ACE署名攻撃評価報告書

目 次

1 はじめに

本報告では、Advanced Cryptographic Engine(ACE)[SS00] に含まれる電子署名方式である ACE 署名に対する攻撃評価を行う. 評価は、素因数分解問題 (正確には RSA 問題) の困難性を仮定した上での、適応的選択文書攻撃に対する耐性の評価、公開鍵に含まれる合成数の素因数分解に対する安全性の 2 点について行った.

2 ACE 署名の概要

Advanced Cryptographic Engine(以下 ACE) は公開鍵暗号方式だけでなく電子署名方式も実装しているソフトウェア・ルーチンライブラリである。以後本報告の対象となる電子署名方式を ACE 署名と呼ぶ.ACE 署名の特徴は、以下に示す 3 つの仮定のもと安全性の証明が可能であることである.

(1) Strong RSA 仮定.

(2) SHA-1 第 2 プレイメージ衝突耐性.

(4) MARS 累積/カウンタ・モード疑似乱数性.

またランダムオラクルモデル [FS87, BR93, BR94, PS96] を仮定した場合、(1) の Strong RSA 仮定に置き換えることが可能になる.

2.1 仕様書の概要

次に仕様書 [SS00] の概要をまとめる. 仕様書は、公開鍵暗号の ACE 暗号の内容も含まれており、また安全性に関する記述も書かれている.ACE 署名に関する部分は、

1 章 はじめに

2 章 安全性の目標、証明可能安全性、安全な電子署名についての解説.ACE 署名の安全性を証明する上で仮定されている、RSA 仮定、Strong RSA 仮定、SHA-1 第 2 プレイメージ衝突耐性、MARS 累積/カウンタ・モード疑似乱数性についての解説.

3 章 仕様書で用いている、用語と表記法についての説明.

4 章 基本的に ACE 暗号の仕様および安全性の解析についての記述であるが、ACE 署名で用いる汎用一方向性関数のアルゴリズムの記述がある.

5 章 電子署名方式のアルゴリズム説明および安全性の解析.

6 章 ASN.1 鍵の構文.

7 章 Power PC および Pentium 上での実装結果 (速度性能) の紹介.

このうち安全性に関しては、5 章において、最大時間 t 内に、最大 k 回の真の署名者に署名要求 (但し、署名対象文章全体のバイト長が最大 l である) を行い、そこで得た情報をもとに署名要求を行った文章以外の文章に対して署名の偽造を行う偽造者を想定し、偽造者が署名の偽造に成功する確率 $\mathbf{AdvSig}(t, k, l)$ を評価している.ACE 署名の安全性の根拠は、この確率 $\mathbf{AdvSig}(t, k, l)$ が無視できる程度であることである.

偽造者が署名の偽造に成功する確率 $\mathbf{AdvSig}(t, k, l)$ は、最大 t 時間内に Strong RSA 問題が解ける確率 $\mathbf{AdvFlexRSA}(t)$、最大 t 時間内に RSA 問題が解ける確率 $\mathbf{AdvRSA}(t)$、SHA-1 の第 2 プ

レイメージを見つける全てのアルゴリズムが最大 t 時間内にプレイメージを見つけることに成功する確率の最大値、$\mathbf{AdvSHA}(t)$、長さ l の MARS 累積/カウンタ・モードの出力と乱数を区別することができる確率、$\mathbf{AdvMARS}(t, l)$ を用いて評価している. 但し、証明の大きな流れのみが記述され、細かい部分の証明については、[CS99, Sho00a] 等の論文を参照している.

2.2 知られている攻撃法

現在のところ攻撃論文等の発表はない.

3 電子署名の安全性

本章では、ACE 署名の攻撃法の検討を行うにあたって、電子署名の安全性に関して整理を行う.

電子署名の安全性は、どのような偽造ができるか、攻撃者にどのような攻撃を許すか 2 つの観点から議論することができる.

まず最初の観点では、偽造できるレベルにより次のような安全性 (偽造困難性) が存在する.

1. **一般的偽造不可** (universally unforgeable): 署名の偽造のできない文章が存在する.
2. **選択的偽造不可** (selectively unforgeable): ある決められた文章以外に対しては署名の偽造ができない.
3. **存在的偽造不可** (existentially unforgeable): どのような文章に対しても署名の偽造ができない.

2 つ目の観点では、次の攻撃法が存在する.

1. **受動攻撃** (passive attck;key-only-attack): 公開鍵だけを用いて偽造を行う.
2. **一般選択文書攻撃** (generic chosen-message attck): 署名偽造者が前もって選んだ文章に対して真の署名者に署名させた後に、そこで得た乗法を用いて第 3 の文章の署名を偽造する攻撃.
3. **適応的選択文書攻撃** (adaptive chosen-message attack): 署名偽造者が毎回適応的に任意に選んだ文章に対して真の署名者に署名させ、最後にそこで得た情報を用いて第 3 の文章の署名を偽造する攻撃.

従って、最も安全な電子署名は、適応的選択文章攻撃に対して存在的偽造不可な署名法である [GMR88]. 簡単にいうと安全な電子署名とは、次に示す偽造モデルにおいて署名偽造者が電子署名の偽造に成功する確率が無視できる程度であるものをいう.

1. 公開鍵/秘密鍵のペアを生成し、攻撃者に公開鍵を与える.
2. 攻撃者は攻撃者の選択した文章を署名者に送り電子署名を得る、攻撃者はこの署名要求を (それ以前の結果に依存して) 適応的に多項式回数要求することができる.
3. 最後に攻撃者は署名要求を行った文章と異なる文章の署名を偽造し出力する.

4 攻撃対象

ACE 署名は,以下に示す、内部的に SHA-1[SHA95] を用いたハッシュ関数、内部的に MARS[BCD+98] を用いた素数生成関数と共に提案されている.

1. **アルゴリズム 5.5.1** 認証された素数の生成 GenCertPrime
   認証された 161 ビットの素数 e を、53 ビットの素数 P、整数 R を用いて $2PR+1$ の形で高速に生成する. また e が素数を保証するのみならず、e がある特定の方法で生成されていることを保証する正当性の認証子も同時に生成する. また内部的には、MARS 暗号 [BCD+98] を用いている.

2. **アルゴリズム 5.7.2** 素数認証の検証 CertPrime
   アルゴリズム 5.5.1 で生成された素数 e がある特定の方法で生成されていることを検証する.

3. **アルゴリズム 4.9.1** 汎用一方向性関数 UOWHash
   任意の長さの k および文章 m より 160 ビットのハッシュ値 h を出力する. 内部的には SHA-1[SHA95] を用いている.

4. **アルゴリズム 5.6.1** 汎用一方向性関数 UOWHash”
   内部的に UOWHash を用いて、任意の長さの k および文章 m より 160 ビットのハッシュ値 h を出力する

5. **アルゴリズム 5.6.2** 汎用一方向性関数 UOWHash”’
   内部的に UOWHash” を用いて、4 つの入力の組 $(k', l, x', \tilde{k}')$ より 160 ビット以下のハッシュ値 r を出力する

厳密な評価を行うためには、これらのアルゴリズムを含めて評価を行う必要があるが、評価対象が複雑になるという問題がある. また本報告では、素因数分解問題との関連を中心に評価を行うため、汎用一方向性ハッシュ関数および素数生成関数の内部構造に立ち入らなくてもすむよう、以下の簡略化を行った.

1. 161 ビット素数 e の生成に関しては、単にランダムに生成するものとする.

2. ハッシュ関数に関しては、内部構造に立ち入らないため、単に汎用一方向性ハッシュ関数の条件を満たしていることのみを条件とする.

次に本報告で評価対象とする、署名方式を示す. これは [CS99] に示されている汎用一方向性ハッシュ関数を用いた署名方式と同一である. 以後、特にことわりがない場合、以下の署名方式を ACE 署名と呼ぶ.

4.1 鍵生成

入力: サイズ・パラメータ m.

出力: 公開鍵 (n, h, x, e', k'), 秘密鍵 (p, q).

1. 素数 p', q' を用いて $p = 2p' + 1$, $2q' + 1$ と表すことのできる $\lfloor m/2 \rfloor$ ビットの素数 p, q をランダムに選択する.
2. $n = pq$ とおく.
3. $l + 1$ ビット素数 e' をランダムに生成する.
4. $h, x \in QR_n$ なる整数 h, x をランダムに生成する.
5. ハッシュ鍵 k' をランダムに生成する.
6. 公開鍵/秘密鍵のペア $((n, h, x, e', k'), (p, q))$ を出力し終了する.

なお ACE 署名の仕様書では $l = 160$, $1024 \leq m \leq 16,384$ が推奨されている. また $l + 1 < \lfloor m/2 \rfloor$ を満たすものとし、QR_n は n を法として平方剰余である整数からなる $\mathbb{Z}_n$ の部分群を表すものとする.

4.2 署名生成

入力: 文章 m, 公開鍵 (n, h, x, e', k'), 秘密鍵 (p, q).

出力: 署名 (e, y, y', k).

1. 文章 m のハッシュ値を計算するため、次の手順を実行する.
   1.1. ハッシュ鍵 k をランダムに生成する.
   1.2. l ビットの出力を持つ汎用一方向性ハッシュ関数関数 H を用いて $m_k = H_k(m)$ を計算する.
2. $y' \in QR_n$ なる整数 y' を生成する.
3. $x' = (y')^{e'} h^{-m_k}$ を計算する.
4. $e \neq e'$ なる $l+1$ ビット素数 e をランダムに生成する.
5. 汎用一方向性ハッシュ関数 H を用いて $r = H_{k'}(k, x')$ を計算する.
6. $y = (xh^r)^{1/e}$ を計算する.
7. 署名 (e, y, y', k) を出力し終了する.

4.3 署名検証

入力: 文章 m 及び対応する署名 (e, y, y', k), 公開鍵 (n, h, x, e', k').

出力: 署名が有効であれば Accept そうでなければ Regect を出力.

1. e が e' と異なる $l+1$ ビットの素数であるかどうかを調べる、条件を見なしていなければ、Reject を出力し終了.
2. $x' = (y')^{e'} h^{-H_k(m)}$ を計算する.
3. $x = y^e h^{-H_{k'}(k,x')}$ であれば Accept を返し、そうでなければ Reject を返して終了する.

5 攻撃方針

攻撃方法としては署名の偽造が考えられる. 本報告では、以下の 2 つの観点から評価を行う.

(1) 汎用一方向性ハッシュ関数及び flexible RSA 問題の困難性を仮定した上での、適応的選択文書攻撃に対する安全性.

(2) 公開鍵に含まれる RSA 型の合成数 n の素因数分解が可能かどうか.

(1) については、何らかな方法で、署名の偽造 (適応的選択文書攻撃の意味で) が行うことができる偽造者の存在を仮定する. このとき偽造者の存在が,Strong RSA 仮定および RSA 仮定に矛盾しないことが示されれば無視できない確率で署名の偽造が行えることになる.

(2) については公開鍵の内、合成数 n の素因数分解を行うことができれば, 秘密鍵である 2 つの素数 p, q を用いて署名を偽造することが可能になるため、ACE 署名において行われている鍵生成方法を検証し、何らかの素因数分解アルゴリズムの適用が可能であるかの検証を行う.

6 攻撃を行う上での仮定

本章では、適応的選択文章攻撃に対する安全性を評価する上で必要な仮 2 つの仮定をまとめる.

6.1 RSA 問題

本節では RSA 問題に関する定義を与える.

定義 6.1 (FACTORING). 正の整数 n が与えられた場合において $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ を満たす異なる素数 $p_1, p_2, \ldots, p_k$ および $e_i \geq 1$ $(1 \leq i \leq k)$ を見つける問題を素因数分解問題 *(FACTORING)* と呼ぶ.

定義 6.2 (RSAP). *2* つの異なる奇素数 p, q からなる合成数 $n = pq$, $\gcd(e, (p-1)(q-1)) = 1$ を満たす e、および $c \in \mathbb{Z}_n$ が与えられた場合において $m^e = c \pmod{n}$ を満たす $m \in \mathbb{Z}_n$ を見つける問題を *RSA* 問題 *(RSAP)* と呼ぶ.

RSA 問題を解くことが困難であるという仮定を RSA 仮定と呼ぶ.

定義 6.3 (flexible RSAP). *2*つの異なる奇素数 p, q からなる合成数 $n = pq$, $\gcd(e, (p-1)(q-1)) = 1$ を満たす e, および $c \in \mathbb{Z}_n$ が与えられた場合において $m^e = c \pmod{n}$ を満たす $m \in \mathbb{Z}_n$ を見つける問題を *flexible RSA* 問題 *(flexible RSAP)* と呼ぶ.

flexible RSA 問題を解くことが困難であるという仮定を strong RSA 仮定と呼ぶ. strong RSA 仮定は [BP97] において導入され、以後いくつかの暗号スキームの評価 [FO99, GHR99] に用いられている.

RSA 問題において n および r を固定すると、任意の $z \in \mathbb{Z}_n^*$ について $y = z^{1/r}$ を計算する問題は、$s \in \mathbb{Z}_n^*$ をランダムに選択し、$\tilde{z} = s^r z$ とおくと、$\tilde{y} = \tilde{z}^{1/r}$ を計算する問題に帰着できる. これにより、後者の問題に対して効率的なアルゴリズムが与えられれば、前者の問題を効率的に解けることになる. RSA 問題および flexible RSA 問題は、少なくとも合成数 n を素因数分解することにより解くことができるが、計算量的に等価であるかどうかは現在のところ知られていない.

6.2 汎用一方向性ハッシュ関数

汎用一方向性ハッシュ関数 (universal one-way hash function) は、Naor と Yung[NY89] により導入された概念で, 関数 H と、平文 x およびランダムな鍵 k が与えられた場合において、$H_k(x) = H_k(y)$ となるような x と異なる平文 y を求めることが難しいという条件を満たしている関数のことである.

明らかに分かるように、ハッシュ関数に対する要求条件としては、汎用一方向性ハッシュ関数の方が衝突困難ハッシュ関数よりも弱い.

7 適応的選択文書攻撃に対する安全性

本章では [CS99] において示されている以下の定理の証明を検証することにより ACE 署名の適応的選択文書攻撃に対する安全性の検証を行う.

定理 7.1. 汎用一方向性ハッシュ関数が存在するという仮定の下で、*flexible RSA* 問題が難しいと仮定すると、*ACE* 署名は、適応的選択文書攻撃に対して安全である.

定理の検証する前に、補題を示す.

補題 7.1. $x^a = y^b$ および $\gcd(a, b) = 1$ を満たす $x, y \in \mathbb{Z}_n^*$, $a, b \in \mathbb{Z}$ を与える. このとき $\tilde{x}^a = y$ を満たす $\tilde{x} \in \mathbb{Z}_n^*$ を効率的に計算できる.

(証明) 拡張ユークリッド互助法を用いて $bb' = 1 + ak$ を用いて整数 b' および k を計算する.

$$(x^a)^{b'} = (x^{b'})^a = y^{bb'} = y^{1+ak}.$$

次に $(x^{b'})^a = y^{1+ak}$ の両辺を y^{ak} で割ることにより

$$(x^{b'}y^{-k})^a = y.$$

$\tilde{x} = x^{b'}y^{-k}$ とおく、これにより補題が証明できた.

□

定理 7.1 の証明に戻る. まず t 回の署名要求の後、署名の偽造を行うアルゴリズムを考える.$m_i(1 \leq i \leq t)$ を i 番目に署名要求を行った文章、(e_i, y_i, y'_i, k_i) を i 番目の署名、x_i を $x'_i = (u'_i)^{e'}h^{-H_{k_i}(m_i)}$ と定義する. 次に (e,y,y',k) を文章 m ($m \neq m_i$ for all $1 \leq i \leq t$) に対する偽造署名, さらに $x' = (y')^{e'}h^{-H_k(m)}$ とする.

署名の偽造を次の 3 種類に分類する.

- **Type I**: ある j $(1 \leq j \leq t)$ に対して、$e = e_j$ かつ $(k, x') = (k_j, x'_j)$.
- **Type II**: ある j $(1 \leq j \leq t)$ に対して、$e = e_j$ かつ $(k, x') \neq (k_j, x'_j)$.
- **Type III**: 全ての i $(1 \leq i \leq t)$ に対して、$e \neq e_i$.

ここで全ての e_i は互いに異なり、また e' と異なることを仮定する.

無視できない確率で署名を偽造できる偽造者がいるとすれば、Type I,Type II,Type III の偽造の内いずれか一つが無視できない確率で偽造されているものとする.

証明の方針は、t 回の真の署名者に対する署名要求の後に署名の偽造を無視できない確率で行うことができる署名者が存在すると仮定する. このとき秘密鍵を知ることなく真の署名者を模倣できるシミュレーターが構成できるならば、偽造者とシミュレーターが共謀して、RSA 問題または、flexible RSA 問題を効率的に解くことができるアルゴリズムが構成できる. このことは RSA 問題または、flexible RSA 問題が難しいという仮定に反するため、そのような偽造者は存在することができない. これより ACE 署名が適応的選択文書攻撃に対して安全であることが示される.

Type I および Type II の偽造署名を作成する偽造者をブラックボックスとして用いることにより RSA 問題を解くアルゴリズムが構成でき、Type III の偽造を行う偽造者をブラックボックスとして用いることにより flexible RSA 問題を解くアルゴリズムが構成できる. 以下 Type I,Type II,Type III のそれぞれの場合について議論を行う.

Type I

無視できない確率で Type I の偽造署名を生成できる偽造者が存在すると仮定する. このとき偽造者をブラックボックスとして用いて、効率的に RSA 問題の解く方法を示す. すなわち RSA 型の合成数 n, 乱数 $z \in \mathbb{Z}_n^*$ およびランダムな $l+1$ ビットの素数 r が与えられた場合において、$z^{1/r}$ を求める効率的な方法を示す.

最初に秘密鍵を知ることなく真の署名者を模倣するシミュレーターの動作を記述する. まず $l+1$ ビット素数 $e_1, \ldots, e_t$ を用いて公開鍵を次のように作成する.

$$h = z^{2\prod_i e_i}.$$

次に $w \in \mathbb{Z}_n^*$ をランダムに選択し x を次のように計算する.

$$x = w^{2\prod_i e_i}$$

最後に、ハッシュ鍵 k' を選択し、$e' = r$ とおく.

文章 m_i に署名を行う。シュミレーターは $y'_i \in QR_n$ をランダムに選択し、$x'_i = (y'_i)^{e'} h^{-H_{k_i}(m_i)}$ を計算する. 次にシュミレーターは $y_i^{e_i} = xh^{H(x'_i)}$ を y_i を解く、これは $x^{1/e_i} = w^{2\prod_{j\neq i} e_j}$ および $h^{1/e_i} = w^{2\prod_{j\neq i} e_j}$ であることを用いることにより容易に計算できる. 以上により、シミュレーターは攻撃者から見た真の署名者を完全に模倣することができる.

次に、偽造者が文章 m に対する Type I の偽造署名 (e, y, y', k) を偽造したと仮定する. このときある j $(1 \leq j \leq t)$ に対して、$e = e_j$ および $x' = s'_j$ が成り立つ. これより次の 2 つの式が導かれる.

$$\begin{aligned} (y')^{e'} &= x' h^{H_k(m)}, \\ (y'_j)^{e_j} &= x' h^{H_{k_j}(m_j)}. \end{aligned}$$

ハッシュ関数 H の汎用一方向性および $k = k_j$ というから、$H_k(m) \neq H_{k_j}(m_j)$ であると仮定できる. 2 つの式の両辺をそれぞれ割ることにより次の式を満たす $v \in \mathbb{Z}_n^*$ および a を求めることができる.

$$v^{e'} = z^{2a\prod_i e_i} \ (= h^a).$$

$H_k(m),\ H_{k_j}(m_j)$ の出力は l ビット以下の整数であること、e' が $l+1$ ビット素数であることから、$a \not\equiv 0 \pmod{e'}$ であることが容易に示せる. よって $\gcd(2a\prod_i e_i, e') = 1$ となり、$e' = r$ と補題 7.1 より、z の r 乗根を容易に計算できる.

Type II

Type I の場合と同じく、偽造者をブラックボックスとして用いて RSA 型の合成数 n, 乱数 $z \in \mathbb{Z}_n^*$ および r に対して $z^{1/r}$ を求めるアルゴリズムを構成する. また Type II の偽造署名における j の値は固定されているとする、そうでない場合は、j の値を推測する.

次にシュミレーターの動作を記述する. まず公開鍵を次のように生成する.$i \neq j$ を満たす $i(1 \leq i \leq t)$ に対して、$(l+1)$ ビットの素数 e_i を選択する.$e_j = r$ とおき、ランダムな $l+1$ ビット素数 e' およびランダムなハッシュ鍵 k' を選択する.h を次のようにおく.

$$h = z^{2e'\prod_{i\neq j} e_i}.$$

$w \in \mathbb{Z}_n^*$ をランダムに選択し y'_j を次のようにおく.

$$y_j = w^{2\prod_{i\neq j} e_i}.$$

$u \in \mathbb{Z}_n^*$ をランダムに選択し x'_j を次のようにおく.

$$x'_j = u^{2e'}.$$

ランダムにハッシュ鍵 k を選択し、x を計算する.

$$x = y_j^{e_j} h^{-H_{k_j}(x'_j)}.$$

次に文章 m_i に署名の方法を記述する. 最初に $i \neq j$ を仮定する. シュミレーターは $y'_i \in QR_n$ をランダムに選択し、$x'_i = (y'_i)^{e'} h^{-H_{k_i}(m_i)}$ を計算する. 次にシュミレーターは $y_i^{e_i} = xh^{H_{k'}(k_i, x'_i)}$ を y_i を解く、これは x および h の e_i 乗根を知っているため容易に計算できる. 次に $i = j$ を仮定する.h および x'_j の e' 乗根を知っているため y'_j の正しい値を計算できる.

これによりシミュレーターの記述が完了する.

偽造者が文章 m に対して $e = e_j$ および $(k, x') \neq (k_j, x'_j)$ を満たす Type II の偽造署名 (e, y, y', k) を生成したと仮定する. このとき次の 2 式が成り立つ.

$$y^e = xh^{H_{k'}(k, x')}, \tag{1}$$

$$y_j^e = xh^{H_{k'}(k_j, x'_j)}. \tag{2}$$

ここで Type I と同様の議論を行う前に、$H_{k'}(k, x') = H_{k'}(k_j, x'_j)$ となる確率が無視できる程度であることを証明する必要がある.

証明は、$H_{k'}(k, x') = H_{k'}(k_j, x'_j)$ を満たす (k, x') を無視できない確率で見つけることができる攻撃者の存在を仮定する. このとき異なるシミュレーターを用いて H の汎用一方向性を破ることができることを示す. このことは汎用一方向性ハッシュ関数が存在するという仮定に反することから、そのような攻撃者は存在しないことが示される.

次に、この新しいシミュレーターの動作を記述する. 最初にシミュレーターは署名用に公開鍵・秘密鍵のペアをハッシュ鍵 k' の選択を除いて行う. 次にシミュレーターは Type II の偽造署名において定義されている j の値を推定し、ランダムに $x'_j \in QR_n$ をランダムなハッシュ鍵 k'_j とともに選択する.

ハッシュ鍵 k' が選択された後、シミュレーターは攻撃者をブラックボックスとして用いて衝突 $H_{k'}(k, x') = H_{k'}(k_j, x'_j)$ を見つける. 次にシミュレーターはハッシュ鍵 k' を加えることにより公開鍵を完成させる. 以後、攻撃者は公開鍵を用いることができるようになる. シミュレーターは署名用の秘密鍵を知っているため、$1 \leq i \leq t$ および $i \neq j$ を満たす i 番目の署名を容易に生成できる.$i = j$ となる場合、シミュレーターは、(k_j, x'_j) が既に選択されている (k_j, x'_j) に等しくなるよう平文 m_j に対して署名を行う. シミュレーターは n の素因数分解結果を用いることができるため容易に行える. 次にシミュレーターは素数 e_j を生成し、k_j,x'_j,e_j および k' を用いて $y_j^{e_j} = xh^{H_{k'}(k,x'_j)}$ および $(y'_j)^{e'_j} = x'h^{H_{k_j}(m_j)}$ を y_j および y'_j について解く. これによりシミュレーターの動作の記述が終了する. よって衝突 $H_{k'}(k, x') = H_{k'}(k_j, x'_j)$ を見つけることができる攻撃者が存在するとすれば、H の持つ汎用一方向性を破ることができる. このことは汎用一方向性ハッシュ関数が存在するという仮定に反するため、そのような攻撃者は存在しない. 以上の議論より $H_{k'}(k, x') \neq H_{k'}(k_j, x'_j)$ を仮定できる.

次に (1) ,(2) の両辺を割ることにより次の式を満たす $v \in \mathbb{Z}_n^*$ および a を求めることができる.

$$v^e = z^{2a\prod_{i\neq j} e_i} \ (= h^a).$$

$H_{k'}(k, x')$ $H_{k'}(k_j, x'_j)$ の出力は l ビット以下の整数であることと e が l ビットの素数であることから、$a \not\equiv 0 \pmod{e'}$ であることが容易に示せる. よって $\gcd(2a\prod_{i\neq j} e_i, e) = 1$ となり、$e = r$ と補題 7.1 より、z の r 乗根を容易に計算できる.

Type III

無視できない確率で Type III の偽造署名を生成できる偽造者が存在すると仮定する. このとき、効率的に flexible RSA 問題を効率的に解く方法を示す. すなわち、$n, z \in \mathbb{Z}_n^*$ が与えられた場合において、$r > 1$ および $z^{1/r}$ を求める.

シミュレーターの動作を説明する. ランダムに $(l+1)$ ビット素数 $e', e_1, \ldots, e_t$ を生成する. 次に示すように h を計算する.

$$h = z^{2e'\prod_i e_i}.$$

次に $a \in \{1, \ldots, n^2\}$ をランダムに選択し、$x = h^a$ を計算する. 構成法により QR_n は位数 $p'q'$ の巡回群となる.

ここで $a = bp'q' + c$ $(0 \leq c < p'q')$ とおく. このとき a は十分に広い区間よりランダムに選択されているため、c の分布と $\{0, \ldots, p'q' - 1\}$ 上の一様分布は統計的に区別することができない. さらに b の分布と $\{0, \ldots, \lfloor n^2/p'q' \rfloor\}$ 上の一様分布は統計的に区別することができない. このことより c と b は本質的に独立であると見なせる.

c の分布が一様分布と見なせることにより、x は QR_n 上のランダムな要素と見なせる. 鍵生成の最後にハッシュ鍵 k' をランダムに選択する. また、Type I および Type II の場合と同様に x および h の e_i 乗根を容易に計算できるため、全ての文章 m_i に対して署名することができる.

偽造者が Type III の偽造署名 (e, y, y', k) を生成したと仮定する. このとき

$$y^e = xh^{H_{k'}(k,x')} = z^s,$$

但し

$$s = 2e' \prod_i e_i \cdot (a + H_{k'}(k, x')).$$

$d = \gcd(e, s)$ とおく. このとき $\gcd(d, 2p'q') = 1$ より $y^{e/d} = z^{s/d}$. よって補題 7.1 より z の (e/d) 乗根を求めることができる. ここで署名の生成方法より、e は $l+1$ ビットの素数であるから d は e または 1 となる. これより無視できない確率で $e \not| s$ であることを示せば十分である.Type III の偽造署名の満たしている条件より、$e \not| 2e' \prod_i e_i$ が示される. よって $e \not| (a + H_{k'}(k, x'))$ が無視できない程度の確率で起こることを示せば十分である. ここで $a = bp'q' + c$ と書けること、e は c に依存するかもしれないが b と c は本質的に独立と見なせることに注意する. $l+1$ ビットの素数 e は素数 p' および素数 q' より小さいため $e \not| p'q'$ が成り立つことより、$a + H_{k'}(k, x') \equiv 0 \pmod{e}$ となる確率は $1/e$ に近いものとなる. これより無視できない確率で $r \not| (a + H_{k'}(k, x'))$ であることが示される. 以上により z の e 乗根を求めることができる.

8 素因数分解問題に対する安全性

ACE 署名は、公開鍵に含まれる合成数 n を素因数分解し、2 つの素数 p,q を得ることができれば容易に署名を偽造することができる. 本章では素因数分解に対する安全性の検証を行う.

8.1 素因数分解の概要

素因数分解のアルゴリズムは大きく分けて、次の 2 種類に分類することができる.

1. 合成数 n の素因数分解の計算量が、n の大きさに依存するアルゴリズム. 例としては、2 次ふるい法、数体ふるい法などがある [Len87, Po75].
2. 合成数 n の素因数分解の計算量が、n の素因数 p の性質に依存するアルゴリズム. 例としては、$p-1$ 法、$p+1$ 法、楕円曲線法などがある [LLMP90, Pom85, Sil87].

前者の分類に属する、数体ふるい法は

$$x^2 \equiv y^2 \pmod{n}$$

を満たす x,y の組を求める. 次に $\gcd(x \pm y, n)$ を計算して n の素因数を求める方法で、RSA 暗号で用いられるような大きな素数からなる合成数に対する素因分解アルゴリズムとしては現在最も高速である. その計算量は、

$$L_n[\alpha, c] = O\left(e^{(c+o(1))(\log n)^{\alpha}(\log\log n)^{1-\alpha}}\right).$$

を用いると、数体ふるい法の計算量は、$L_n[1/3, (64/9)^(1/3)]$ となる. また 2 次ふるい法についても数体ふるい法と同様に、$x^2 \equiv y^2 \pmod{n}$ を満たす x,y の組を求めることにより素因数分解を行う. 計算量的には数体ふるい法に劣り $L_n[1/2, 1]$ である.

合成数のサイズに依存する素因数分解法は n のサイズを大きくすることにより避けることが可能になることから、本報告では、素因数 p の性質に依存するアルゴリズムを中心に評価する. 以下、代表的な、素因数依存型アルゴリズムについて簡単に説明する.

8.1.1 $p-1$ 法

$p-1$ 法は合成数 n が、$p-1$ が小さな素数の冪の積となる素数 p を素因数として持つときに有効な方法である. 素因数分解の原理は、$\mathbb{Z}_p$ の 0 でない元が、乗法群 $\mathbb{Z}_p^*$ 上で位数 $p-1$ の群をなし、従って $p-1|k$ となるならば、$a^k \equiv 1 \pmod{p}$(但し a は $0 < a < n$ を満たす任意の整数) を満たす事実に基づいている. この事実より $a^k - 1$ は p の倍数となり、$\gcd(a^k, n)$ を計算することにより p を求めることができる.

8.1.2 $p+1$ 法

$p+1$ 法は合成数 n が、$p+1$ が小さな素数の冪の積となる素数 p を素因数として持つときに有効な方法である. 素因数分解の原理は、$\gcd(n, b(a^2+4b)) = 1$ を満たす整数 a, b を用いて、フィボナッチ数列の拡張である数列 y_n を次のように定める.

$$y_0 = 0, \quad y_1 = 1, \quad y_{i+1} = ay_i + by_{i-1}$$

このとき、$d = a^2 + 4b$ が　$\bmod p$ で平方剰余であるかどうかに従って、$y_{p-1} \equiv 0 \pmod{p}$ または $y_{p+1} \pmod{p}$ となる. ここで $y_{q+1} \equiv 0 \pmod{p}$ となるとする. このとき、$q+1$ の倍数 m に対して $y_m \equiv 0 \pmod{p}$ となる. ここで $p-1$ 法と同様に、$\gcd(y_m, n)$ を計算することにより素因数分解が可能になる.

8.1.3 楕円曲線法

楕円曲線法は、$p-1$ 法の一般化で、群 $\mathbb{Z}_p^*$ を有限体 $\mathbb{F}_p$ 上の楕円曲線上の点のなす群 $C(\mathbb{F}_p)$ に、また整数 a を点 $P \in C(\mathbb{F}_p)$ で置き換えるものである. 素因数分解の方法は $p-1$ 法の場合と同様に、小さい素数の積からなる整数 k をとる、このとき $C(\mathbb{F}_p)$ の元の数が偶然 k を割り切っているならば、$C(\mathbb{F}_p)$ 上で $kP = \mathcal{O}$(但し $\mathcal{O}$ は $C(\mathbb{F}_p)$ の単位元) となる. このとき $p-1$ 法と同様にして、n の非自明な約数が得られる. 楕円曲線法が $p-1$ 法に対する利点は、p を固定した場合楕円曲線上の点の個数 $\#C(\mathbb{F}_p)$ は

$$p + 1 - 2\sqrt{p} \ \le \ \#C(\mathbb{F}_p) \ \le \ p + 1 + 2\sqrt{p}$$

の範囲に一様に分布するため $p-1$ 法と異なり p を固定したとしても楕円曲線の選択の自由度が大きく、ある楕円曲線で失敗したとしても他の楕円曲線に取り換えることが可能であるため、素因数分解の成功する可能性が大きくなる.

また楕円曲線法は、後者に分類されるアルゴリズムの中で最も高速で、合成数 n の最大素因数を p とすれば、その計算量は、$L_p[1/2, \sqrt{2}]$ となる. このため楕円曲線法の計算量は p のサイズ $\log p$ に依存するため、p のサイズが小さければ n が大きい場合においても高速な素因数分解が可能となる.

合成数 n がほぼ同じサイズの素数 p, q の積となる場合は、計算量は、$L_n[1/2, 1]$ となり合成数 n のサイズに依存する. この計算量は、2 次ふるい法と同じであるが、楕円曲線上の演算が複雑なため、2 次ふるい法に比べて効率面で劣る.

8.2 ACE 署名の素因数分解に対する安全性

最初に ACE 署名と同じタイプの合成数を用いる RSA 暗号の鍵生成の条件を述べる. RSA 暗号の鍵生成では、素数 p, q を生成する必要がある. このとき、p, q は以下の条件を満たすことが推奨されている.

- $p-1$ $(q-1)$ が大きな素数を素因数に持つ. (大きな素数を r とする).

- $p+1$ $(q+1)$ が大きな素数を素因数に持つ.
- $r-1$ が大きな素数を素因数に持つ.

最初の 2 つの条件は、$p-1$ 法および $p+1$ 法に対する安全性を考慮したもので、最後の条件は、周期攻撃 [SN77] を考慮したものとなっている.

しかし、$p-1$ 法の一般化である楕円曲線法に対しては、上記の条件だけでは不十分であり、また条件を列挙することも困難である. しかし RSA 型の合成数に対する計算量が $L_n[1/2, 1]$ であり、数体ふるい法に比べて計算量的に劣ることから、十分に大きなサイズの p, q をランダムに選ぶことで回避できると考えられる.

次に ACE 署名における鍵生成について述べる.ACE 署名の鍵生成では RSA 暗号と同じく、2 つの素数 p, q を生成する必要がある.ACE 署名の仕様書では、生成にあたっては素数 p', q' を用いて $p = 2p'1$ および $q = 2q' + 1$ で表される同じサイズの素数 p および q を選択するとしている. また合成数 n のサイズは 1023〜16, 384 ビットを推奨しており、現在のところ RSA 暗号が 1024 ビット程度以上の合成数を用いた場合、安全であると考えられていることを考慮すると、数体ふるい法等の合成数のサイズに依存する素因数分解法に対して十分な安全性を持つと考えられる.

次に $p-1$ 法に対しては、$p' = (p-1)/2$ および $q' = (q-1)/2$ が十分に大きな素数であるから安全であるが、$p+1$ 法に対しては、$p+1$ および $q+1$ が大きな素数を素因数に持っている保証はできないため、安全性に問題が生じる可能性がある.

9 結論

本報告では、簡略化した ACE 署名に対して、以下の 2 つの観点から評価を行った.

(1) 汎用一方向性ハッシュ関数及び flexible RSA 問題の困難性を仮定した上での、適応的選択文書攻撃に対する安全性.

(2) 公開鍵に含まれる RSA 型の合成数 n の素因数分解が可能かどうか.

(1) については [CS99] に基づき ACE 署名の適応的選択文書攻撃に対する安全性の検証を行った. 結果、本質的な問題点は特になく適応的選択文書攻撃に対して安全であることが検証できた. (2) については、合成数の大きさが 1023〜16, 384 ビットと大きいため数体ふるい法に対しては安全である事が示された. また $(p-1)/2$ が素数となるように素数 p を選択しているため $p-1$ 法に対しては安全であることが示されるが、$p+1$ 法に対しては、$p+1$ が大きな素数を素因数に持っている保証はできないため、安全性に問題が生じる可能性がある. 但し (1) の観点からの評価では問題がないため、素数生成時に、$p+1$ 法に対する条件を付け加えることにより問題は回避できる.

参考文献

[BCD+98] C. Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S. Matyas Jr., L. O'Connor, M. Peyravian, D. Safford, and N. Zunic. MARS–a candidate cipher for AES. June 1998.

[BP97] N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology–Eurocrypt '97*, pages 480–494, 1997.

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *First ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[BR94] M. Bellare and P. Rogaway. Optimal asymmetric encryption. In *Advances in Cryptology–Crypto '94*, pages 92–111, 1994.

[Bra93] S. Brands. An efficient off-line electronic cash system based on the representation problem, 1993. CWI Technical Report, CS-R9323.

[BS96] E. Bach and J. Shallit. Algorithmic Number Theory, volume 1. MIT Press, 1996.

[CGH98] R. Canetti, O. Goldreich, and S. Halevi. The random oracle model, revisted. In *30th Annual ACM Symposium on Theory of Computing*, 1998. 45

[CHJ99] D. Coppersmith, S. Halevi, and C. Jutla. ISO 9796-1 and the new forgerystrategy. Unpublished manuscript, 1999.

[CS99] R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. In *6th ACM Conf. on Computer and Communications Security*, 1999.

[Dam87] I. Damgard. Collision free hash functions and public key signature schemes. In *Advances in Cryptology–Eurocrypt '87*, 1987.

[DH76] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Info. Theory*, 22:644–654, 1976.

[FO99] E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Advances in Cryptology–Crypto '97*, 1999.

[FS87] A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. *In Advances in Cryptology–Crypto '86*, Springer LNCS 263, pages 186–194, 1987.

[GHR99] R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In *Advances in Cryptology–Eurocrypt '99*, pages 123–139, 1999.

[GMR88] S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17:281–308, 1988.

[Kra94] H. Krawczyk. LFSR-based hashing and authentication. In *Advances in Cryptology–Crypto '94*, pages 129–139, 1994.

[LLMP90] A.K. Lenstra, H.W. Lenstra Jr, M.S. Manasse, J.M. Pollard. The Number Field Sieve. *Proc of STOC*, pages 564–572, 1990.

[Len87] H.W. Lenstra Jr. Factoring Integers with elliptic Curves. In *Ann. of Math*,127:649–673, 1987.

[MvOV97] A. Menesez, P. van Oorschot, and S. Vanstone. Handbook of Applied Cryptography. CRC Press, 1997.

[NR97] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th Annual Symposium on Foundations of Computer Science*, 1997.

[NY89] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *21st Annual ACM Symposium on Theory of Computing*, 1989.

[OY97] 岡本龍明 山本博資. 現代暗号, 産業図書、 1997.

[Po75] J.M. Pollard. Monte Carlo method for Factorization. *BIT*,15:918–924, 1975

[Pom85] C.Pomerance. The Quardratic Sieve Algorithm. LNCS 209, pages 169–182 1985.

[PS96] D. Pointcheval and J. Stern. Security proofs for signature schemes. In *Advances in Cryptology–Eurocrypt '96*, pages 387–398, 1996.

[SHA95] Secure hash standard, National Institute of Standards and Technology (NIST), FIPS Publication 180-1, April 1995.

[Sho00a] V. Shoup. A composition theorem for universal one-way hash functions. In *Advances in Cryptology–Eurocrypt 2000*, 2000.

[Sil87] R.D. Silverman. The Multiple Polynomial Quardratic Sieve. In *Math. Comp.*, pages 143–264, 1987.

[Sim98] D. Simon. Finding collisions on a one-way street: can secure hash functions be based on general assumptions? In *Advances in Cryptology–Eurocrypt '98*, pages 334–345, 1998.

[SS00] T. Schwecinberger and V. Shoup. ACE: The Advanced Cryptographic Engine IBM, 2000.

[SN77] G.j. Simmons and M.j.Norris. Preliminary Comments on the M.I.T. Public-Key Cryptosystem. In *Cryptologia.* 1:406–414, 1977.

[WA97] 和田秀男. コンピュータと素因子分解, 遊星社、1987