自己評価書
HIME(R) 暗号

(株) 日立製作所

概 要

本ドキュメントは，公開鍵暗号 HIME(R) の自己評価に関するものである．

HIME(R) は，合成数 $N = p^d q$ (但し，p,q は素数．$d > 1$) を法とする剰余環上のモジュラー平方関数（Rabin 暗号化関数）を暗号化関数とし，復号化に高速計算方法を用いている．また，セキュリティ強化のために，OAEP[3] を利用している．

HIME(R) は，次の優れた特徴を持つ．

- 合成数 $N = p^d q$ の素因数分解問題の困難性を仮定として，ランダムオラクルモデルの下で，適応的選択暗号文攻撃に対して強秘匿（IND-CCA2）であることが証明できる．
- 非常に高速な暗号化処理が可能（１回のモジュラー積のみ）．
- 復号化速度について，HIME(R) (1536 bits) は RSA-OAEP (1024 bits)[3] に対して，約 2.5 倍の高速処理が可能（モジュラー積の個数による比較）．
- RSA-OAEP と同等以上の十分大きな平文空間を持つ．
- 合成数 N のサイズを大きく選んでも，従来方式に比べ，暗号化・復号化の効率性を損なわない．

このように，HIME(R) は素因数分解問題の困難性を前提として安全性証明可能な実用的公開鍵暗号方式である．

本ドキュメントでは，HIME(R) の安全性及び性能評価結果の詳細について述べる．

目 次

1 設計方針および概要

HIME(R) の設計方針は次の通りである.

(1) 安全面：プリミティブな問題（素因数分解問題や離散対数問題のように十分な研究の下で計算量的困難性が予想されている問題）の計算量的困難性を仮定として，IND-CCA2 の意味で安全性が証明できること.

(2) 効率面：

(2-1) 暗号化および復号化処理スピードが速いこと.

(2-2) 平文と暗号文の比“ 平文/暗号文 ”が小さくならないようにすること.

(2-3) 平文空間が十分大きいこと.

(2-4) 公開鍵暗号として共通鍵暗号とのハイブリッドにしないこと（公開鍵暗号を実現させるために，共通鍵暗号を利用しない.）.

暗号学的仮定として利用する数論的問題としては，素因数分解問題または離散対数問題が理想に近いものであると考えた．なぜなら，それらの問題は十分な研究の下での計算量的困難性が予想されており [17, 22, 23]，また，従来，実用的なスキームにおいて暗号学的仮定として用いられる数論問題の２つのカテゴリー（素因数分解問題系と離散対数問題系）において最も難しい問題であることが理由である（すなわち，できる限り弱い仮定の下で安全性を証明できることが理想）.

素因数分解問題系：素因数分解問題，RSA 問題，平方剰余問題，等,

離散対数問題系：離散対数問題，Diffie-Hellman 計算問題，Diffie-Hellman 決定問題，等.

HIME(R) では効率面を考慮して素因数分解問題の困難性と等価の安全性を持つように設計する方針とした．そこで，モジュラー平方関数（Rabin 暗号化関数）に着目した．$N = pq$（p, q は素数）を法とするモジュラー平方関数の逆関数を求めることは N の素因数分解問題の困難性と等価であることが知られている以外にも，暗号化処理スピードが速いという利点がある．しかし，

(P-1) モジュラー平方関数は一方向性置換ではない（すなわち，復号化が一意的に行われない）.

(P-2) Rabin 暗号は，選択暗号文攻撃に対して安全でない.

(P-3) 復号化処理スピードが速くない（RSA 暗号と同程度）.

の問題があった.

そこで, (P-1) と (P-2) の問題を解決するために, OAEP [3] を利用した (この点の詳細については, 第 2.3 節で述べる.). OAEP は落し戸付き一方向性置換から得られる公開鍵暗号を IND-CCA1 に変換する方式である (当初, IND-CCA2 に変換できると考えられていたが, 一般には IND-CCA1 であることが指摘された [32].). OAEP を利用することで,(ランダムオラクルモデル上で) 復号化一意性を確率的に保証することができ, また, Coppersmith のアルゴリズムを利用することにより, ランダムオラクルモデル上で IND-CCA2 であることが証明できた.(Rabin 暗号に OAEP を適用することにより, (P-1) と (P-2) を解決するアイデアは HIME-2 [19] において既に用いている. その後, OAEP とはパディングの方法が異なるが, Rabin-SAEP, Rabin-SAEP+ でも同様のことが行われている.) さらに, OAEP を利用することで, 上記設計方針の (2-2), (2-3) の条件もある程度クリアすることができる. 特に, (2-3) の平文空間を大きく取ることについては, 次の理由から重要であると考えた : 公開鍵暗号の主たる目的は共通鍵暗号のデータ暗号化鍵の配送である. しかし, 実システムでは, SET (Secure Electronic Transaction) のように, データ暗号化鍵だけではなく付加情報 (暗号の種類, ユーザの ID 情報, 等) を一緒に送ることがシステムが多数存在するためである.

(P-3) の問題を解決するために, 法とする合成数を $N = p^d q$ (p, q : 素数, $d > 1$) として, これに応じた新しい計算方法を用いた. 従来, このような合成数 N を法として高速な復号化を行う変形版 RSA 暗号が提案されている [33]. 従来方法では, Chinese Remainder Theorem (CRT) を用いて $\mathbb{Z}_N$ を $\mathbb{Z}_{p^d}$ と $\mathbb{Z}_q$ の直積に分解し, $\mathbb{Z}_{p^d}$ において高速計算方法を利用した後, $\mathbb{Z}_q$ 上の計算結果とあわせて再度 CRT により張り合わせを行っている. HIME(R) ではモジュラー積計算の個数を減らすことを目的に, CRT を用いることのない計算方法を開発した. これにより, 従来方式に比べて次のメリットがある.

- モジュラー積計算の個数が減少した (第 3.2 節で詳しく述べる).
- 新方式では, CRT を使わないため Euclid の互除法による計算部分が不要になる. これにより, 実測処理時間及び実装サイズの短縮が図れる.

この差は 1 回の復号化を通常のパソコンで実行した場合は殆ど無視できる差であるが, IC カード等の計算能力の低い媒体を使って計算する場合や一度の多くの復号化処理を必要とするシステムに適用した場合は無視できない差となって表れるものと予想される.

また, 上記 (2-4) の公開鍵暗号として共通鍵暗号とのハイブリッド方式*を避けた理由としては,

(a) IC カード等のメモリが限られた媒体に実装する場合を考慮して, プログラムのサイズを大きくしないため.

(b) 共通鍵暗号のデータ暗号化鍵の配送に利用する場合, 使用される共通鍵暗号アルゴリズムに依存させないため. 例えば, ハイブリッド方式では, 最悪の場合, 2 種類の共

*例えば, 素因数分解問題ベースの hybrid 方式としては, EPOC-2 [8], EPOC-3 [28] が挙げられる.

通鍵暗号アルゴリズムを用意する必要があり開発コストが問題になるケースが考えられる．

等が挙げられる．

以上のような方針で設計された HIME(R) は次の特徴を持つ．

(H-1) 合成数 $N = p^d q$ の素因数分解問題の困難性を仮定として，ランダムオラクルモデル上で IND-CCA2 の意味において安全であることが証明できる．

(H-2) 非常に高速な暗号化処理が可能（１回のモジュラー積のみ）．

(H-3) 復号化速度について，HIME(R) (1536 bits) は RSA-OAEP (1024 bits)[3] に対して，約 2.5 倍の高速処理が可能（モジュラー積の個数による比較）．

(H-4) RSA-OAEP と同等以上の十分大きな平文空間を持つ．

(H-5) 合成数 N のサイズを大きく選んでも，従来方式に比べ，暗号化・復号化の効率性を損なわない．

上記 (H-1) の安全性についての詳細は，第 2 章にて述べる．また，上記 (H-2)〜(H-4) の効率性についての詳細は，第 3 章にて述べる．上記 (H-5) については，第 2.6 節と第 3.1 節にて詳しく述べる．

2 HIME(R)の安全性

2.1 諸定義

合成数生成器 $\mathcal{G}$ は確率的多項式時間（PPT）アルゴリズムであり，パラメータ k を入力として，k ビットの合成数を出力する．すなわち，$N \leftarrow \mathcal{G}(1^k)$ (但し，$|N| = k$) .

定義 2.1. $\mathcal{G}$ を合成数生成器とする．

$$\Pr\left[N \leftarrow \mathcal{G}(1^k) \ : \ M(N) = (p_1, p_2, \ldots, p_d)\right] \geq \epsilon,$$

が成立するとき，アルゴリズム M は $\mathcal{G}(1^k)$ において (t, ϵ)-素因数分解可能と呼ぶ．但し，$N = \prod_{i=1}^{d} p_i$ (各 p_i は素数)，M は高々 t ステップで停止する．

無視できない ϵ に対して，$\mathcal{G}(1^k)$ において (t, ϵ)-素因数分解可能な多項式時間アルゴリズム M が存在しないとき，単に $\mathcal{G}$ の素因数分解は困難であると呼ぶ．

さらに，文献 [2] から次の定義を引用する．

定義 2.2 (IND-CPA, IND-CCA1, IND-CCA2 in the random oracle model). $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ を公開鍵暗号スキーム，$A = (A_1, A_2)$ を adversary とする．atk $\in \{$cpa, cca1, cca2$\}$，$k \in \mathbb{N}$ に対して，

$$\begin{aligned}\mathbf{Adv}^{\mathrm{ind-atk}}_{A,\Pi}(k) \ \stackrel{\mathrm{def}}{=} \ 2 \cdot \Pr\big[(pk, sk) \leftarrow \mathcal{K}(1^k) \, ; \quad H \leftarrow \mathbf{Hash} \, ; \quad (x_0, x_1, s) \leftarrow A_1^{H, \mathcal{O}_1}(pk) \, ; \\ b \leftarrow \{0,1\} \, ; \quad y \leftarrow \mathcal{E}^H_{pk}(x_b) \ : \quad A_2^{H, \mathcal{O}_2}(x_0, x_1, s, y) = b\big] - 1,\end{aligned}$$

とする．但し，

If atk=cpa	then	$\mathcal{O}_1(\cdot) = \varepsilon$	and	$\mathcal{O}_2(\cdot) = \varepsilon$
If atk=cca1	then	$\mathcal{O}_2(\cdot) = D_{sk}(\cdot)$	and	$\mathcal{O}_2(\cdot) = \varepsilon$
If atk=cca2	then	$\mathcal{O}_2(\cdot) = D_{sk}(\cdot)$	and	$\mathcal{O}_2(\cdot) = D_{sk}(\cdot)$.

上記において，A_1 は $|x_0| = |x_1|$ なる x_0, x_1 を出力し，s は adversary の情報を表わす．また，**Hash** は理想的ランダム関数からなる有限集合の族を表わす．

全ての多項式時間能力の A に対して $\mathbf{Adv}^{\mathrm{ind-atk}}_{A,\Pi}(\cdot)$ が無視できる値であるとき，Π はランダムオラクルモデル上で IND-ATK の意味において安全であると呼ぶ．

2.2 OAEP，OAEP+，SAEP，SAEP+

OAEP [3] は，落し戸付き一方向性置換 f から導かれる公開鍵暗号スキームを，f^{-1} を求めることの困難性を前提に，IND-CCA2 の意味において安全な方式に変換する一般的方法として知られてきた．しかし，最近になって，一般的な f については，IND-CCA1 の意味においてしか安全な方式に変換することができないことが指摘された [32]．但し，RSA-OAEP

については，IND-CCA2 の意味において安全になることが知られている [32, 14]．そして，OAEP の問題点を改良した OAEP+ [32] が提案された．但し，OAEP+ では OAEP に比べて，さらにもう 1 つ理想的ハッシュ関数が必要となる．

また，OAEP, OAEP+ の簡易版である SAEP, SAEP+ が提案され，IND-CCA2 の意味において安全な公開鍵暗号スキームとして Rabin-SAEP，Rabin-SAEP+，RSA-SAEP+ が同時に提案された [6]．

OAEP, OAEP+, SAEP, SAEP+ のパディングの方法は以下の通りである．

(1) OAEP, OAEP+:

$$\mathrm{OAEP}(m,r) = s||(r \oplus H(s)),$$
$$\mathrm{OAEP+}(m,r) = s'||(r \oplus H(s')).$$

但し，

$$s = (m||0^{k_1}) \oplus G(r),$$
$$s' = (m \oplus G(m||r))||H'(m||r),$$

H は $\{0,1\}^{n+k_1}$ から $\{0,1\}^{k_0}$ へのハッシュ関数，G は $\{0,1\}^{k_0}$ から $\{0,1\}^{n+k_1}$ へのハッシュ関数，H' は $\{0,1\}^{n+k_0}$ から $\{0,1\}^{k_1}$ へのハッシュ関数，$m \in \{0,1\}^n$，$r \in \{0,1\}^{k_0}$．

(2) SAEP, SAEP+:

$$\mathrm{SAEP}(m,r) = ((m||0^{s_0}) \oplus H(r))||r,$$
$$\mathrm{SAEP+}(m,r) = ((m||G(m||r)) \oplus H(r))||r.$$

但し，H は $\{0,1\}^{s_1}$ から $\{0,1\}^{n+s_0}$ へのハッシュ関数，G は $\{0,1\}^{n+s_1}$ から $\{0,1\}^{s_0}$ へのハッシュ関数，$m \in \{0,1\}^n$，$r \in \{0,1\}^{s_1}$．

2.3 HIME(R) と Rabin-SAEP, Rabin-SAEP+

HIME(R) では，パディングの方法として OAEP を採用している．HIME(R) はモジュラー平方関数をベースとしており，モジュラー平方関数 $f(x) = x^2 \bmod N$ (N は合成数) は落し戸付き一方向性置換ではないため，OAEP の適用対象外となる．しかし，OAEP を確率的な復号化一意性にも利用することにより，IND-CCA2 の意味において安全であることを証明できる．これは，SAEP，SAEP+ についても同様である．このアイデアは，HIME-2 [19] において，SAEP，SAEP+ が提案される前に利用されている（但し，HIME-2 発表以後に，V.Shoup によって OAEP の問題が指摘されたため，文献 [19] の安全性の主張は正しいが証明は間違いとなる．第 2.5 節で述べるように Coppersmith のアルゴリズムを利用した証明に訂正する必要がある．）．

HIME(R) において，OAEP を採用した理由は次の通りである．

(1) 実システムへの適用を考慮し，理想的ハッシュ関数に依存する安全性を軽減させる．

ランダムオラクルモデル上での安全性証明は理想的ランダム関数の存在が前提となっており，実システムでは理想的ランダム関数は SHA [26] のような実用的ハッシュ関数に置き換えられてしまうため，安全性証明はその効力を失ってしまう．よって，この点を考慮した．具体的には，SAEP および SAEP+ は OAEP および OAEP+ よりも，安全性をより理想的ハッシュ関数に依存していると考えられる．例えば，adversary は $f^{-1}(y)$ の最初の m_1 ビットと最後の m_2 ビットを計算できたとする．但し，$m_1 + m_2 < |N|/2$，y はターゲットとなる暗号文，f は Rabin-SAEP, Rabin-SAEP+ または RSA-SAEP+ の暗号化関数で N は法を意味する．このとき，$f^{-1}(y)$ の残りのビットを計算するのに，Coppersmith のアルゴリズムは適用できないことに注意する．さらに，SAEP および SAEP+ におけるハッシュ関数 H は（実世界では理想的ランダム関数ではないため）入力値の最後の m_2 ビットに応じて，出力値の最初の m_1 ビットに偏りがあったと仮定する．すなわち，x の最後の m_2 ビットを知っていれば，$H(x)$ の最初の m_1 ビットを $1/2^{m_1}$ よりも良い確率で求めることができる．このとき，明らかに，adversary は正しい b を $1/2$ よりも良い確率で推測することができる（cf. 定義 2.2）．

これに対して，OAEP や OAEP+ の場合，メッセージ文が 2 つのハッシュ関数 G，H によって，2 重に守られているため，SAEP や SAEP+ に比べて，上記のような問題は起きにくいと考える．

(2) 平文空間を大きく取る．

公開鍵暗号の主たる目的は，共通鍵暗号の鍵配送である．しかし，実際のシステムでは，鍵のみを配送するのではなく，ID 情報等の付加情報を一緒に送る場合が少なくない．HIME(R) では，このように鍵情報と付加情報を一緒に送信する場合や，複数の鍵情報を送信する場合を考慮して平文空間を大きく取ることを設計方針の 1 つとした．SAEP の場合，法 N が 1024 ビットのときメッセージ文の長さは 256 ビットまで，SAEP+ の場合は 384 ビットまでと，OAEP や OAEP+ と比較して平文空間が小さい（詳しくは，第 3.3 節を参照）．

一方，OAEP と OAEP+ を比較した場合，OAEP+ は OAEP に比べて別のハッシュ関数を容易しなければならないというデメリットがありが，特にメリットは見当たらないため，モジュラー平方関数に適用する場合は OAEP が最善と考えた．

2.4 Coppersmith のアルゴリズム

本節では，Coppersmith のアルゴリズム [10] について，結果のみを与える．

[Coppersmith] N を大きな合成数とし，N の素因数分解は判っていないものとする．$f(x)$ を，

$$f(x) = x^k + a_{k-1}x^{k-1} + \cdots + a_2x^2 + a_1x + a_0 \in \mathbb{Z}[x]$$

なる k 次のモニックな多項式とする．

このとき，

$$f(x_0) = 0 \pmod N \qquad \text{and} \qquad |x_0| < N^{1/k}.$$

なる全ての $x_0 \in \mathbb{Z}$ を求める多項式時間アルゴリズムが存在する．

以下では，k 次の多項式 $f \in \mathbb{Z}[x]$ の解を計算するとき，Coppersmith のアルゴリズムの計算時間を $T_C(N,k)$ によって表わす．

2.5 HIME(R) の安全性

HIME(R) の安全性について，次の定理が成立する．

定理 2.1. HIME(R) は，合成数 $N\,(=p^dq)$ の素因数分解問題の困難性の仮定として，ランダムオラクルモデル上で IND-CCA2 の意味において安全である．

以下，定理 2.1 を証明する．

$\mathcal{G}$ を $N\,(=p^dq) \leftarrow \mathcal{G}(1^k)$ なる合成数生成器とし，この N は HIME(R) における N と同じ確率分布を持つものとする．

定理 2.1 は，次の定理から直ちに導かれる．

定理 2.2. $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ を HIME(R) の公開鍵暗号スキームとする．但し，k_0，k_1 を付随パラメータ，n を平文長パラメータとする．このとき，各 k に対して，次のようなオラクルマシン U が存在する．アルゴリズム $A = (A_1, A_2)$ は，IND-CCA2 の意味において $\Pi(1^k)$ を $(t, q_D, q_G, q_H, \epsilon)$-解読可能と仮定する．すなわち，

$$2 \cdot \Pr[(pk, sk) \leftarrow \mathcal{K}(1^k)\,;\, G, H \leftarrow \mathbf{Hash}\,;\, (x_0, x_1, c) \leftarrow A_1^{\mathcal{D}_{sk}, G, H}(pk)\,;$$
$$b \leftarrow \{0,1\}\,;\, y \leftarrow \mathcal{E}_{pk}^{G,H}(x_b)\ :\ A_2^{\mathcal{D}_{sk}, G, H}(y, x_0, x_1, c) = b] - 1 \geq \epsilon.$$

ここで，A は高々 t ステップ以内に停止し，復号化オラクルに対して高々 q_D 回のアクセスを行い，ハッシュ関数 G に対して高々 q_G 回のアクセスを行い，ハッシュ関数 H に対して高々 q_H 回のアクセスを行う．

このとき，$\mathcal{G}(1^k)$ において (t', ϵ')-素因数分解可能なアルゴリズム $M = U^A$ が存在する．但し，

$$t' = t + q_H\, T_C(N, 2) + q_G q_H T_S(k) + \tilde{T}(k) + \mathcal{O}(k)$$
$$\epsilon' = \frac{1}{3}\left(\epsilon - \frac{q_G}{2^{k_0}}\right)\left(1 - \frac{q_G}{2^{k_0}}\right)\left(1 - \frac{q_D}{2^{k_1-1}} - \frac{q_D}{2^{k_0}}\right).$$

ここで，$T_S(k)$ は暗号化関数 $\mathcal{E}_{pk}(\cdot)$ の計算時間（ステップ数）を表わし，$\tilde{T}(k)$ は N と共通素因数を持つ整数が与えられたときに N の素因数分解を行う計算時間（ステップ数）を表わす．

Proof. 最初に，N の素因数分解を行うためのアルゴリズム M の構成方法について述べる．M は，合成数 $N\,(=p^dq)$ を入力値として，N の素因数を出力値とする．

(0) M に対して合成数 N を入力する．但し，$N \leftarrow \mathcal{G}(1^k)$.

(1) M は $w \in \{0,1\}^{k-1}$ をランダムに選び，$y = w^2 \bmod N$ を計算する．

(2) M は，G-list と H-list と呼ばれる 2 つのリストが空になるように初期化する．M は，$b \in \{0,1\}$ をランダムに選ぶ．

次に示すように，M は $A = (A_1, A_2)$ の 2 つのステージをシミュレートする．

(3) (**find**-stage のシミュレーション) M は，pk を A_1 に入力し，A_1 を起動する．但し，pk は HIME(R) における公開鍵．M は A に乱数を与えることで，次のように A のランダムオラクル G と H をシミュレートする．

(3.1) A_1 がオラクル H への質問 $h \in \{0,1\}^{n+k_1}$ を行ったとき，M は Coppersmith のアルゴリズムを用いて $(x + 2^{k_1}h)^2 \equiv y \pmod N$ なる $x \in \{0,1\}^{k_0}$ を（もし，そのような x が存在すれば）計算する．そして，M は $w^* = h||x$ として，h を H-list に加える．そのような x が存在しなければ，M は A_1 に乱数 $H_h \in \{0,1\}^{k_0}$ を与えて，h を H-list に加える．

(3.2) A_1 がオラクル G への質問 $g \in \{0,1\}^{k_0}$ を行ったとき，M は A_1 に対して乱数 $G_g \in \{0,1\}^{n+k_1}$ を与えて，g を G-list に加える．

(x_0, x_1, c) を A_1 の出力値とする．

(4) (**guess**-stage のシミュレーション) M は (y, x_0, x_1, c) を A_2 に入力し，A_2 を起動する．M は A_2 のランダムオラクルへの質問に対して次のように対応する．

(4.1) A_2 がオラクル H への質問 $h \in \{0,1\}^{n+k_1}$ を行ったとき，M は Coppersmith のアルゴリズムを用いて $(x + 2^{k_1}h)^2 \equiv y \pmod N$ なる $x \in \{0,1\}^{k_0}$ を（もし，そのような x が存在すれば）計算する．そして，M は $w^* = h||x$ として，h を H-list に加える．そのような x が存在しなければ，M は A_2 に乱数 $H_h \in \{0,1\}^{k_0}$ を与えて，h を H-list に加える．

(4.2) A_2 がオラクル G への質問 $g \in \{0,1\}^{k_0}$ を行ったとき，M は A_2 に対して乱数 $G_g \in \{0,1\}^{n+k_1}$ を与えて，g を G-list に加える．

(5) (復号化オラクルのシミュレーション) A が復号化オラクルに対して，質問 y' を行ったとする．ここで，(s_i, H_i)（$i = 1, 2, \ldots, q_H$）をオラクル H への質問 s_i とその答え H_i の組とし，(r_i, G_i)（$i = 1, 2, \ldots, q_G$）をオラクル G への質問 r_i とその答え G_i の組とする．このとき，$i = 1, 2, \ldots, q_H$ および $j = 1, 2, \ldots, q_G$ について，M は，

(5.1) $t_{i,j} = H_i \oplus r_j$ とする．

(5.2) $x'_{i,j}||z'_{i,j} = s_i \oplus G_j$ なる $x'_{i,j} \in \{0,1\}^n$ と $z'_{i,j} \in \{0,1\}^{k_1}$ を求める．

(5.3)

$$\begin{cases} x'_{i,j} & \text{if there is an } i,j \text{ such that } z'_{i,j} = 0^{k_1} \text{ and } y' = (s_i||t_{i,j})^2 \bmod N, \\ * & \text{otherwise.} \end{cases}$$

を出力する．

(6) 上記シミュレーションにおいて，w^* が定義されれば，M は w^* を出力し，シミュレーションを停止する．そうでない場合は **fail** を出力する．

(7) M は $\alpha = \gcd(w - w^*, N)$ および $l = (d+1)|\alpha|/k$ を計算し，$0 < \alpha < N$ であれば，

$$(\sqrt[l]{\alpha}, \quad N/\sqrt[l]{\alpha^d}),$$

または，

$$(\sqrt[d-l+1]{N/\alpha}, \quad N/\sqrt[d-l+1]{(N/\alpha)^d})$$

を出力し，そうでなければ **fail** を出力する．

（Note）
（１）H-list と G-list は，**find**-stage と **guess**-stage の両方の質問を含んでいる．
（２）M は w^* が定義された場合は A に対して質問の答えを返さない．
（３）ステップ (3.1) および (3.2) において，$2k_0 < k$ であることから Coppersmith のアルゴリズムが有効である．

次に，確率空間についての考察を行う．上記 M の振る舞いを “Game 1” と呼び，Game 1 における確率を $\Pr_1[\cdot]$ によって表わす．
Game 1 を実行するために必要な計算時間（ステップ数）は，

$$t' = t + q_H\, T_C(N, 2) + q_G q_H T_S(k) + \tilde{T}(k) + \mathcal{O}(k)$$

であることは容易に確認できる．また，M の計算を U^A によって行うマシン U が存在することも明らかである．

実際の復号化オラクルと M のシミュレータの違いを見るために，次のイベントを定義する．

$$\textbf{FAIL} \text{ は true} \overset{\text{def}}{\Longleftrightarrow} \text{“シミュレータの出力値} \neq \mathcal{D}_{sk}^{G,H}(y')\text{”},$$

但し，sk は HIME(R) の秘密鍵．
このとき，次の補題を得る．

補題 2.1. 実際の復号化オラクルの出力と M のシミュレータの出力が異なる確率は，

$$\Pr_1[\textbf{FAIL}] \leq \frac{q_D}{2^{k_1-1}} + \frac{q_D}{2^{k_0}}$$

となる．

Proof. w^* が定義されると，M はシミュレーションを停止する．これより，M は w^* が定義されるまで，復号化オラクルをシミュレートする点に注意して証明を進める．

y をターゲットとなる暗号文，y' を復号化オラクルへの質問である暗号文とする．さらに，$s = x_b 0^{k_1} \oplus G(r), t = r \oplus H(s), (s||t)^2 \equiv y \pmod N$, $s' = x'0^{k_1} \oplus G(r'), t' = r' \oplus H(s')$ and $(s'||t')^2 \equiv y' \pmod N$ とする．但し，$x' \in \{0,1\}^n$ and $r, r' \in \{0,1\}^{k_0}$.

ここで，次のイベントを考える．

AskR' は true $\overset{\text{def}}{\Longleftrightarrow}$ r' は G-list に含まれる．

AskS' は true $\overset{\text{def}}{\Longleftrightarrow}$ s' は H-list に含まれる．

W'=**AskR'** $\wedge$ **AskS'**.

このとき，$\Pr_1[\textbf{FAIL} \mid \textbf{W'}] = 0$ であり，

$$\begin{aligned}\Pr_1[\textbf{FAIL}] &= \Pr_1[\textbf{FAIL} \mid \textbf{W'}] \cdot \Pr_1[\textbf{W'}] + \Pr_1[\textbf{FAIL} \mid \neg\textbf{AskS'}] \cdot \Pr_1[\neg\textbf{AskS'}] \\ &\quad + \Pr_1[\textbf{FAIL} \mid \textbf{AskS'} \wedge \neg\textbf{AskR'}] \cdot \Pr_1[\textbf{AskS'} \wedge \neg\textbf{AskR'}] \\ &\leq \Pr_1[\textbf{FAIL} \mid \neg\textbf{AskS'}] + \Pr_1[\textbf{FAIL} \mid \textbf{AskS'} \wedge \neg\textbf{AskR'}] \qquad (1)\end{aligned}$$

が成立する．

また，$s \neq s'$ であることより，

$$\Pr_1[\textbf{FAIL} \mid \neg\textbf{AskS'}] \leq \frac{q_D}{2^{k_1}} \qquad (2)$$

となる．

次に，$\Pr_1[\textbf{FAIL} \mid \textbf{AskS'} \wedge \neg\textbf{AskR'}]$ について考える．

$$\begin{aligned}\Pr_1[\textbf{FAIL} \mid \textbf{AskS'} \wedge \neg\textbf{AskR'}] &= \Pr_1[\textbf{FAIL} \mid \textbf{AskS'} \wedge \neg\textbf{AskR'} \mid r \neq r'] \cdot \Pr_1[r \neq r'] + \\ &\quad \Pr_1[\textbf{FAIL} \mid \textbf{AskS'} \wedge \neg\textbf{AskR'} \mid r = r'] \cdot \Pr_1[r = r'] \\ &\leq \Pr_1[\textbf{FAIL} \mid \textbf{AskS'} \wedge \neg\textbf{AskR'} \mid r \neq r'] + \Pr_1[r = r']\end{aligned}$$

が成立し，さらに，

$$\Pr_1[\textbf{FAIL} \mid \textbf{AskS'} \wedge \neg\textbf{AskR'} \mid r \neq r'] \leq q_D/2^{k_1}$$

が成立する．

また，$r = r'$ であるためには，$t' = t \oplus H(s) \oplus H(s')$ である必要があるが，オラクル H への質問 s は行われていないため，

$$\Pr_1[r = r'] \leq q_D/2^{k_0} \qquad (3)$$

となる．

よって，式 (1), (2), (3) から，

$$\Pr_1[\textbf{FAIL}] \leq \frac{q_D}{2^{k_1-1}} + \frac{q_D}{2^{k_0}}.$$

を得る． □

直感的には，補題 2.1 は adversary が復号化オラクルから新たな情報を引き出すことは難しいことを主張している．

Game 1 において，イベント **FAIL** が true でない場合を “Game 2” と呼び，このとき，$\Pr_2[\cdot] = \Pr_1[\cdot \mid \neg\mathbf{FAIL}]$ と定義する．

$s = x_b 0^{k_1} \oplus G(r)$, $t = r \oplus H(s)$ and $y = (s||t)^2 \bmod N$ として，次のイベントを考える．

BAD は true $\overset{\text{def}}{\Longleftrightarrow}$

- ランダムオラクル G への質問 r は **find**-stage または **guess**-stage で行われている．
- $G_r \neq s \oplus x_b 0^{k_1}$

G = ¬**BAD**

もし，ある $x \in \{0,1\}^{k_0}$ に対して $(h||x)^2 \equiv y \pmod N$ となる質問 h $(\in \{0,1\}^{n+k_1})$ を A がランダムオラクル H に対して行えば，直ちに M はシミュレーションを停止する．よって，G への質問 r は，そのような h が H-list にない場合に行われることに注意する．

Game 2 において，イベント **G** が true である場合を “Game 3” と呼び，このとき，$\Pr_3[\cdot] = \Pr_2[\,\cdot\,|\mathbf{G}]$ と定義する．

次に，A のアドバンテージについて考察を行う．

$N \leftarrow \mathcal{G}(1^k)$，$\mathcal{E}_*$ を HIME(R) の暗号化関数とする．このとき，

$$G_*, H_* \leftarrow \mathbf{Hash};\quad (x_0^*, x_1^*, c^*) \leftarrow A_1^{G_*, H_*, \mathcal{D}_*}(N, k_0, k_1);\quad b_* \leftarrow \{0,1\};\quad y^* \leftarrow \mathcal{E}_*^{G_*, H_*}(x_b^*),$$

として，$A_2^{G_*, H_*, \mathcal{D}_*}(y^*, x_0^*, x_1^*, c^*)$ を起動させる．但し，$\mathcal{D}_*$ は HIME(R) の復号化関数．これを Game 1^* と呼び，このゲームにおける確率を $\Pr_1^*[\cdot]$ で表わす．

さらに，Game 1^* と少し異なった Game 2^* を次のように定義する．

(1) $N \leftarrow \mathcal{G}(1^k)$，$\mathcal{E}_*$ を HIME(R) の暗号化関数とする．

(2) y^* を暗号化関数の像集合からランダムに選び，$H_* \leftarrow \mathbf{Hash}$，$b_* \leftarrow \{0,1\}$ とする．

(3) $(s^*||t^*)^2 \equiv y^* \pmod N$ なる $s^* \in \{0,1\}^{n+k_1}$，$t^* \in \{0,1\}^{k_0}$ をランダムに選び，$r^* = t^* \oplus H(s^*)$ とする．

(4) オラクル G_* を次のように定義する．

$$G_*(x^*) = \begin{cases} s^* \oplus x_b^* 0^{k_1} & \text{if } x^* = r^*, \\ g_* \in_R \{0,1\}^{n+k_1} & \text{if } x^* \neq r^*. \end{cases}$$

(5) $(x_0^*, x_1^*, c^*) \leftarrow A_1^{G_*, H_*, \mathcal{D}_*}(N, k_0, k_1)$ に対して，$A_2^{G_*, H_*, \mathcal{D}_*}(y^*, x_0^*, x_1^*, c^*)$ を起動する．

Game 3 と Game 2^* について，ある $x \in \{0,1\}^{k_0}$ について $(h||x)^2 \equiv y \pmod N$ となる H への質問 h が行われるまでについて見れば，A にとって同一のゲームである．

Game 1 において，次のイベントを定義する．

AskH は true $\overset{\text{def}}{\iff}$ **guess**-stage が終わった時点で，ある $x \in \{0,1\}^{k_0}$ について $(h||x)^2 \equiv y \pmod N$ となる h $(\in \{0,1\}^{n+k_1})$ が，H-list にある．

AskR は true $\overset{\text{def}}{\iff}$ **guess**-stage が終わった時点で，r は G-list にある．

AskS は true $\overset{\text{def}}{\iff}$ **guess**-stage が終わった時点で，s は H-list にある．

$\mathbf{W} = \mathbf{AskR} \wedge \mathbf{AskS}$.

補題 2.2. Game 2 において，イベント **G** が true でない確率 $\Pr_2[\neg\mathbf{G}]$ について，

$$\Pr_2[\neg\mathbf{G}] \ \leq \ \frac{q_G}{2^{k_0}}.$$

が成立する．

Proof.

$$\Pr_2[\neg\mathbf{G}] = \ \Pr_2[\neg\mathbf{G} \mid \neg\mathbf{AskH}] \leq \ \Pr_2[\neg\mathbf{G} \mid \neg\mathbf{AskS}] \leq \Pr_2[\mathbf{AskR} \mid \neg\mathbf{AskS}] \leq \frac{q_G}{2^{k_0}}.$$

□

次の補題は，$\Pr_3[A=b]$ と $\Pr_3[\mathbf{W}]$ の関係を示す．

補題 2.3. $\Pr_3[A=b]$ と $\Pr_3[\mathbf{W}]$ について，次の関係が成立する．

$$\Pr_3[\mathbf{W}] \geq 2\Pr_3[A=b] - 1 - \frac{q_G}{2^{k_0}}.$$

但し，“$A=b$” は A は正しいビット b を出力することを意味する．

Proof. まず，

$$\begin{aligned}
\Pr_3[A=b] &= \Pr_3[A=b \mid \mathbf{W}] \cdot \Pr_3[\mathbf{W}] + \Pr_3[A=b \mid \neg\mathbf{AskR}] \cdot \Pr_3[\neg\mathbf{AskR}] \\
&\quad + \Pr_3[A=b \mid \mathbf{AskR} \wedge \neg\mathbf{AskS}] \cdot \Pr_3[\mathbf{AskR} \wedge \neg\mathbf{AskS}] \\
&\leq \Pr_3[\mathbf{W}] + \Pr_3[A=b \mid \neg\mathbf{AskR}] \cdot \Pr_3[\neg\mathbf{AskR}] + \Pr_3[\mathbf{AskR} \wedge \neg\mathbf{AskS}] \\
&= \Pr_3[\mathbf{W}] + \Pr_3[A=b \mid \neg\mathbf{AskR}] \cdot (1 - \Pr_3[\mathbf{W}] - \Pr_3[\mathbf{AskR} \wedge \neg\mathbf{AskS}]) \\
&\quad + \Pr_3[\mathbf{AskR} \wedge \neg\mathbf{AskS}] \qquad (4)
\end{aligned}$$

が成立する．

$\neg\mathbf{AskR}$ が true であるとき，A は正しいビット b をアドバンテージを持っては推測できない．よって，

$$\Pr_3[A=b \mid \neg\mathbf{AskR}] = \frac{1}{2}. \qquad (5)$$

また，

$$\Pr_3[\mathbf{AskR} \wedge \neg\mathbf{AskS}] \leq \frac{q_G}{2^{k_0}}. \tag{6}$$

である．

よって，式 (4), (5), (6) から，

$$\Pr_3[\mathbf{W}] \geq 2\Pr_3[A = b] - 1 - \frac{q_G}{2^{k_0}},$$

を得る． □

補題 2.3 から，

$$\Pr_2[\mathbf{W}] \ \geq \ \epsilon - \frac{q_G}{2^{k_0}}. \tag{7}$$

を得る．

補題 2.1, 補題 2.2 と式 (7) から，

$$\begin{aligned}
\Pr_1[\mathbf{AskH}] &\geq \Pr_1[\mathbf{AskS}] \geq \Pr_2[\mathbf{AskS}] \cdot \Pr_1[\mathbf{FAIL}] \\
&\geq \Pr_2[\mathbf{AskS} \mid \mathbf{G}] \cdot \Pr_2[\mathbf{G}] \cdot \Pr_1[\mathbf{FAIL}] \\
&\geq \Pr_3[\mathbf{AskS}] \cdot \Pr_2[\mathbf{G}] \cdot \Pr_1[\mathbf{FAIL}] \\
&\geq \Pr_3[\mathbf{W}] \cdot \Pr_2[\mathbf{G}] \cdot \Pr_1[\mathbf{FAIL}] \\
&\geq \left(\epsilon - \frac{q_G}{2^{k_0}}\right)\left(1 - \frac{q_G}{2^{k_0}}\right)\left(1 - \frac{q_D}{2^{k_1-1}} - \frac{q_D}{2^{k_0}}\right)
\end{aligned}$$

を得る．

仕様書において示したように，2 次方程式 $X^2 \equiv y \pmod N$ は $\mathbb{Z}_N$ において高々 4 つの解を持ち，それらの 2 つは $N/2$ 以下である．$w, w^* \in \{0,1\}^{k-1}$ および $N/2 < 2^{k-1}$ であることから，$\alpha = \gcd(w - w^*, N)$ に対して $1 < \alpha < N$ となる確率は 1/3 以上となる．

以上のことから，

$$\Pr_1[N \leftarrow \mathcal{G}(1^k) \ : \ \ M(N) = (p,q)] \geq \frac{1}{3}\left(\epsilon - \frac{q_G}{2^{k_0}}\right)\left(1 - \frac{q_G}{2^{k_0}}\right)\left(1 - \frac{q_D}{2^{k_1-1}} - \frac{q_D}{2^{k_0}}\right)$$

が成立する． □

2.6 素因数分解問題について

HIME(R) では, (p, q, d) は $N = p^d q$ の素因数分解が困難であるように選ぶ必要がある．d が大きい ($d \approx \sqrt{\log p}$) 場合には $N = p^d q$ を分解する効率的アルゴリズムが知られている [7] が, 小さい場合には n の分解は困難であろうと思われている．

現在知られている素因数分解アルゴリズムは主に二つの方法に分けられる, すなわち合成数依存型と素因子依存型である．

素因子依存型としては, ρ-法, $p-1$-法, $p+1$-法, 楕円曲線法 ([23], [29]) などが知られている. 一般数体ふるい法は合成数依存型である.

1024-bit RSA-型の合成数と同等強度を持たせる場合, HIME(R) は, $N = p^2q$ 型のとき, 1300∼1400-bit の合成数 (素因子は 430∼460-bit) を, また, $N = p^3q$ の場合には 1500∼1600-bit の合成数を用いる. これらの合成数に対しては, ρ-法は効果的でなく, また HIME(R) の鍵生成の際, 適切な素因子 ($p-1$ と $p+1$ が大きな素因子を持つなど) を選べば $p-1, p+1$-法はやはり効果的でない.

N の素因子 p を見つける際の計算量は, 楕円曲線法において $L_p[1/2, \sqrt{2}]$ ($L_p[a,b] = \exp((b+o(1))(\log p)^a(\log\log p)^{1-a})$), であり, 数体ふるい法では $L_N[1/3, 1.901]$ ([9]), ともに準指数オーダーである. 実際には分解法の実装や計算機の能力に依存しており, 楕円曲線法により分解が成功しているのは素因子が 180 ∼ 190-bit 程度の場合であり, 数体ふるい法では合成数が 512-bit 程度の場合である.

以下ではより正確に計算量を評価してみる. $t_{\mathrm{EC}}(p) = \log(L_p(1/2, \sqrt{(}2)))$ を楕円曲線法による計算量の対数値とし, $t_{\mathrm{NFS}}(N) = \log(L_N(1/3, 1.901))$ を数体ふるい法によるものとする.

このとき 1024-bit RSA-型合成数 $n = pq$ (p, q : 512 bits) に対しては数体ふるい法の方がより効果的で,

$$\alpha := t_{\mathrm{NFS}}(\text{1024-bit } N) = C_{\mathrm{NFS}} + 59.42,$$

である. ただし C_{NFS} は O 部分に関する定数である.

一方, HIME(R) に対し 1344-bit 合成数 $N = p^2q$ (p, q : 448 bits) を用いれば, 楕円曲線法がより効果的で,

$$\beta(448) := \alpha - t_{\mathrm{EC}}(\text{448-bit } p) = C - 0.28,$$

を得る. ただし, C は O 部分からくるある定数である. (β のグラフを図 1 に示す.)

これにより, 1024-bit RSA-型合成数の分解は, 1344-bit p^2q-型合成数の分解に比べ, $e^{0.28} = 1.32$ 倍早いことになる. 従って, 1344-bit HIME(R) のモジュラスは 1024-bit RSA のモジュラスと同等強度を持つということができる.

$N = p^3q$ 型の場合, HIME(R) は 1500∼1600 bit の N (素因子は 375∼400 bit) を用いる. 例えば, N のビット長が 1536 のとき,

$$\beta(448) = C + 4.9.$$

となる. このとき $e^{4.9} = 134.3$ であり, やはりこのタイプの合成数も 1024-bit RSA-型合成数と同等強度ということができる.

さらに, HIME(R)-型合成数と 2048-bit, および 4096-bit の RSA-型合成数を比較する. 2048-bit RSA-型合成数と同等強度が必要ならば, 2304-bit p^2q-型合成数または 3072-bit p^3q-型合成数を用い, 4096-bit RSA-型合成数に対しては, 4032-bit p^2q-型合成数, または 4928-bit p^3q-型合成数を用いる (図 2). p^2q-型合成数では, ビット長が 2700 を超えると楕円曲線法よ

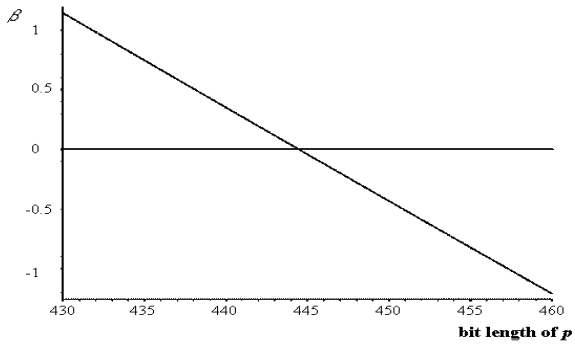

図 1: The graph of β

り数体ふるい法の方が効果的となる. 従って HIME(R) のモジュラスとして 4096-bit p^2q-型合成数を用いることができる. しかし, 上では p と q が同じサイズとなるよう 4032-bit 合成数を選択した.

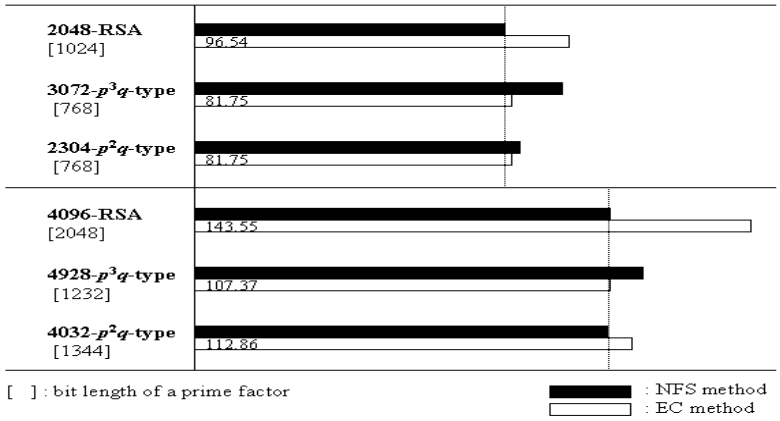

図 2: Complexity for long modulus

2.7 Manger の選択暗号文攻撃について

最近，Manger によって，インテグリティチェックを利用した PKCS #1 v2.0 に対する攻撃が発表されている [24]. HIME(R) についても，実装を行う上では，この攻撃方法を考慮して文献 [24] に記載の対策を施す必要がある．具体的には，HIME(R) の復号化手順にお

いて，与えられた暗号文の平方根のビット長が正しい（暗号文の）ビット長であるか否かを第三者に判らないような設計を行う．本件については，HIME(R) 固有の問題ではなく，実装設計における根本的問題でもあるので，本ドキュメントにおいてはその詳細については省略する．

3 性能評価

3.1 鍵長について

HIME(R) におけるパラメータ k_0, k_1 および k について, まず $|k_0|, |k_1| \geq 128$ を推奨する.

次に，表 1 に RSA-OAEP [3], RSA-OAEP+ [32], Rabin-SAEP [6], Rabin-SAEP+[6], EPOC-1,2,3 [8, 28] と HIME(R) の各モジュラス長, すなわち $|k|$, の比較をまとめる. 各モジュラス長は数体ふるい法と楕円曲線法を用いて素因数分解を行った場合に同等の困難さを持つように決定している (*cf.* 2.6 節).

表 1: The length of modulus

	Modulus length (bits)		
RSA(-OAEP, OAEP+)	1024	2048	4096
Rabin(-SAEP, -SAEP+)	1024	2048	4096
EPOC(-1, -2, -3)	1344	2304	4032
HIME(R) ($N = p^2q$)	1344	2304	4032
HIME(R) ($N = p^3q$)	1536	3072	4928

表 2 に各方式における公開鍵, 秘密鍵の長さをまとめる. 但し，簡単のため, ハッシュ関数及びビット長を表わすパラメータについては省略した.

表 2: Public and secret key length (bits)

	Modulus length	Public key	Secret key
RSA(-OAEP, OAEP+)	1024	1026〜2048	2048
Rabin(-SAEP, SAEP+)	1024	1024	1024
EPOC(-1, -2, -3)	1344	4032	1344
HIME(R) ($N = p^2q$)	1344	1344	896
HIME(R) ($N = p^3q$)	1536	1536	728
RSA(-OAEP, OAEP+)	2048	2050〜4096	4096
Rabin(-SAEP, SAEP+)	2048	2048	2048
EPOC(-1, -2, -3)	2304	6912	2304
HIME(R) ($N = p^2q$)	2304	2304	1536
HIME(R) ($N = p^3q$)	3072	3072	1536
RSA(-OAEP, OAEP+)	4096	4098〜8192	8192
Rabin(-SAEP, SAEP+)	4096	4096	4096
EPOC(-1, -2, -3)	4032	12096	4032
HIME(R) ($N = p^2q$)	4032	4032	2688
HIME(R) ($N = p^3q$)	4928	4928	2464

表 2 により, HIME(R) の鍵長は他の方式と比較して同等またはより短いことがわかる.

3.2 暗復号化処理における剰余乗算回数

各暗号方式 RSA-OAEP, RSA-OAEP+, Rabin-SAEP, Rabin-SAEP+, EPOC-1,2,3, HIME(R) に対し, 暗号化, 復号化速度の効率性の目安として，比較的計算負担の大きい剰余乗算回数を表 3 にまとめた. RSA-OAEP, RSA-OAEP+ においては, 復号化で中国人剰余定理を適応している. また, 公平を期すために全ての暗号方式で用いる乱数は 128 bit に設定した.

表 3: Efficiency by the (converted) number of modular multiplications.

	Modulus length	Encryption	Decryption
RSA(-OAEP, -OAEP+)	1024 (bits)	$2 \sim 1536$	388
Rabin(-SAEP, -SAEP+)	1024 (bits)	1	388
EPOC-1	1344 (bits)	$386 \sim 1351$	$1415 \sim 2380$
EPOC-2	1344 (bits)	386	1415
EPOC-3	1344 (bits)	386	1029
HIME(R) ($N = p^2q$)	1344 (bits)	2	260
HIME(R) ($N = p^3q$)	1536 (bits)	3	168
RSA(-OAEP, -OAEP+)	2048 (bits)	$8 \sim 12288$	3088
Rabin(-SAEP, -SAEP+)	2048 (bits)	4	3088
EPOC-1	2304 (bits)	$1134 \sim 6804$	$6319 \sim 11989$
EPOC-2	2304 (bits)	1134	6319
EPOC-3	2304 (bits)	1134	5185
HIME(R) ($N = p^2q$)	2304 (bits)	6	1305
HIME(R) ($N = p^3q$)	3072 (bits)	9	1320
RSA(-OAEP, -OAEP+)	4096 (bits)	$32 \sim 98304$	24640
Rabin(-SAEP, -SAEP+)	4096 (bits)	16	24640
EPOC-1	4032 (bits)	$2977 \sim 31256$	$30762 \sim 59041$
EPOC-2	4032 (bits)	3473	30762
EPOC-3	4032 (bits)	3473	27785
HIME(R) ($N = p^2q$)	4032 (bits)	16	6974
HIME(R) ($N = p^3q$)	4928 (bits)	23	5411

ここで, べき乗剰余 a^x ($x:$ k bit) に対し, 通常のバイナリ法を用いるとするとき, $3k/2$ 回の剰余乗算が必要と仮定する. また, a^xb^y ($x, y:$ k bit) には拡張バイナリ法 [20] により $7k/4$ 回の剰余乗算が必要と仮定する.

1024 bit の法での剰余乗算を基準とする. 一般に n-bit の法での剰余乗算 1 回は $(n/1024)^2$ のコストが必要であると仮定する. 各方式における bit 長の選択については 2.6 節参照のこと.

また, 図3に, RSA (RSA-OAEP, RSA-OAEP+, Rabin-SAEP, Rabin-SAEP+) と HIME(R)

について, さらに大きい法での評価をまとめた.

これらの結果から, HIME(R) はモジュラスが大きくなるにつれて，他方式との効率性の差が大きくなることがわかる.

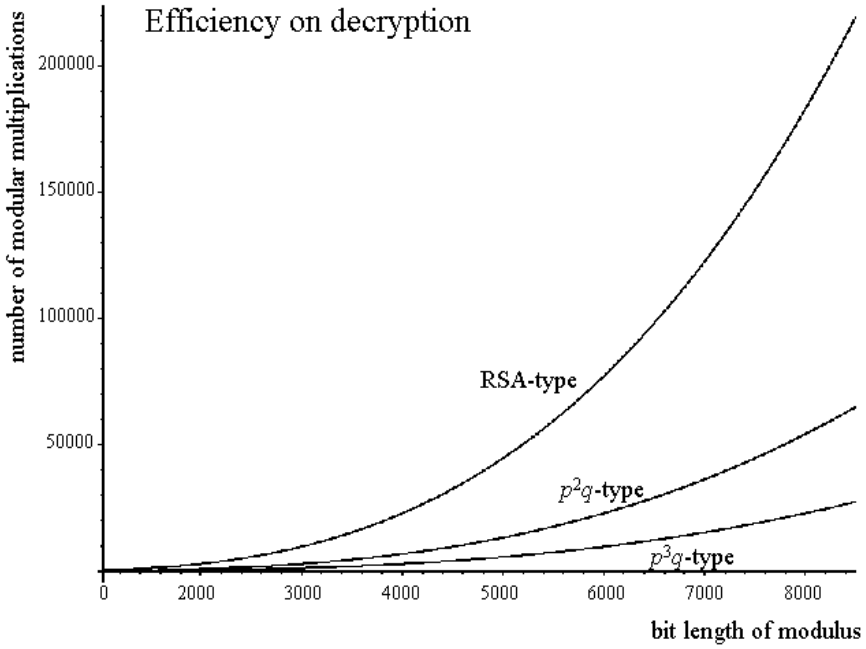

図 3: Efficiency on decryption

ここで，第 1 章で触れた Takagi の計算方法 [33] と HIME(R) における復号化計算方法の比較を簡単に行っておく．HIME(R) の計算方法による剰余乗算の回数を T_1，Takagi の計算方法による剰余乗算の回数を T_2 とすると，$T_1 = |p|/3 + 11/6$，$T_2 = |p|/3 + 25/6$ が成立し，若干であるが HIME(R) の方が計算量は少ない．また，HIME(R) の計算方式では，中国人の剰余定理を使わないため Euclid の互除法による計算部分が不要になる．これにより，実測処理時間及び実装サイズの短縮が図れると考える．この処理時間の差は，通常のパソコンで 1 回の処理をした場合では殆ど無視できる程度と思われるが，IC カードのような計算能力の低い媒体で計算する場合や同時に多くの計算をする必要のあるシステム等では無視できないものになると考える．例えば，平均して 1 度に 600 回の復号化処理を行うシステムでは，この差は 1400 回の乗算剰余の差になって現れる.

3.3 平文空間と暗号文空間

表 4 に各方式における平文長と暗号文長をまとめる．ここで, 各方式のモジュラス長は, 表 1 に従って設定されている．また, 各方式において，乱数およびチェックビットの長さは

128 bits とした.

表 4: Plaintext and ciphertext lengths (bits)

	Modular length	Plaintext length	Ciphertext length
RSA(-OAEP, -OAEP+)	1024	768	1024
Rabin-SAEP	1024	256	1024
Rabin-SAEP+	1024	384	1024
EPOC-1	1344	320	1344
EPOC(-2, -3)	1344	Arbitrary	1344+α
HIME(R) ($N = p^2q$)	1344	1088	1344
HIME(R) ($N = p^3q$)	1536	1280	1536
RSA(-OAEP, -OAEP+)	2048	1792	2048
Rabin-SAEP	2048	512	2048
Rabin-SAEP+	2048	896	2048
EPOC-1	2304	640	2304
EPOC(-2, -3)	2304	Arbitrary	2304+α
HIME(R) ($N = p^2q$)	2304	2048	2304
HIME(R) ($N = p^3q$)	3072	2816	3072
RSA(-OAEP, -OAEP+)	4096	3840	4096
Rabin-SAEP	4096	1024	4096
Rabin-SAEP+	4096	1920	4096
EPOC-1	4032	1216	4032
EPOC(-2, -3)	4032	Arbitrary	4032+α
HIME(R) ($N = p^2q$)	4032	3776	4032
HIME(R) ($N = p^3q$)	4928	4672	4928

EPOC-2,3 は秘密鍵暗号とのハイブリッド方式であるため，平文長は任意であり，α は秘密鍵暗号による暗号文の長さを意味する.

公開鍵暗号方式の主な用途は共通鍵暗号方式におけるデータ暗号化鍵を配送することである. しかしながら, 知る限りにおいては, 単にデータ暗号化鍵のみを送るプロトコルは稀であり, 多くのプロトコルでは, ユーザのＩＤ情報など, 付加情報を同時に送ることを要求している. 従ってこの要求に応える公開鍵暗号方式を選択することが重要であり, 平文長の比較は公開鍵暗号方式の使用目的を明確にするためにも重要である.

HIME(R) の平文空間は大きく, データ暗号化鍵と付加情報を同時に配送するのに十分である.

3.4 ソフトウェア実装評価

本節では HIME(R) の C 言語での実装結果を述べる. 評価は次の表に示す環境で行った.

ハードウェア	CPU	Pentium†®Ⅲ 800MHz
	RAM	255Mbyte
ソフトウェア	OS	Microsoft®Windows††®98
	コンパイラ	Microsoft®Visual C++6.0
	コンパイルオプション	実行速度 (O2)
	言語	ANSI C

評価は鍵 10 対を使用し, 鍵ペア 1 対に対して平文 100 組 (20 組の平文を 5 回使用) を用いて合計 1000 回の暗号化・復号化処理を行い, 処理速度の各平均の算出を行った. またプログラムのステップ数は 2781 ステップ, 実行ファイルのの大きさは 84KByte である. 次の表に評価結果を示す.

測定内容	平均速度
暗号化	0.6 ms
復号化	37.0 ms

今回の実装はリファレンス実装のため, 数値そのものは参考値であり, 今後, 実装の改良により大幅な性能向上が期待できる.

3.5 ハードウェア実装評価

ここでは, HIME(R) アルゴリズムのハードウェア評価について報告する. 本評価では, 主演算器, 演算用レジスタであるアキュームレータ, 途中経過保持用のレジスタ, HIME(R) を実現するマイクロプログラムを実行する制御部, マイクロプログラムおよびデータを格納するメモリで構成した. 概要は以下の通り.

演算器

以下の演算器を主演算器として用いる.

加算器	1344bit の加算器. 加算, 乗算, 除算, 剰余演算に用いる.
EX-OR	2 つの 1344bit の値のビット毎の排他的論理和を取る.
シフタ	1344bit 幅で 1,32,64,128,256,512bit のシフトが可能なシフタ.
SHA-1	512bit から 160bit のハッシュ値を求める専用演算器. 512bit のレジスタ hash と 160bit のレジスタ sha-1 を持つ.

アキュームレータ

1344bit のアキュームレータ 3 本

レジスタ

1344bit のレジスタ 5 本

†Pentium は、米国およびその他の国における Intel Corp. またはその子会社の商標または登録商標です.
††Microsoft、Windows は、米国 Microsoft Corporation. の米国及びその他の国における登録商標または商標です.

制御部

プログラムカウンタ (PC), フラグレジスタ (FR), ループ回数制御用レジスタ (count) を備える. その他, メモリ制御, システムインタフェース制御を行う.

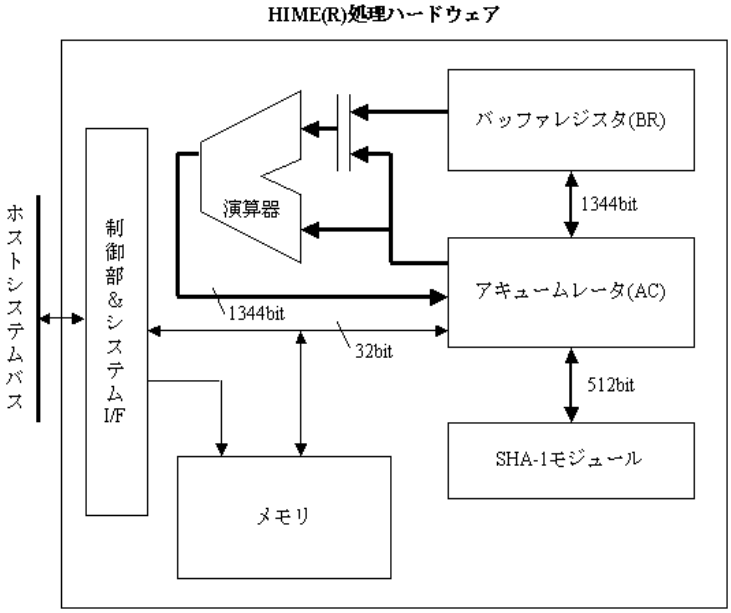

図 4: ブロック図

ハードウェア規模概算

アキュームレータ	FF	6gate×1344bit×3 個	24K gate
	制御	7gate×1344bit×3 個	28K gate
バッファレジスタ	Latch	4gate×1344bit×5 個	27K gate
演算器	加算器	1344bit	20K gate
	EX-OR	2.3gate×1344bit	3K gate
	シフタ	21gate×1344bit	26K gate
SHA-1			19K gate
制御	sel	1344 to 1 selector	3K gate
	PC	8.3gate×16bit+160gate	0.3K gate
	count	10gate×11bit	0.1K gate
	メモリ制御		0.3K gate
	その他 (システム I/F 含む)		3K gate
合計			153.7K gate

概算の結果, HIME(R) 処理ハードウェアの論理回路部分は, 約 154Kgate で実現できることがわかった.

命令

分岐, 条件分岐等の制御命令, データ転送命令, 加算/減算/乗算/除算/剰余演算の演算命令, フラグや値の大小関係で演算を行う条件付き命令を備える.

データ転送命令には, 同じデータの連結をサポートする命令を備える.

乗算, 除算, 剰余演算を除いて, 基本的に 1 命令 1 クロック動作を行う.

詳細説明は省略.

処理時間

HIME(R) 処理プログラムを実行した場合, 以下のクロック数を必要とする.

暗号化： 3417 クロック

復号化： 1647527 クロック (最大)

本ハードウェアの動作クロックを 33MHz とすると各処理に必要な時間は以下の通りである.

暗号化： 103 μs

復号化： 49.4ms

メモリ 本ハードウェアに必要なメモリは, HIME(R) 処理プログラムを格納する ROM 6KByte と引数受け渡し用の RAM 512Byte である.

以上をまとめると, HIME(R) 処理ハードウェアの規模, および処理性能は以下のようになる.

論理回路規模	154 Kgate
メモリ	ROM 6 KByte + RAM 512 Byte

33MHz 動作時	
暗号化	103 μs
復号化	49.4 ms (最大)

4 結論

本ドキュメントは，HIME(R) が公開鍵暗号として理想に近い性質を持つことを示した．

HIME(R) は，合成数 $N = p^d q$ (但し，p,q は素数．$d > 1$) を法とする剰余環上のモジュラー平方関数を用いた公開鍵暗号である．HIME(R) では IND-CCA2 の意味において安全であることを，素因数分解問題の困難性と等価性にて証明できた．多くの実用的公開鍵暗号方式では，その安全性の根拠を素因数分解問題ベース（素因数分解問題，平方剰余問題，等）または離散対数問題ベース（離散対数問題，Diffie-Hellman 問題，等）の数論的問題の計算量的困難性に依存しており，素因数分解問題および離散対数問題は，これら２つのカテゴリーの中で最も難しい問題であることから，素因数分解問題を暗号学的仮定として用いることは（同一カテゴリーに含まれる他の問題を仮定とする場合に比べて）理想的であると言える．

また，HIME(R) の暗号化速度は極めて速く（１回のモジュラー積のみ）, 復号化速度は RSA-OAEP（1024 bits）と HIME(R)（1536 bits）を比較すると HIME(R) は RSA-OAEP の約 2.5 倍速いことを示した（モジュラー積の個数による比較）．さらに，データ暗号化鍵とともに様々な付加情報（例えばユーザのＩＤ情報など）を同時に暗号化して配送することが可能である十分に大きな平文空間を持つなどの実用的暗号として特徴を兼ね備えていることを示した．特に，急速な計算機の進歩により 公開鍵暗号は鍵長延長を余儀なくされているが, HIME(R) は RSA-OAEP 等の従来方式との効率性の比較において優位であり, 将来的に有望な方式であるといえる．このことは，N が十分大きくなると数体ふるい法による素因数分解の効率が，楕円曲線法の効率を上回ってしまうことに起因している．

以上のことから, HIME(R) は現在のみならず将来的においても実用性に優れた安全な公開鍵暗号方式であると考える．

参考文献

[1] M. Bellare, A.Desai, D.Pointcheval and P. Rogaway. : Relations among notions of security for public-key encryption schemes, *Advances in Cryptology – Crypto'98*, LNCS 1462, Springer-Verlag, pp.26–45 (1998)

[2] M. Bellare and P. Rogaway. : Random oracles are practical – a paradigm for designing efficient protocol, *First ACM Conference on Computer and Communications Security*, pp.62–73 (1993)

[3] M. Bellare and P. Rogaway. : Optimal asymmetric encryption – How to encrypt with RSA, *Advances in Cryptology – Eurocrypt'94*, LNCS 950, Springer-Verlag, pp.92–111 (1994)

[4] D. Bleichenbacher. : Chosen Ciphertext Attacks against Protocols Based on the RSA Encryption Standard PKCS#1, *Advances in Cryptology – Crypto'98*, LNCS 1462, Springer-Verlag, pp.1–12 (1998)

[5] M. Blum and S. Goldwasser. : An efficient probabilistic public-key encryption scheme which hides all partial information, *Advances in Cryptology – Crypto'84*, LNCS 196, Springer-Verlag, pp.289-299 (1985)

[6] D. Boneh. : Simplified OAEP for the RSA and Rabin functions, *Advances in Cryptology – Crypto2001*, LNCS 2139, Springer-Verlag, pp.275-291 (2001)

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

[8] Call for Contributions on New Work Item Proposal on Encryption Algorithms, NTT, 2000-3-10.

[9] D. Coppersmith. : Modifications to the number field sieve, *Journal of in Cryptology*, 6, 3, pp.169-180 (1993)

[10] D. Coppersmith. : Finding a small root of a univariate modular equation, *Advances in Cryptology – Eurocrypt'96*, LNCS 1070, Springer-Verlag, pp.155-165 (1996)

[11] R. Cramer and V. Shoup. : A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack, *Advances in Cryptology – Crypto'98*, LNCS 1462, Springer-Verlag, pp.13-25 (1998)

[12] D. Dolve, C. Dwork and M. Naor. : Non-malleable cryptography, *Proceedings of the 23rd Annual Symposium on Theory of Computing*, ACM, pp.542–552 (1991)

[13] T. ElGamal. : A public key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Trans. Information Theory*, IT-31, 4, pp.469-472(1985)

[14] E. Fujisaki, T. Okamoto and D. Pointcheval : RSA-OAEP is secure under the RSA assumption, *Advances in Cryptology – Crypto2001*, LNCS 2139, Springer-Verlag, pp.269-274 (2001)

[15] S. Goldwasser and M. Bellare. : *Lecture Notes on Cryptography*, `http:/www-cse.ucsd.edu/users/mihir/` (1997)

[16] S. Goldwasser and S. Micali: Probabilistic encryption, *Journal of Computer and System Sciences*, 28, 2, pp.270–299 (1984)

[17] D.M. Gordon : Designing and detecting trapdoors for discrete log cryptosystems, *Advances in Cryptology – Crypto'92*, LNCS 740, Springer-Verlag, pp.66-75 (1992)

[18] Specification of HIME-1 CryptoSystem, Hitachi, Ltd. (2000)

[19] Specification of HIME-2 CryptoSystem, Hitachi, Ltd. (2000)

[20] D. E. Knuth. : *The Art of Computer Programming*, Addison-Wesley (1981)

[21] N. Koblitz. : Elliptic curve cryptosystems, *Math. Comp.*, 48, 177, pp.203-209 (1987)

[22] A.K. Lenstra and H.W. Lenstra,Jr. : *The Development of the Number Field Sieve*, Lect. Notes Math. 1554, Springer-Verlag (1993)

[23] H.W. Lenstra,Jr. : Factoring integers with elliptic curves, *Annals of Math.*, 126, pp.649-673 (1987)

[24] J. Manger : A chosen ciphertext attack on RSA optimal asymmetric encryption padding (OAEP) as standardized in PKCS#1 v2.0, *Advances in Cryptology – Crypto2001*, LNCS 2139, Springer-Verlag, pp.230-238 (2001)

[25] V. S. Miller. : Use of elliptic curves in cryptography, *Advances in Cryptology – Crypto'85*, LNCS 218, Springer-Verlag, pp.417-426 (1985)

[26] National Institute of Standards, FIPS Publication 180, Secure Hash Standards (1993)

[27] M.Naor and M.Yung. : Public-key cryptosystems provably secure against chosen ciphertext attacks, *Proceedings of the 22nd Annual Symposium on Theory of Computing*, ACM, pp.427–437 (1990)

[28] T. Okamoto and D.Pointcheval: EPOC-3: Efficient Probabilistic Public-Key Encryption-V3 (Submission to P1363a), May 2000

[29] J. M. Pollard. : A Monte-Carlo method for factorization, BIT 15, pp.331-334 (1975)

[30] M. O. Rabin. : Digital signatures and public-key encryptions as intractable as factorization, MIT, Technical Report, MIT/LCS/TR-212 (1979)

[31] R. L. Rivest, A. Shamir and L.Adleman. : A method for obtaining digital signatures and public-key cryptosystems, *Communications of the ACM*, Vol.21, No.2, pp.120-126 (1978)

[32] V. Shoup. : OAEP reconsidered, *Advances in Cryptology – Crypto2001*, LNCS 2139, Springer-Verlag, pp.239-259 (2001)

[33] T. Takagi. : Fast RSA-type Cryptosystem Modulo $p^k q$, *Advances in Cryptology – Crypto'98*, LNCS 1462, Springer-Verlag, pp.318-326 (1998)

[34] H.C.Williams. : A modification of the RSA public key encryption procedure, *IEEE Trans. on Information Theory*, IT-26, 6, pp.726-729 (1980)

[35] H. Woll. : Reductions among number theoretic problems, *Information and Computation*, 72, 3, pp.167-179 (1987)

[36] Y. Zheng and J. Seberry. : Practical approaches to attaining security against adaptive chosen Ciphertext Attacks, *Advances in Cryptology – Crypto'92*, LNCS 740, Springer-Verlag, pp.292-304 (1992)