標準化機関へ投稿されている。その仕様名は、最初に ESIGN 署名が発表されて以来、何度かの変更が行われている [NTT01a]。参考までにそれらの仕様の違いを次表にまとめる。年数経過とともに、推奨パラメータはより大きくなっており、また安全性理論研究の進展によって、ディジタル署名における最強の意味での安全性を (ある仮定のもとで) 証明できるスキームに変更されている。より安全サイドに最も寄った仕様は、IEEE P1363a と NESSIE である。

表 1: 各標準化機関へ投稿されている ESIGN の比較

標準化機関	推奨パラメータ	エンコーディング方法
電子署名法	$\|n\| \geq 1024$、$e \geq 8$	EMSA
ISO 14888-3	$e \geq 4$、n は規定なし	—
IEEE P1363a	規定なし	EMSA5
NESSIE	$\|n\| = 1152$、$e = 2^{10}$	EMSA5
CRYPTREC 2000	$\|n\| \geq 960$、$e \geq 8$	—
CRYPTREC 2001	$\|n\| = 1152$、$e = 2^{10}$	EMSA

7 まとめ

ESIGN 署名は、適切なパラメータを選択すれば、現段階では安全であると考えられる。しかし、ESIGN 署名が安全性の根拠として帰着してるいる e 乗根近似問題の解法研究は、素因数分解問題や離散対数問題の解法研究ほど活発に行われているとは言い難い。ESIGN 署名は、日本における電子署名法の署名方式の 1 つに指定されていることや、国際的にも IEEE や ISO や NESSIE(1st stage) へ採録されるなど、知名度は高いと思われる。したがって、今後より活発に行われる研究対象になる可能性がある。活発に研究が行われた場合、予期せぬブレークスルーが生じ、ESIGN 署名の安全性が致命的に崩壊する可能性は否定できない。今後の研究動向を注意深くフォローする必要がある。

参考文献

[AA97] 安藤 心, 荒木 純道, “LLL アルゴリズムによる ESIGN の偽造確率に関する数値的検討,” 1997 年電子情報通信学会 基礎境界ソサイエティ大会, A-7-15,p.140 (1997)

[AA99] 安藤 心, 荒木 純道, “有限環上の 2 次不等式解法に関する数値的検討,” 1999 年電子情報通信学会 総合大会, A-7-3,p.231 (1999)

[BD85] E. Brickel, J. DeLaurentis, “An Attack on a Signature Scheme proposed by Okamode and Shisaishi,” *Advances in Cryptology – CRYPTO'85*, LNCS, **218** (1986), Springer-Verlag, 28–32.

[BDG99] D. Boneh, G. Durfee, N. Howgrave-Graham, "Factoring $N = p^r q$ for large r," *Advances in Cryptology – CRYPTO'99*, LNCS, **1666** (1999), Springer-Verlag, 326–337.

[Co96a] D. Coppersmith, "Finding a Small Root of a Univeriate Modular Equation," *Advances in Cryptology – EUROCRYPT'96*, LNCS, **1070** (1996), Springer-Verlag, 155–165.

[Co96b] D. Coppersmith, "Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known," *Advances in Cryptology – EUROCRYPT'96*, LNCS, **1070** (1996), Springer-Verlag, 178–189.

[Cr00] "ESIGN 署名攻撃評価報告書," 2000 年度 CRYPTREC 詳細評価報告書, (2000)

[IEEE] "IEEE P1363a Draft Version 9 Standard Specifications for Public Key Cryptography: Additional Techniques," *IEEE* (2001), available from http://grouper.ieee.org/groups/1363

[Le87] H.W. Lenstra, Jr, "Factoring Integers with Elliptic Curves," *Ann. of Math.*, (2) **126** (1987), 649–673.

[LL93] A.K. Lenstra, H.W. Lenstra, Jr, "The Development of the number field sieve," *Lecture Note in Mathematics*, **1554** (1993), Springer-Verlag.

[Mo87] P. L. Montgomery, "Speeding the Pollard and Elliptic Curve Methods of Factorization," *Mathematics of Computation*, **48** (1987), 243–264.

[NTT01a] NTT 情報流通プラットフォーム研究所, "ESIGN 仕様書," (2001), available from http://info.isl.ntt.co.jp/esign/CRYPTREC/index-j.html

[NTT01b] NTT 情報流通プラットフォーム研究所, "ESIGN 自己評価書," 2001 年度 CRYPTREC 応募書類, (2001)

[VGT88a] B. Vallée, M. Girault, P. Toffin, "How to Break Okamoto's Cryptosystem by Reducing lattice Bases," *Advances in Cryptology – EUROCRYPT'88*, LNCS, **330** (1988), Springer-Verlag, 281–291.

[VGT88b] B. Vallée, M. Girault, P. Toffin, "How to Guess lth Roots Modulo n by Reducing Lattice Bases," *AAECC-6*, LNCS, **357** (1988), Springer-Verlag, 427–442.

A ESIGN 署名の仕様

2001 年度 CRYPTREC 投稿版 [NTT01a] の ESIGN 署名の仕様の概略を記述する。

A.1 鍵生成

入力: $pLen$ セキュリティパラメータ (正整数)
e 8 以上の指数 (正整数)
出力: PK 公開鍵 $(n, pLen, e)$
SK 秘密鍵 (p, q)
処理手順:
Step 1 $pLen$ ビットの 2 つの素数 p、q を選ぶ
Step 2 $n = p^2 q$ を計算する
Step 3 $PK = (n, pLen, e)$、$SK = (p, q)$ を出力する

A.2 署名生成プリミティブ

SP-ESIGN
入力: SK 秘密鍵 (p, q)
PK 公開鍵 $(n, pLen, e)$
f メッセージ、$0 \leq f < 2^{pLen-1}$ である整数
出力: s 署名、$0 \leq s < n$ である整数
処理手順:
Step 1 $\mathrm{GCD}(r, n) = 1$ を満たす $r \in \{1, 2, \cdots, pq - 1\}$ をランダムに選ぶ
Step 2 $z = f \cdot 2^{2 \cdot pLen}$ を計算する
Step 3 $\alpha = (z - r^e) \bmod n$ とする
Step 4 $w_0 = \left\lceil \frac{\alpha}{pq} \right\rceil$ を計算する
Step 5 $t = \frac{w_0}{e \cdot r^{e-1}} \bmod p$ とし、$s = r + tpq$ を計算する
Step 6 s を出力する

A.3 署名検証プリミティブ

VP-ESIGN
入力: PK 公開鍵 $(n, pLen, e)$
s 署名、$0 \leq s < n$ である整数
出力: f 検証データ、$0 \leq f < 2^{pLen-1}$ である整数
処理手順:
Step 1 $T = s^2 \bmod n$ を計算する
Step 2 $f = \left\lfloor \frac{T}{2^{2 \cdot pLen}} \right\rfloor$ を計算する
Step 3 f が $0 \leq f < 2^{pLen-1}$ でなかったら、“invalid”と出力して終了
Step 4 f を出力する

A.4 署名生成スキーム

SSA-ESIGN-Sign

入力: SK 秘密鍵 (p, q)
PK 公開鍵 $(n, pLen, e)$
M 署名対象メッセージ、オクテット列

出力: s 署名

処理手順:

`Step 1` EMSA-ESIGN-Encode を用いて、メッセージ M を整数 f に変換する
$f = \text{EMSA-ESIGN-Encode}(M, pLen)$

`Step 2` SP-ESIGN を用いて、鍵 SK、PK と整数 f から整数 s を得る
$s = \text{SP-ESIGN}(SK, PK, f)$

`Step 3` s を出力する

A.5 署名検証スキーム

SSA-ESIGN-Verify

入力: PK 公開鍵 $(n, pLen, e)$
M 検証対象メッセージ、オクテット列
s 検証対象署名

出力: s “署名は有効” または “署名は無効”

処理手順:

`Step 1` VP-ESIGN を用いて、公開鍵 PK と整数 s から整数 f' を得る
$f' = \text{VP-ESIGN}(PK, s)$

`Step 2` EMSA-ESIGN-Verify を用いて、f' が M の正しいエンコード結果であるか調べる
$Result = \text{EMSA-ESIGN-Verify}(M, f', pLen)$

`Step 3` $Result$ が “矛盾しない”ならば “署名は有効”を出力する。
そうでなければ “署名は無効”を出力する。

A.6 エンコーディング処理

EMSA-ESIGN-Encode

オプション: $Hash$ ハッシュ関数
$hLen$ ハッシュ関数の出力長、ビット長

入力: M エンコード対象のメッセージ
l 出力長 (ビット長)

出力: f エンコード結果

処理手順:

`Step 1` エラー処理:$l - 16 - hLen$ ならば “エラー”を出力して終了

`Step 2` M にハッシュ関数を作用させ、ビット列に変換したものを H とする
$H = \mathrm{OS2BSP}(Hash(M), hLen)$
(OS2BSP はオクテット列をビット列に変換する関数)

`Step 3` ビット長 $l - 16 - hLen$ の任意のビット列を PS とする
ただし $\mathrm{BS2OSP}(PS, l - 16 - hLen)$ は各オクテットに 255 を含まないものとする

`Step 4` 各列を連結して F とする
$F = \mathrm{OS2BSP}(0, 8)||PS||\mathrm{OS2BSP}(255, 8)||H$

`Step 5` F を整数に変換する
$f = BS2IP(F)$
(BS2IP はビット列を整数に変換する関数)

`Step 6` f を出力する

A.7 検証処理

EMSA-ESIGN-Verify

オプション: $Hash$ ハッシュ関数
$hLen$ ハッシュ関数の出力長、ビット長

入力: M メッセージ、オクテット列
f 検証データ
l f の有効ビット長

出力: f “矛盾しない”または “不正”

処理手順:

`Step 1` 検証データ f をビット列 T へ変換する
$T = \mathrm{I2BSP}(f, l)$
(I2BSP は整数をビット列へ変換する関数)

`Step 2` T の下位 $hLen$ ビットを取り出してオクテット列へ変換したものを M' とする
$M' = \mathrm{BS2OSP}([T]_{hLen}, hLen)$
(BS2OSP はビット列をオクテット列に変換する関数)

`Step 3` $Hash(M)$ と M' が同じならば “矛盾しない”を出力する
そうでなければ “不正”を出力する