

# RSA暗号計算 No.1

## RSA暗号とユークリッド互助法

2007年5月

後 保範 (東京工芸大学)

# 目次

1. 暗号化方式
2. RSA暗号化と復号化の方法
3. 算術演算の時間評価
4. ユークリッド互除法とオイラー関数
5. 素数か合成数かの判定法

# 1. 暗号化方式

## (1) 秘密鍵方式

共通鍵、対称鍵とも言う

代表例: DES(Data Encryption Standard), IDEA

長い文章全体を暗号化する

## (2) 公開鍵方式

非対称鍵とも言う。公開鍵と秘密鍵を使用

代表例: RSA方式

短いメッセージ、電子署名、共通鍵の交換

# 1.1 秘密鍵方式

- 送信者と受信者は1つの鍵を秘密に共有し、暗号化と復号化に共通の鍵を使う。
- 暗号の強度を確保するには鍵の長さが重要なファクターとなるが、鍵の長さが増せば処理コストが増加する。現在64ビット又は128ビット
- DESなどのように暗号アルゴリズムを公開したものと、軍事用の様に非公開のものがある
- 受信者が遠隔地にいる場合、この秘密の鍵を安全でない通信路を使って、どのように鍵を配布するのが問題となる。

## 1.2 公開鍵方式

- 公開鍵と秘密鍵の2組の鍵を使う方法を言う
- 「短いメッセージの暗号化」、「電子署名」、「共通鍵の交換」など目的の異なる用途に用いられる。
- 1組の秘密鍵と公開鍵で多数の相手に通信可能
- 公開鍵方式は共通鍵方式に比べて処理コストが高い(1024ビットRSAではDESの約1000倍)
- 素因数分解(RSA)の困難さを使って定義された暗号化や復号化の方式を使う
- 送信者(暗号化する人)は秘密鍵を知らない

## 2. RSA暗号化と復号化の方法

### RSA暗号鍵の作成

- (1) 素数 $p, q$ を選ぶ。
- (2)  $n = p \times q$ 及び $f = (p-1) \times (q-1)$ を計算
- (3) 素数 $e$ を選ぶ。
- (4)  $d = 1/e \pmod{f}$ となる $d$ を計算する。

$(e, n)$ が公開暗号化鍵、 $(d, n)$ が復号鍵となる。

## 2.1 暗号化と復号化

### RSA暗号化

- (1) 文を $n$ 以下の数 $M$ に変換(公開方法)
- (2)  $C=M^e \bmod(n)$ で暗号 $C$ を作成

### RSA復号化

- (1)  $M=C^d \bmod(n)$ で元の数 $M$ に復号
- (2) 数 $M$ を文に変換(公開方法)

## 2.2 復号化の数学理論

$n=p \times q$ と $M$ が互に素なら  $M^{(p-1)(q-1)}=1 \pmod{n}$   
なるオイラーの定理を使用

$$\begin{aligned} C^d \pmod{n} &= (M^e)^d \pmod{n} = M^{\alpha f+1} \pmod{n} \\ &= (M^{(p-1)(q-1)})^\alpha \times M \pmod{n} \\ &= M \pmod{n} = M \end{aligned}$$

$ed = 1 \pmod{f}$ で  $ed = \alpha f + 1$  ( $\alpha$  は整数)を利用



## 2.3 RSA暗号変換例

### (1) 鍵とメッセージ

公開暗号鍵:  $e=43291$ 、 $n=130733$

秘密復号鍵:  $d=105691$

メッセージ: YES  $\iff$   $M = 16836$

### (2) 暗号化

$$C = M^e = 16836^{43291} = 73724 \pmod{130733}$$

### (3) 復号化

$$M = C^d = 73724^{105691} = 16836 \pmod{130733}$$

## 2.3.1 RSAの鍵の作成例

(1) 素数 $p, q$ を選び $n, f$ を計算

$p=239, q=547$  : 選定

$n=p \times q=130733, f=(p-1) \times (q-1)=129948$

(2) 暗号鍵(素数, $e < f$ )を選定

$e = 43291$  ,  $(e, n) = (43291, 130733)$ を公開

(3) 復号鍵(秘密鍵, $d$ )の計算

$d = 1/e = 1/43291 = 105691 \pmod{f}$

この計算はユークリッド互除法で計算する

## 2.3.2 復号鍵の具体的計算(1/3)

$(f,e)=(129948,43291)$ にユークリッド互除法

$$129948 = 3 \times 43291 + 75 \rightarrow \dots$$

$$43291 = 577 \times 75 + 16 \rightarrow \dots$$

$$75 = 4 \times 16 + 11 \rightarrow \dots$$

$$16 = 1 \times 11 + 5 \rightarrow 5 = 16 - 1 \times 11$$

$$11 = 2 \times 5 + 1 \rightarrow 1 = 11 - 2 \times 5$$

## 2.3.2 復号鍵の具体的計算(2/3)

互除法の結果(逆順)からe,fの一次式を作成

$$\begin{aligned}1 &= 11 - 2 \times 5 = 11 - 2 \times (16 - 1 \times 11) \\ &= 3 \times 11 - 2 \times 16 = 3 \times (75 - 4 \times 16) - 2 \times 16 \\ &= 3 \times 75 - 14 \times 16 \\ &= 3 \times 75 - 14 \times (43291 - 577 \times 75) \\ &= 8081 \times 75 - 14 \times 43291 \\ &= 8081 \times (f - 3 \times e) - 14 \times e \\ &= 8081 \times f - 24257 \times e\end{aligned}$$

## 2.3.2 復号鍵の具体的計算(3/3)

$1 = 8081 \times f - 24257 \times e \pmod{f}$  の計算

$$1 = 8081 \times f - 24257 \times e \pmod{f}$$

$$= -24257 \times e \pmod{f}$$

$$= (f - 24257) \times e \pmod{f}$$

$$= (129948 - 24257) \times e \pmod{129948}$$

$$= 105691 \times e \pmod{129948}$$

従って複合鍵(d)は

$$d = 1/e = 105691 \pmod{129948}$$

# 3. 算術演算の時間評価

## 算術演算の時間評価

(1) 暗号処理では、整数 $n$ 又はその桁数 $m$ に対する演算量を $O(n)$ ,  $O(n^{1/4})$ や $O(2^m)$ ,  $O(m^5)$ のように表示する。

(2) 代表的な時間評価

$O(\alpha \log(n)), O(m^\alpha)$  : 多項式時間

$O(2^{c\sqrt{m\log(m)}})$  : 中間時間(指数と多項式)

$O(n^c), O(2^{cm})$  : 指数時間( $c, \alpha$ は定数)

## 3.1 主な演算の時間評価

$n$ は2進 $m$ 桁の数とする。

### 1. RSA暗号化及び複合化

(1)  $M^e$  or  $C^d \pmod{n}$  :  $O(m^2 \log(m))$

### 2. 素数判定

(1) 2002年のアルゴリズム :  $O(m^3)$

### 3. 素因数分解

(1) 試行割算法 :  $O(2^{m/2})$

(2)  $\rho$ 法 :  $O(m^3 2^{m/4})$

(3) 2次ふるい法 :  $O(2^{(m \cdot \log(m))^{1/2}})$

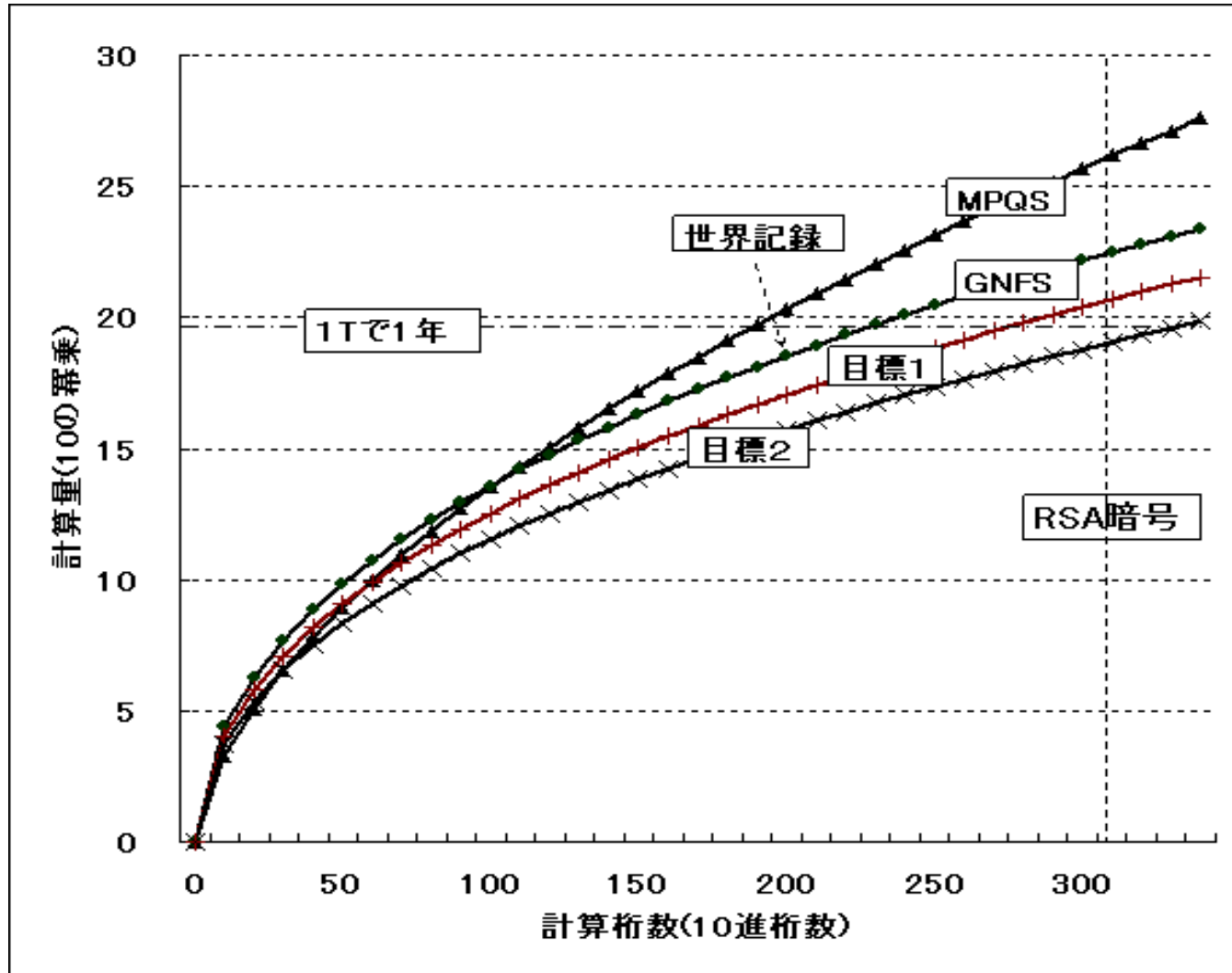
(4) 数体ふるい法 :  $O(2^{m^{1/3} \cdot \log(m)^{2/3}})$

## 3.2 時間評価の具体的値

m	$m^3$	$2^{\sqrt{m \log(m)}}$	$2^{m/4}$	$2^m$
50	$1. \times 10^5$	$1. \times 10^5$	$6. \times 10^3$	$1. \times 10^{15}$
100	$1. \times 10^6$	$6. \times 10^7$	$3. \times 10^7$	$1. \times 10^{30}$
200	$8. \times 10^6$	$6. \times 10^{11}$	$1. \times 10^{15}$	$2. \times 10^{60}$
400	$6. \times 10^7$	$5. \times 10^{17}$	$1. \times 10^{30}$	$3. \times 10^{120}$
800	$5. \times 10^8$	$3. \times 10^{26}$	$2. \times 10^{60}$	$7. \times 10^{240}$



### 3.3 RSA暗号解読の計算量



## 4. ユークリッド互除法とオイラー関数

最小公倍数 (Greatest Common Divisor)

- (1)  $\text{GCD}(a,b)$  : 整数 $a,b$ の最小公倍数
- (2)  $a > b$ で下記が成立 ( $a/b$ は整数以外)  
$$\text{GCD}(a,b) = \text{GCD}(a \bmod(b), b)$$
- (3) (2)を繰り返し使用すると  
ユークリッド互除法が得られる。

## 4.1 ユークリッド互除法の計算例

GCD(1547,560)を求める

$$\text{GCD}(1547, 560)$$

$$= \text{GCD}(1547 \bmod(560), 560)$$

$$= \text{GCD}(427, 560)$$

$$= \text{GCD}(427, 560 \bmod(427))$$

$$= \text{GCD}(427, 133) = \text{GCD}(28, 133)$$

$$= \text{GCD}(28, 21) = \text{GCD}(7, 21) = 7$$

## 4.2 オイラーの関数( $\varphi(n)$ )

### 1. オイラー関数( $\varphi(n)$ )の定義

$n$ より小さく $n$ とは素な非負整数の個数

$$\varphi(n) \equiv |\{0 < t < n \mid \text{GCD}(t, n) = 1\}|$$

### 2. $\varphi(n)$ の計算方法

$$n = \prod_{j=1}^m p_j^{\alpha_j} \text{とする。}$$

$$\varphi(n) = \prod_{j=1}^m p_j^{\alpha_j} (p_j - 1)$$

## 4.3 オイラーの定理

### 1. フェルマーの小定理

素数 $p$  と整数 $M$  ( $<p$ ) に対して

$$M^{p-1} = 1 \pmod{p}$$

### 2. オイラーの定理

$GCD(a, m) = 1$  ならば

$$a^{\varphi(m)} = 1 \pmod{m}$$

## 4.4 平方剰余と相互法則

### 平方剰余とルジャンドル記号

- (1)  $b^2 = a \pmod{p}$  となる、整数  $b$  が  
存在する :  $a$  は法  $p$  で平方剰余  
存在しない :  $a$  は法  $p$  で非平方剰余

- (2) ルジャンドル記号 ( $p$  は 3 以上の素数)

$$\left(\frac{a}{p}\right) = a^{(p-1)/2} \pmod{p} = \begin{cases} 0 : a \text{ が } p \text{ の倍数} \\ 1 : a \text{ が法 } p \text{ で平方剰余} \\ -1 : a \text{ が法 } p \text{ で非平方剰余} \end{cases}$$

# 5. 素数か合成数かの判定法

- 昨年(2002年)多項式時間で、素数か合成数かを判定する方法をインドの数学者が発見
- インドのカンプルにあるインド工科大学で、Manindra Agrawal氏と、その教え子のNeeraj Kayal氏、Nitin Saxena氏が開発した新しいアルゴリズムは、毎回正確な結果が得られるとされている。
- 米ラトガーズのニュージャージー州立大学でコンピュータサイエンスを教えるEric Allender教授は、次のように語る。「現在の暗号化ソフトの最も明白な弱点は、素数判定の結果が正しいという保証がない点だ。この新しいアルゴリズムは、何世紀も前から存在し、数十年にわたって熱心な研究が行われてきた根本的な疑問を解決するものだ」。
- このアルゴリズムに関するAgrawal氏の論文「PRIMES is in P」は2002年に発表され、この分野でかなりの話題を呼んでいる。古代中国やギリシャの時代から、数学者の心を捕らえてきた数学的問題を解決する方法だからだ。

## 5.1 Rabin素数判定法

- 確率的な素数判定法(素数確率:  $1 - (1/4)^L$ )

$n-1=2^s m$ となる整数 $s$ と奇数 $m$ を求める。

$k=1, 2, \dots, L$ か合成数となるまで反復する。

$2 < b < n-1$ なる乱数 $b$ を求める。

$y = b^m \pmod n$

if( $y=1$  or  $y=-1$ )  $\rightarrow$   $n$ は擬素数(次の $k$ へ)

$i=1, 2, \dots, s$ まで反復

$y = y^2 \pmod n$

if( $y=1$ )  $\rightarrow$   $n$ は合成数(終了)

if( $y=-1$ )  $\rightarrow$   $n$ は擬素数( $i$ の反復終了, 次の $k$ へ)

$n$ は合成数(終了)