G-23

数値解析を用いた Java による RSA 暗号

○日本大学理工学部数学科 4 年　山川　祐太朗

1.概要

暗号化は以前より様々な方法が考えられてきた。RSA暗号はそれらの内の一つであり、今現在最も安全とされている暗号である。本論文では、この RSA 暗号をプログラミング言語 Java を用いて作成する。Java はオブジェクト指向のプログラミング言語であり、RSA 暗号の中心となる大整数の演算も「大整数クラス」が準備されているため作成時に都合がよいためである。

また RSA 暗号を理解した上で、改良点を見つけ、数値解析を用いることでより複雑な暗号アルゴリズムを作成していく。

2.暗号

「暗号」とは、情報を秘匿する技術である。すなわち、ある文を意味が分からないように変換した文が暗号である。この際、元々の文は「平文」、変換された文は「暗号文」と呼ばれ、平文から暗号文への変換を「暗号化」、逆を「復号化」と言う。

この暗号化や復号化は「鍵（key）」を用いて行われる。第三者はこの鍵を見つけることで暗号の解読が可能となるため、解読が不可能、もしくは限りなく難しい暗号システムを構築することが重要である。

2-1.公開鍵暗号システム

暗号にも色々な形式があるが、本論文の中心となる RSA 暗号は「公開鍵暗号システム」という形に属する。

このシステムの概要を図示したものが**図 2-1** であり、形式化すると次のようになる。

$$C = E_e(P),$$
$$P = D_d(C),$$
$$e \neq d$$

(eは公開鍵、dは秘密鍵)

2-2.RSA 暗号

本節では本論文の中核である「RSA暗号」について述べる。RSA 暗号は、アメリカの Rivest、Shamir、Adleman によって提案された公開鍵暗号システムであり、大きい整数の「素因数分解」が困難であることを利用した暗号システムである。

現在、「素因数分解」の効率的なアルゴリズムは知られていないため、この計算の非効率性が RSA 暗号の安全性の根底となっている。

3.RSA アルゴリズム

では、RSA のアルゴリズムについて説明していく。

まず、二つの大きな素数(p, q)を選び、その合成数である

$$n = pq$$

を求める。

次にオイラー関数$(\varphi(n) = (p-1)(q-1))$を計算し、

$$gcd\big(d, \varphi(n)\big) = 1,$$
$$ed \equiv 1 \bmod \varphi(\mathrm{n})$$

を満足する対(e, d)を求める。ここで、公開鍵は対(n, e)であり、秘密鍵はdである。

暗号化では、送信者 A は平文(m)を

$0 \leq m < n$の整数で記述し、受信者 B の公

開鍵(n,e)を用いて暗号文(c)を

$$c = m^e \pmod n$$

とする。なお、cは $0 \leq c < n$ を満足する整数となる。

よって、暗号化関数(E_e)は、

$$E_e = m_e \pmod n$$

と定義される。ここでeは B の公開鍵である。

復号化では、B は暗号文を自分だけが知っている秘密鍵(d)を用いて、

$$m = c^d \pmod n$$

より、平文(m)を得る。よって復号化関数(D_d)は、

$$D_d = c^d \pmod n$$

と定義される。

以上が RSA のアルゴリズムである。

3-1.補足、使用定理

上記のdを求める際は、最大公約数を求めるためにユークリッド・アルゴリズム、そして拡張ユークリッド・アルゴリズムを用いる。以下にこれらの定義を示す。

3-1-1.ユークリッド・アルゴリズム

任意の$a,b \in \boldsymbol{Z}$について$(a > b > 0)$、ある$q,r \in \boldsymbol{Z}$が、

$$a = bq + r$$

を満たすならば、

$$gcd(a,b) = gcd(b,r)$$

である。

3-1-1.拡張ユークリッド・アルゴリズム

ユークリッド･アルゴリズムを$a,b \geq 0$に適用した時の剰余数列を、$r_0 = a, r_1 = b, \cdots, r_{n+1}$とし、商数列を$q_1, q_2, \cdots, q_n$ とする。今、$x = (-1)^n x_n,\ y = (-1)^{n+1} y_n$が次のような性質を満足するように、数列$(x_k),(y_k)$を構成する。

$x_0 = 1, x_1 = 0, y_0 = 0, y_1 = 1,$

$x_{k+1} = q_k x_k + x_{k-1}, y_{k+1} = q_k y_k + y_{k-1}$

$(1 \leq k \leq n)$

そうすると、$1 \leq k \leq n+1$ について、

$$r_k = (-1)^k x_k a + (-1)^{k+1} y_k b$$

が成り立つ。

4.まとめ

RSA 暗号のアルゴリズムを作成する上で、必要な Java の知識・大整数クラス「BigInteger」の利用を学び、ユークリッド・アルゴリズム、拡張ユークリッド・アルゴリズムを Java を用いて作成してきた。その上で、実際に RSA 暗号のアルゴリズムを Java で作成し作動することが確認できた。しかし、課題として、大整数ではない 7 や 11 等の小さい整数を入力した場合、平文と暗号文に差は見られなかった。ユークリッド・アルゴリズムによる最大公約数を求める際に小さな整数同士であると、差が出ない事に起因すると考えているが、より正確に追求していきたい。

更に、この RSA では数字のみの暗号であるため、英文などは対応していない。そこに注目し、アルファベットを数字に変換するアルゴリズムを作成し、挿入することでより実用的になり、かつ複雑な暗号化になると考えている。

参考文献

赤間世紀　Java による暗号理論入門(工学社)

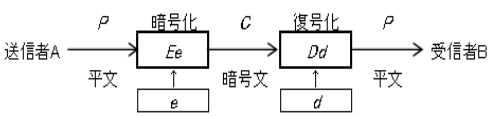

図 2-1:公開鍵暗号システム