電気通信大学 電気通信学部 情報通信工学科

情報通信工学実験 A・B
実験項目 1. 情報通信 — 情報・セキュリティ —

課題「暗号化の理解と実装」
— — RSA 暗号の理解と実装 (プログラミング) — —[1]

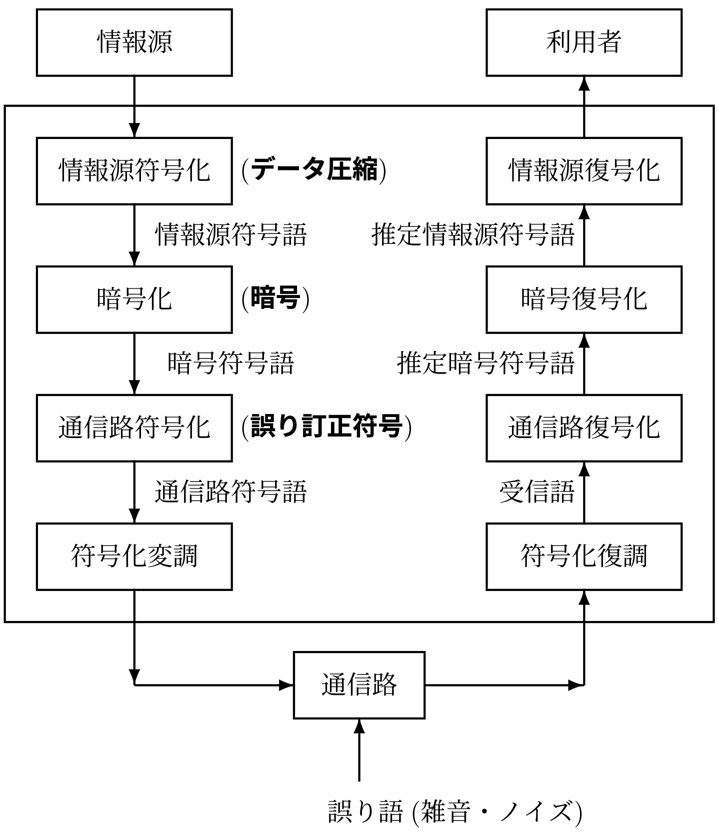
本実験項目「情報・セキュリティ」を履修する学生は、実験担当教員の指示に従って以下の 3 課題の中から一つの課題を選択し実験を行なう。

本テキストは、課題 **「暗号化の理解と実装」— RSA 暗号の理解と実装 (プログラミング)** — に関するテキストである。

1. 「データ圧縮の理解と実装」— ハフマン符号の理解と実装 (プログラミング) —
2. 「暗号化の理解と実装」— RSA 暗号の理解と実装 (プログラミング) —
3. 「誤り訂正符号の理解と実装」— リード・ソロモン符号と最小距離復号の理解と実装 (プログラミング) —

実験項目「情報・セキュリティ」では、下記図に示す 3 種類の**符号化**に関するテーマを扱う。一般に、情報を送信者 (情報源) から受信者 (利用者) に送信する際、次のような 3 種類の符号化を行なう。はじめに、効率性を目的として、**情報源符号化（データ圧縮）**を行なう。次に、安全性を目的として、**暗号化（暗号化）**を行なう。最後に、信頼性を目的として、**通信路符号化（誤り訂正符号）**を行なう。本実験では、いずれかの符号化の具体的手法について理解し、実際に計算機上でプログラムを作成し、実装を行なう。

ディジタル伝送・記憶システム



[1]©電気通信大学 情報通信工学科 栗原正純 (e-mail: kuri@ice.uec.ac.jp), 2003 – 2010. (2010/9/22/18:35 修正)
本テキストは、http://www.code.ice.uec.ac.jp/kuri/C3/ より入手可能である。
(11x3: /doc/tex/uec/jikken/2010/istext2010crpt.tex)

目的（課題）

1. 暗号の一つである RSA 暗号の具体的な暗号化と復号化の方法について理解する．
2. 次に，RSA 暗号の暗号化と復号化のプログラムを作成し，計算機上で実装する．具体的には、5 節の課題を解く。

提出レポートの内容

1. 「RSA 暗号」をキーワードにして「暗号」と「認証」について調べたことを１ページ以内（A4 サイズ）に簡潔にまとめ，記述する．
2. 暗号化・復号化のプログラムを実装するに当たり，プログラミングで工夫した点を記述する．
3. プログラムのソースを印刷し，プログラムの各行（あるいは各部分）が何を実行する箇所なのかなどの注釈を記入したものをレポートに添付する．
4. レポートには “参考文献” という節（あるいは項目）を設け，主に参考にした文献を明記する．ここで，文献とは，書籍に限らない．インターネットなどを利用して得られたものも明記すること．その場合，URL(Uniform Resource Locator)(Web アドレス) を明記する．また，そのページのタイトルがあればそれも明記する．自分のアイデアと他人のアイデアを明確に区別すること．
5. 紙のレポートとは別にソースプログラムをメイルにて担当教員宛に送る．その際，そのファイルのコンパイル方法，実行方法も忘れずにメイルにて送る．

   Subject 欄には半角英数字を用いて “3jikken(name)” と記述し，メイル文章の初めに，氏名と学籍番号を書くこと．ここで、 “name” は各自の氏名のローマ字表記を書くこと。

1　はじめに

本テキストの目的は、本実験課題で扱う RSA 暗号 [5] とよばれる暗号化アルゴリズムの説明およびその実装のための諸注意などを説明することにある。

一般に、あるひとつの暗号化アルゴリズムを定義したり、説明する方法は、一通りではなく幾通りかあり、それぞれ趣が異なる場合がある。そこで、本テキストでは、本実験課題の目的を達成するのに都合のよいと思われる方法で種々の定義や説明を与えることにする。もっと一般的な暗号（暗号化）の取り扱いについては、暗号に関連する講義や暗号の専門書の内容に譲ることとする。

本テキストでの暗号に関する説明を記述するにあたり、文献 [1, 2] を参考にした。

2　暗号化

本節では、暗号化の具体例とそれを通して暗号システムの形式的な記述および暗号システムの分類について説明する。

2.1　暗号の具体例（シフト暗号）

はじめに、暗号やそれに関連した数学的準備などの形式的な話しをする前に、具体的な暗号についてみていこう。

A さんから B さんへ伝えたいメッセージを第三者の C さんに見られても分からないようにメッセージを暗号化することを考える。そこで、以下のような暗号を利用することにする。

定義 2.1 メッセージおよびメッセージを暗号化した暗号文はいずれも 26 文字からなるアルファベット, A,B,...,Z, により構成されているものとする。アルファベットの各文字を辞書式順序に並べる。

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

このとき、暗号化は上記の順序に従って各文字を右へ k だけ循環シフトさせる変換に対応させる。このようにして暗号化された暗号文をもとのメッセージに復元する（復号化する）には、暗号文の各文字を左に k だけ循環シフトさせる変換を行なえばよい。 ∎

この暗号化の様子から上記の暗号を**シフト暗号**ということにする。実際には、上記の暗号を利用して A さんから B さんへ暗号文を送信する前に、k という鍵となる情報を共有しておく必要がある。しかも、この情報は第三者の C さんには知られていないものとする。

例 2.2 例えば、$k = 5$ として、A さんが B さんに伝えたいメッセージ
WEWILLMEETATCHOFUSTATION
を暗号化するとその暗号文は、
BJBNQQRJJYFYHMTKZXYFYNTS
となる。これを B さんに送信すればよい。

第三者の C さんが A さんと B さんの間でシフト暗号を利用することを知っていても、鍵 k の値を知らなければ暗号文からもとのメッセージを読みとることはできない、と言えなくもない。データ伝送の簡単な様子を図 1 に示す。しかし、現実的には、鍵の選択肢は 26 通りしかないのですべての値を試して、復号化後の文を読むことで暗号文を解読することはそう難しくはないかもしれない。

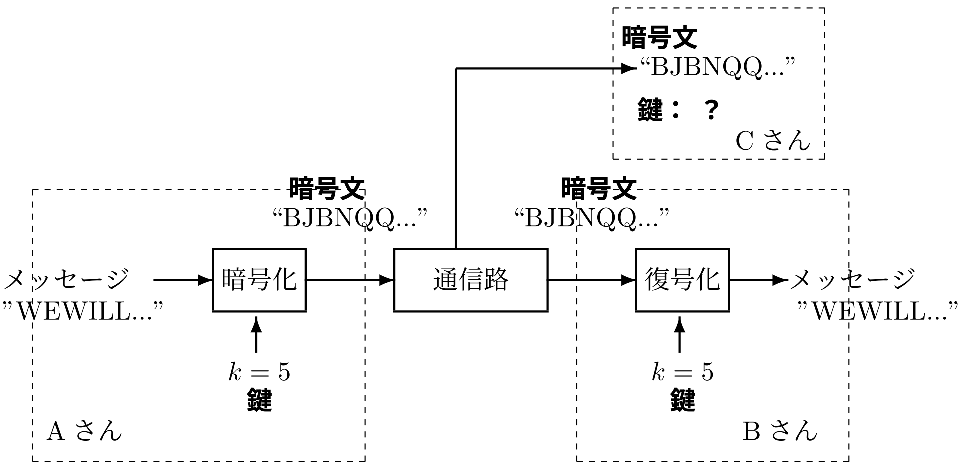


図 1: シフト暗号を用いたデータ伝送の様子

∎

2.2 暗号システムの形式的記述

本節では、前記の具体的な暗号の例を参考に、暗号に関連する定義や術語などを整理する。

暗号の目的は、A さんと B さんの二人の間で盗聴可能な通信路を利用して情報を通信することができることである。ただし、その際、第三者の C さんにはその情報の内容が分からない方法で通信できなければならない。このような盗聴可能な通信路を**公開通信路**ということにする。

伝達したいメッセージや情報を**平文（ひらぶん）**といい、記号 M で表す。暗号化用の鍵 K_e と K_e によって定まる復号化用の鍵 $K_d(= K_{K_e})$ をもつある暗号方法を利用することを考える。ここで、K_{K_e} の意図は、復号化用の鍵が、暗号化用の鍵 K_e によって定まることを示すために、添字に K_e を記述している。このとき、A さんと B さん以外に C さんも利用する暗号方法を既知とするが、暗号化用の鍵 K_e を知るのは A さんのみで、また、復号化用の鍵 K_d を知るのは B さんのみであるとする。

A さんは暗号化用の鍵 K_e を用いて伝達したい平文 M を**暗号文** C に変換し、公開通信路を利用して暗号文 C を B さんに送る。B さんは復号化用の鍵 K_d を用いて受け取った暗号文 C をもとの平文 M に変換し、A さんが伝えたかった情報を理解することができる。一方、C さんは公開通信路を盗聴することで暗号文 C を入手することはできるが、復号化用の鍵 K_d を持たないため

暗号文 C からもとの平文 M を得ることは困難で、容易にその内容を知ることはできない。上記の説明の概略を図 2 に示す。このような暗号方法の考え方を少しばかり形式的に以下に示す。

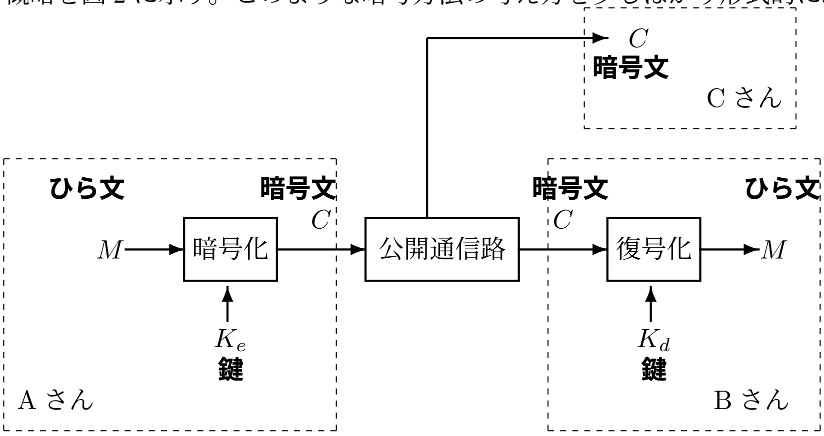


図 2: 暗号システムの概略図

定義 2.3 暗号システムは次の 6 項目を満たす 5 つの要素 $(\mathcal{M}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ から構成されるものと考える（定義しよう）。

1. $\mathcal{M}$：平文全体からなる有限集合
2. $\mathcal{C}$：暗号文全体からなる有限集合
3. $\mathcal{K}$：暗号化鍵全体からなる有限集合
4. $\mathcal{E}$：各鍵 $K \in \mathcal{K}$ に対し定まる暗号化の規則全体からなる集合
   鍵 K による規則 $e_K \in \mathcal{E}$ は $\mathcal{M}$ から $\mathcal{C}$ への写像と考える。
5. $\mathcal{D}$：各鍵 $K \in \mathcal{K}$ に対し定まる復号化の規則全体からなる集合
   鍵 K による規則 $d_K \in \mathcal{D}$ は $\mathcal{C}$ から $\mathcal{M}$ への写像と考える。
6. 任意の暗号化鍵 $K \in \mathcal{K}$ と平文 $M \in \mathcal{M}$ に対し、$d_K(e_K(M)) = M$ を満たす。 ∎

前記の A さんが鍵 K_e を用いて平文 M を暗号化することは、K_e によって定まる暗号化の規則を用いて平文 M を暗号化することである。また、B さんが復号化用の鍵 K_d を用いて暗号文 C を復号化することは、K_d によって定まる復号化の規則を用いて暗号文 C を復号化することであるが、しかし、そもそも K_d は暗号化鍵 K_e に依存して定まるものであるから、K_d によって定まる復号化の規則というものは、実は暗号化鍵 K_e によって定まる復号化の規則であると考えてよいことに注意する。

上記の暗号システムの定義にしたがって前節にて説明したシフト暗号を形式的に記述してみよう。まず、準備として、以下の二点について説明する。はじめに、アルファベットの文字を数字に対応させることでシフト処理を算術演算に置き換えることを考る。アスキーコードを 10 進数に変換し、文字と数字の対応を以下のようにする。

A	B	C	D	E	F	G	H	I	J	K	L	M
65	66	67	68	69	70	71	72	73	74	75	76	77
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
78	79	80	81	82	83	84	85	86	87	88	89	90

(1)

次に、65 から 90 までの 26 個の整数の集合 $\{65, 66, \ldots, 90\}$ を $\mathbf{Z}_{26}$ で表す。そして、$\mathbf{Z}_{26}$ の中での足し算 $\oplus$ と引き算 $\ominus$ を次のように定義する。任意の $a, b \in \mathbf{Z}_{26}$ に対し、

$$a \oplus b := c \quad \text{such that} \quad a + b \equiv c \pmod{26} \text{ and } 65 \leq c \leq 90, \tag{2}$$

$$a \ominus b := c \quad \text{such that} \quad a - b \equiv c \pmod{26} \text{ and } 65 \leq c \leq 90. \tag{3}$$

この定義より、常に $a \oplus b \in \mathbf{Z}_{26}$ と $a \ominus b \in \mathbf{Z}_{26}$ が成り立つのは明らか。たとえば、$(a,b) = (70,76)$ の場合、$70 \oplus 76 = 68$. また、$(a,b) = (78,89)$ の場合、$78 \oplus 89 = 89$ となる。同様に、$(a,b) = (70,76)$ の場合、$70 \ominus 76 = 72$. また、$(a,b) = (89,78)$ の場合、$89 \ominus 78 = 89$ となる。以上より、シフト暗号を形式的に記述する：

1. $\mathcal{M} = \mathcal{C} = \mathcal{K} = \mathbf{Z}_{26}$.
2. $K \in \mathcal{K}$ と $M \in \mathcal{M}$ に対し、$e_K(M) := M \oplus K$.
3. $K \in \mathcal{K}$ と $C \in \mathcal{C}$ に対し、$d_K(C) := C \ominus K$.

ここで、シフト暗号におけるシフト数 k に対応する暗号化鍵 $K \in \mathbf{Z}_{26}$ は、$k \equiv K \pmod{26}$ を満たすものであることに注意する。

暗号システムの定義の中では特に復号化鍵 K_d についての説明はないが、シフト暗号の場合で言えば $K_d = K$ と考えてもさして問題なく、暗号化鍵と一致すると考えればよい。

例 2.4 (続き) メッセージ
WEWILLMEETATCHOFUSTATION
を数字に対応させて変換すると
87 69 87 73 76 76 77 69 69 84 65 84 67 72 79 70 85 83 84 65 84 73 79 78
となる。シフト数 $k = 5$ の場合、鍵は $K = 83$ となり、その暗号文は、
66 74 66 78 81 81 82 74 74 89 70 89 72 77 84 75 90 88 89 70 89 78 84 83
となる。ここで、各文字を数字で表現した場合に「区切り文字」として、空白を挿入していることに注意する。 ∎

2.3 二つの暗号システム

本節では、幾つかある暗号の分類の仕方の中で、暗号システムを**秘密鍵暗号システム** (または、共通鍵暗号システム) と**公開鍵暗号システム**の二つに分類する場合の説明をする。

2.3.1 秘密鍵暗号システム (または、共通鍵暗号システム)

A さんと B さんは暗号化鍵 $K \in \mathcal{K}$ を選ぶことで、K に依存した暗号化規則 e_K と復号化規則 d_K が定まる。このとき、K から d_K を導出する困難さが e_K を導出するのと同程度ならば、K を第三者に公開してしまうと暗号が安全でなくなる。したがって、このような場合、K を第三者には**秘密にする**必要があるため、このような暗号システムを**秘密鍵暗号システム** (または、共通鍵暗号システム) という。前節までに説明したシフト暗号は、この秘密鍵暗号システムの一種である。秘密鍵暗号システムの簡単な概念図は図 3 のようになる。

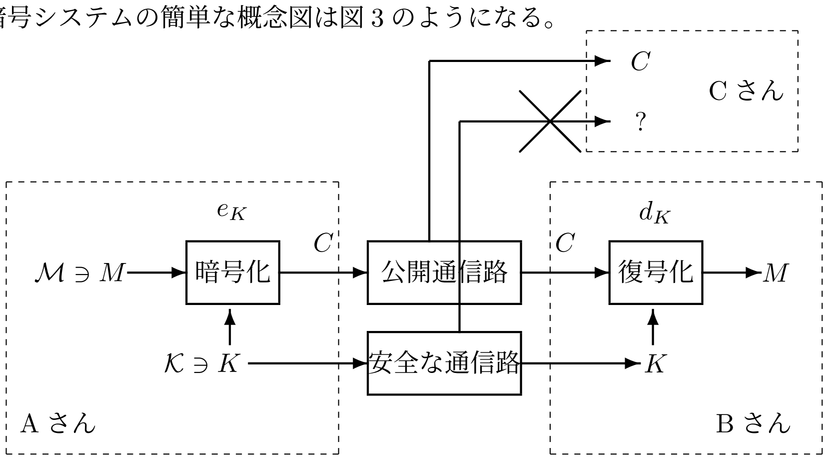


図 3: 秘密鍵暗号システム

秘密鍵暗号システムの問題点は、暗号文を送信する前に第三者には知られないように A さんと B さんの間で暗号化鍵 K を共有できるように**安全な通信路**を利用して送信する必要があることで

ある。しかし、もし、第三者には知られないように暗号化鍵 K を通信し、共有できる方法があれば、平文もそのような方法で通信すればよいことになる。現実的にはそのような安全な通信路を利用することは容易ではないと考えられている。

2.3.2 公開鍵暗号システム

暗号化鍵 K から d_K を導出する困難さが e_K を導出することと比較して計算量的に実行不可能であると考えられる場合、暗号化鍵 K を A さんと B さん以外の第三者にも**公開する**ことができる。このことで、事前に K を安全な通信路を利用して通信する必要はなくなる。一方、暗号化鍵 K から直接 d_K を導出することは容易ではない。しかし、公開された暗号化鍵 K とは異なるある秘密の情報となる**秘密鍵** K_d が存在し、その鍵 K_d を用いると比較的容易に d_K を導出することができるならば、B さんのみが秘密鍵 K_d をもつことで、暗号文を復号化することができる。このような暗号システムを**公開鍵暗号システム**という。本課題で扱う RSA 暗号は、この公開鍵暗号システムの一種である。公開鍵暗号システムの簡単な概念図は図 4 のようになる。

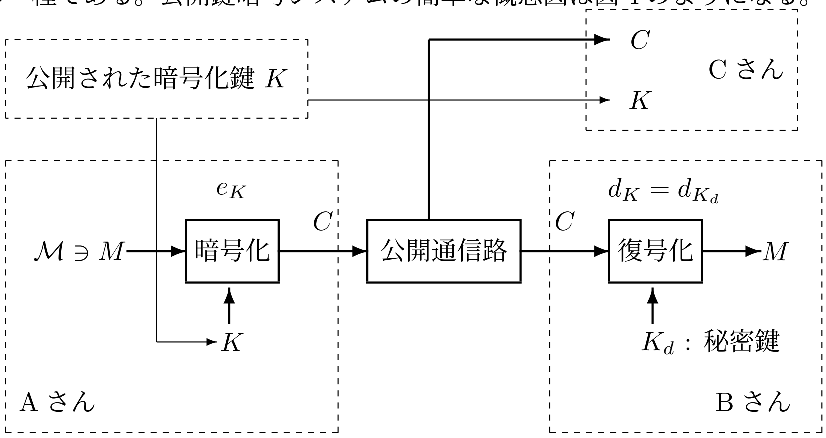


図 4: 公開鍵暗号システム

公開鍵暗号システムの説明をもう少し簡単に説明しよう。秘密鍵暗号システムでは、平文 M を暗号化する暗号化鍵 K_e と暗号文 C を復号化する復号化鍵 K_d は同じものである。そのため、A さんと B さんは鍵 $K_e = K_d$ を第三者に分からないように通信し、共有する必要があった。

一方、公開鍵暗号システムでは、平文 M を暗号化する暗号化鍵 K_e と暗号文 C を復号化する復号化鍵 K_d は異なるものと考えている。そのため、暗号化鍵 K_e を用いても暗号文 C を復号化することはできなく、復号化できるのは復号化鍵 K_d を用いた場合だけであると考えている。したがって、鍵を共有する必要はない。

そこで、暗号文を受け取る側の B さんは B さん専用の**暗号化鍵**と**復号化鍵**を作成し、暗号化鍵は**一般に公開し**、復号化鍵は**秘密**にしておく。そして、A さんは B さんに伝えたい平文を公開されている暗号化鍵を用いて暗号文を作成し、公開通信路を利用して B さんに送信すればよい。第三者の C さんは、公開されている暗号化鍵と公開通信路を利用して暗号文を入手できるが、その二つだけでは暗号文を元の平文に容易には戻せないということである。このとき、A さん自身も復号化鍵を知らないので自分で作成した暗号文を元の平文へは復元できない。したがって、A さんが作成した暗号文を復号化できるのは復号化鍵をもつ B さんのみだけである。これで暗号の目的である情報の通信が可能であると考える訳である。

3　数学的準備（整数の諸性質）

この節は、読み飛ばしても構わない。本節では、次節で説明する RSA 暗号で必要となる整数に関する諸性質について簡単にまとめたものを説明する。しかし、まずは、本節は読み飛ばし、RSA 暗号の説明で分からない数学的用語などがある場合にその都度、本節に戻って参考にすればよいと思う。

では、以下に整数に関連する事実をおさらいする。

3.1　除法の定理

定理 3.1 a, b を整数とし、$b > 0$ とする。そのとき、

$$a = bq + r, \quad 0 \leq r < b$$

を成り立たせる整数 q と r がたかだか 1 組だけ存在する。 ∎

たとえば、$(a, b) = (57, 17)$ のとき、$(q, r) = (3, 6)$ となる。また、$(a, b) = (-57, 17)$ のとき、$(q, r) = (-4, 11)$ となる。

上記の q, r をそれぞれ a を b で割った**整商**、**余り**（または**剰余**）という。詳しくは、r は a を b で割った負でない最小剰余という。$r = 0,$ すなわち $a = bq$ となる場合には、a は b で割り切れるという。

3.2　最大公約数

a, b を 2 つの整数とする。もし、$a = bq$ となる整数 q が存在するならば、a は b で割り切れる、b は a を割り切るという。またこのとき、a は b の倍数、b は a の**約数**という。このことを記号で $b|a$ と書く。

a_1, a_2 を 2 つの整数とし、0 でないとする。これらをすべて割り切る整数は、a_1, a_2 の**公約数**とよばれる。d が正の公約数で、さらに、次の性質をもつとき、d は a_1, a_2 の**最大公約数**とよばれる。

「e を a_1, a_2 の任意の公約数とすれば、$e|d$ である。」

最大公約数は（もし存在すれば）一意に定まる。それでは、最大公約数が存在することは次の定理で保証される。

定理 3.2 a_1, a_2 を 2 つの整数とし、0 でないとする。x_1, x_2 を任意の整数として

$$x_1a_1 + x_2a_2$$

の形に表される整数全部の集合を J とし、J に含まれる最小の正の元を d とする。そのとき、d は a_1, a_2 の最大公約数である。また、J は d のすべての倍数の集合と一致する。 ∎

a_1, a_2 の最大公約数を、greatest common divisor の頭文字を用いて、$\gcd(a_1, a_2)$ と表す。

系 3.3 a_1, a_2 の最大公約数を d とすれば、

$$d = u_1a_1 + u_2a_2$$

となるような整数 u_1, u_2 が存在する。 ∎

2 つの整数の最大公約数を求める方法としては、Euclid 法がある。

補題 3.4 整数 a, b の差 $a - b$ が $m(\neq 0)$ で割り切れるならば、

$$\gcd(a, m) = \gcd(b, m)$$

である。 ∎

この補題にもとづいて Euclid 法 は以下のように説明できる。

アルゴリズム 3.5 (Euclid 法) $a \geq b$ と仮定し、次のような等式の系列をつくる：

$$\begin{array}{ll} a = bq_1 + r_2, & 0 < r_2 < b \\ b = r_2 q_2 + r_3, & 0 < r_3 < r_2 \\ r_2 = r_3 q_3 + r_4, & 0 < r_4 < r_3 \\ \vdots & \vdots \\ r_{n-2} = r_{n-1} q_{n-1} + r_n, & 0 < r_n < r_{n-1} \\ r_{n-1} = r_n q_n & \end{array} \tag{4}$$

補題 3.4 より、

$$\gcd(a,b) = \gcd(b,r_2) = \gcd(r_2,r_3) = \cdots = \gcd(r_{n-1},r_n)$$

となるが、$r_n | r_{n-1}$ であるから $\gcd(r_{n-1}, r_n) = r_n$. したがって、最後の除数 r_n が a, b の最大公約数である。ゆえに、$\gcd(a,b) = r_n$ となる。これが Euclid 法 である。 ∎

整数 a, b に対して $\gcd(a,b) = 1$ であるとき、a, b は**互いに素**であるという。

例 3.6 $(a,b) = (144,5)$ のとき、以下の計算より $\gcd(144,5) = 1$ となる。

$$\begin{array}{ll} 144 = 5 \cdot 28 + 4, & 0 < 4 < 5 \\ 5 = 4 \cdot 1 + 1, & 0 < 1 < 4 \\ 4 = 1 \cdot 4 & \end{array} \tag{5}$$

また、$(a,b) = (144,60)$ のとき、以下の計算より $\gcd(144,60) = 12$ となる。

$$\begin{array}{ll} 144 = 60 \cdot 2 + 24, & 0 < 24 < 60 \\ 60 = 24 \cdot 2 + 12, & 0 < 12 < 24 \\ 24 = 12 \cdot 2 & \end{array} \tag{6}$$

∎

Euclid 法を用いることで、a, b の最大公約数 $\gcd(a,b) = d$ を求めることができた。このとき、系 3.3 より a, b, d はある整数 f, g を用いて次のような関係式で表される。

$$fa + gb = d$$

例えば、$(a,b) = (144,60)$ の場合、$\gcd(144,60) = 12$ であるが、これは、$(f,g) = (-2,5)$ として、

$$-2 \cdot 144 + 5 \cdot 60 = 12$$

と書くことができる。また、$(a,b) = (144,5)$ の場合、$\gcd(144,5) = 1$ であるが、これは、$(f,g) = (-1,29)$ として、

$$-1 \cdot 144 + 29 \cdot 5 = 1$$

と書くことができる。

整数 a, b に対して、その最大公約数と同時にこのような f, g を求める方法を**拡張 Euclid 法**という。この方法を以下に説明する：

アルゴリズム 3.7 (拡張 Euclid 法) アルゴリズムへの入力は、a, b である。ただし、$a > b$ とする。初期値として、

$$\begin{array}{rcl} r_{-1} & := & a,\ f_{-1} := 1,\ g_{-1} := 0 \\ r_0 & := & b,\ f_0 := 0,\ g_0 := 1 \end{array}$$

とする。

このとき、各 $i \geq 1$ に対して、r_{i-2} を r_{i-1} で割ったときの整商 q_i と余り r_i を計算する：

$$r_{i-2} = r_{i-1} q_i + r_i, \quad 0 < r_i < r_{i-1}$$

そして、f, g を更新する：

$$\begin{aligned} f_i &:= f_{i-2} - q_i f_{i-1} \\ g_i &:= g_{i-2} - q_i g_{i-1}. \end{aligned}$$

r_i は単調減少しているために必ず終了が来る。ここで、$r_n \neq 0$ とする。このとき、$\gcd(a, b) = r_n$ と $f_n a + g_n = r_n$ を得ることができる。 ∎

例 3.8 $(a, b) = (144, 60)$ の場合：

i	f_i	g_i	r_i	q_i
-1	1	0	144	$-$
0	0	1	60	$-$
1	1	-2	24	2
2	$\underline{-2}$	$\underline{5}$	$\underline{12}$	2
3	5	-12	0	2

(7)

したがって、$\gcd(144, 60) = 12$ と $f_2 a + g_2 b = d \to \underline{-2} \cdot 144 + \underline{5} \cdot 60 = \underline{12}$ を得る。 ∎

例 3.9 $(a, b) = (144, 5)$ の場合：

i	f_i	g_i	r_i	q_i
-1	1	0	144	$-$
0	0	1	5	$-$
1	1	-28	4	28
2	$\underline{-1}$	$\underline{29}$	$\underline{1}$	1
3	5	-144	0	4

(8)

したがって、$\gcd(144, 5) = 1$ と $f_2 a + g_2 b = d \to \underline{-1} \cdot 144 + \underline{29} \cdot 5 = \underline{1}$ を得る。 ∎

3.3 素数

a を 1 より大きい整数とすれば、a は少なくとも 2 つの正の約数 1 と a をもつ。a がこれら以外に正の約数をもたないとき、a を**素数** (prime number) という。たとえば、$2, 3, 5, 7, 11, 13, \ldots$ である。素数でない整数 ≥ 2 は合成数とよばれる。

命題 3.10 B を整数 > 1 とする。そのとき、任意の正の整数 a は、

$$a = c_k B^k + c_{k-1} B^{k-1} + \cdots + c_1 B^1 + c_0 B^0 \tag{9}$$

$0 \leq c_0 < B$, $0 \leq c_1 < B$, ...,$0 \leq c_k < B$, の形に一意に表される。この表現を a の B **進展開**という。 ∎

3.4 合同関係

整数全体の集合 $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$ を $\mathbf{Z}$ で表す。同値関係の具体例となる $\mathbf{Z}$ における合同関係について説明する。

m を 1 つの与えられた正の整数とする。a, b を 2 つの整数とするとき、もし $a - b$ が m で割り切れるならば、a, b は m を**法** (modulo) として（あるいは法 m に関して）合同であるという。このことを記号で

$$a \equiv b \pmod{m}$$

と書く。

次に、m を法とする合同関係による $\mathbf{Z}$ の類別について考える。この場合の各類は法 m に関する剰余類とよばれる。1つの剰余類は m を法として互いに合同な数全体の集合である。すなわち、a をその剰余類の代表とすれば、$a+mt$ の形に表される整数全体の集合である。

任意の整数 a は

$$a = mq + r, \quad 0 \leq r < m$$

の形に一意に表される。したがって、a は $0 \leq r < m$ であるような1つしかもただ1つの r と、m を法として合同となる。いいかえれば、任意の整数 a は、それぞれ $0, 1, \ldots, m-1$ を代表とする m 個の剰余類のいずえれか1つ、しかもただ1つだけに含まれる。ゆえに、法 m に関する剰余類は全部でちょうど m 個存在する。

例 3.11 $m = 3$ とすれば、法3に関して $\mathbf{Z}$ は3個の剰余類に分割される。それらは、3 の倍数全部の集合、3で割ると1余る数全体の集合、3で割ると2余る数全体の集合である。 ∎

定理 3.12 m_1, m_2 を互いに素な正の整数とし、b_1, b_2 を任意の整数とする。そのとき、連立合同式

$$x \equiv b_1 (\mathrm{mod}\ m_1) \tag{10}$$

$$x \equiv b_2 (\mathrm{mod}\ m_2) \tag{11}$$

は、積 $m = m_1 m_2$ を法としてただ1つの解をもつ。 ∎

3.5 剰余環（商環） $\mathbf{Z}_m$

整数 m に対し，集合 $\mathbf{Z}_m$ を $\{0, 1, 2, \ldots, m-1\}$ と定義する．このとき，集合 $\mathbf{Z}_m$ 上に m を法とする加法 $+$ と乗法 $\times$ を定義することで，集合 $\mathbf{Z}_m$ は、 0 を零元、1 を単位元とする m 個の元からなる有限環となる．

4 RSA 暗号

本節では、RSA 暗号について説明する。はじめに、次のようないくつかの整数を選択したり、計算したりする。

1. 任意の互いに異なる素数 p, q を選び、その積を計算する。

$$n := pq \tag{12}$$

2. $(p-1)$ と $(q-1)$ の最小公倍数 (least common multiple) L を計算し、L と互いに素で L より小さな任意の整数 e を選ぶ。

$$L := \mathrm{lcm}(p-1, q-1) \tag{13}$$
$$\gcd(e, L) = 1, \quad 1 < e < L \tag{14}$$

3. 次式を満たす整数 d_e を計算する。

$$ed_e \equiv 1 \pmod{L} \tag{15}$$

このような d_e を求める方法として拡張 Euclid 法を利用すればよい。

このとき、RSA 暗号の形式的な記述は以下のようになる：

1. 任意の互いに異なる素数 p, q に対し、 $n := pq$ とする。
2. $\mathcal{M} = \mathcal{C} = \mathbf{Z}_n$.
3. $\mathcal{K} = \{(e, n) \mid \gcd(e, L) = 1 \text{ and } 1 < e < L\}$ where $L = \mathrm{lcm}(p-1, q-1)$.
4. d_e： $ed_e \equiv 1 \pmod{L}$ を満たす整数。
5. $K = (e, n) \in \mathcal{K}$ と $M \in \mathcal{M}$ に対し、$e_K(M) := M^e \pmod{n}$.
6. $K = (e, n) \in \mathcal{K}$ と $C \in \mathcal{C}$ に対し、$d_K(C) := C^{d_e} \pmod{n}$.
7. 暗号化鍵 (e, n) を公開し、p, q, d_e の値を秘密にする。d_e が秘密鍵で (d_e, n) が復号化鍵となる。

以上を図示したものが、図 5 となる。

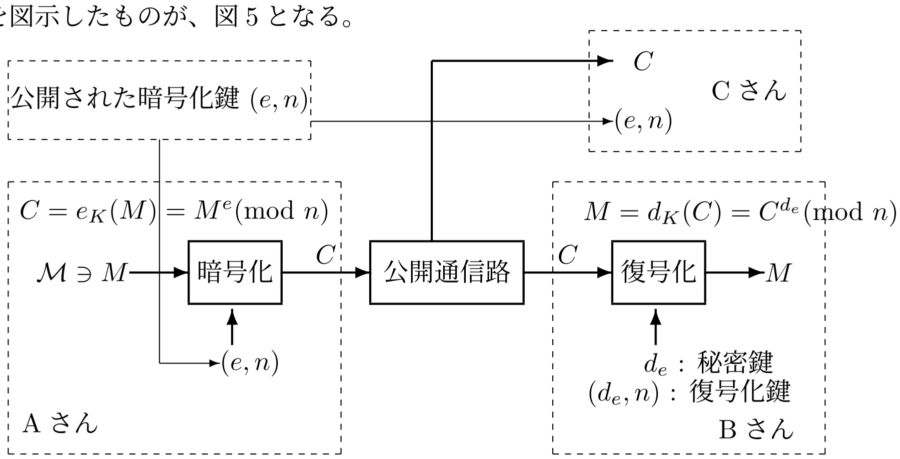


図 5: RSA 暗号を用いた公開鍵暗号システム

RSA 暗号の暗号化規則 e_K と復号化規則 d_K の合成写像は、$d_K(C) = d_K(e_K(M)) = M$ となる。つまり、

$$C^{d_e} = (M^e)^{d_e} = M^{ed_e} \equiv M \pmod{n} \tag{16}$$

RSA 暗号において、この関係の成立が保証されていることが大切である。

公開されている情報 (e,n) から整数 d_e を求めることで RSA 暗号を解読することを考える。d_e の定義からその値を容易に求めるには二つの素数 p,q の値を求める必要がある。そのためには整数 n を素因数分解すればよい。しかし、この整数 n の素因数分解を完了することは計算量的に実行不可能であるという考えに基づくと、整数 d_e を求めることで RSA 暗号を解読することは困難であるというになる。したがって、d_e を求めるという解読において、RSA 暗号の安全性は、 素因数分解の困難さに基づいているといえる。

例 4.1 先の例で取り上げた 26 文字のアルファベットからなる平文を RSA 暗号で暗号化することを考える。

文字数は 26 個であるから選択すべき二つの素数 p,q は、$n = pq \geq 26$ を満たす必要がある。そこで、$(p,q) = (17,19)$ とすることで、$n = 323$ となる。

次に、$(p-1)$ と $(q-1)$ の最小公倍数は、$L = 144$ となる。$L = 144$ に対し、$1 < e < 144 = L$ を満たし、互いに素なる整数 e として 5 を選択する。

そして、拡張 Euclid 法を利用し、秘密鍵 d_e を計算すると $d_e = 29$ となる。

以上より、公開鍵（符号化鍵）は $K = (e,n) = (5,323)$, 秘密鍵は $d_e = 29$, 復号化鍵は $(d_e,n) = (29,323)$ となる。

たとえば、文字 “C” に対応する数字は “67” であるから、これを暗号化すると、$(67)^5 \equiv 288 \pmod{323}$ より、$e_K(67) = 288$ となる。一方、これを復号化すると、$(288)^{29} \equiv 67 \pmod{323}$ より、$d_K(288) = 67$ となり、確かにもとの平文 “C” と一致する。

このとき、$(67)^5$ を一度に計算する必要はなく、以下のように計算することが可能である。

1) $(67)^2 = 4489 \equiv 290 \pmod{323}$,
2) $(67)^3 \equiv 290 \times 67 = 19430 \equiv 50 \pmod{323}$,
3) $(67)^4 \equiv 50 \times 67 = 3350 \equiv 120 \pmod{323}$,
4) $(67)^5 \equiv 120 \times 67 = 8040 \equiv 288 \pmod{323}$.

つまり、$\pmod{323}$ において扱う整数は 0 から $322 \times 322 = 103684$ までで十分である。ゆえに、103684 以下の整数計算を正しくできる電卓があれば一応手計算でも暗号化を行なうことができるということ。計算量を考慮した場合、べき乗の計算には、例 4.2 に示すような工夫をすることができる。

それでは、メッセージ WEWILLMEETATCHOFUSTATION を暗号化する。まず、各文字を数字に変換すると

87 69 87 73 76 76 77 69 69 84 65 84 67 72 79 70 85 83 84 65 84 73 79 78

であるから、二桁ずつ数字を取り出して暗号化の計算をすると暗号文

83 103 83 99 247 247 229 103 103 50 12 50 288 21 129 185 187 87 50 12 50 99 129 10

を得る。ここで、各文字を数字で表現した場合に「区切り文字」として、空白を挿入していることに注意する。さて、A さんは、この暗号文を B さんに送信すればよい。そして、 B さんは自分しか知らない秘密鍵を含む復号化鍵 $(d_e,n) = (29,323)$ を用いて暗号文を復号化する。 ∎

例 4.2 (二進展開法) たとえば、2 の 13 乗のべき冪, 2^{13}, の計算を素直に実行すれば、$2\times 2\times \cdots \times 2 = 8192$ のように 12 回の乗算を実行する必要がある。ここでは、なるべく少ない乗算回数でべき乗計算の結果を得る一方法について説明する。

まず、13 を二進展開すると $1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0$ より、1101 となる。上位桁を左側に記すように 0 と 1 の系列を書いている。

そこで、系列の左側から **2 番目**の値から順に右側方向に値を読み取っていく。このとき、変数 x を次のように更新していく。ただし、x の初期値は 1 とする。系列の値が 1 ならば $x := 2x+1$、そうでなく値が 0 ならば $x := 2x$ とする。具体的に、変数 x は $1 \to 3 \to 6 \to 13$ と変化する。

$$\begin{array}{ccccccc} & & 1 & & 0 & & 1 \\ 1 & \to & 3 & \to & 6 & \to & 13 \end{array} \tag{17}$$

この結果を利用して、べき冪, 2^{13}, の計算を次のように行なう。

$$\begin{aligned} 2^2 \times 2 (= 2^{1\times 2+1}) &= 8 \\ 8^2 (= 2^{3\times 2}) &= 64 \\ 64^2 \times 2 (= 2^{6\times 2+1}) &= 8192 \end{aligned}$$

これより、乗算は 5 回で十分であることが分かる。以上を簡単にまとめると以下のように書ける。
整数 M の d 乗, M^d, を考える。整数 d の二進展開を $b_k b_{k-1} \cdots b_1 b_0$ とする。ただし、$b_k = 1$.

1. 初期値 $i := k - 1$, $x := M$.
2. $b_i = 0$ ならば $x := x^2$、さもなければ $x := x^2 \times M$ を計算する。
3. $i = 0$ ならば x を出力、さもなければ $i := i - 1$ としてステップ 2 へ戻る。

∎

$(p, q) = (17, 19)$ 程度の大きさの値でも、暗号化や復号化の計算をすべて電卓で計算するのは容易ではない。そこで、本課題の目的の一つとして、RSA 暗号の暗号化および復号化プログラムを書くことにする。

5　課題

RSA 暗号の理解と実装に関連する実験テーマとして以下の 5 個の課題を必修とする。

1. RSA 暗号の暗号化および復号化のプログラムを作成し、実装せよ。具体的には, 以下のような暗号化および復号化の仕様を満たすこと。

   暗号化:

入力	：暗号化鍵 (e, n) と ひら文のファイル.
出力	：暗号化されたファイル.

   復号化:

入力	：復号化鍵 (d_e, n) と 暗号化されたファイル.
出力	：元のひら文のファイル.

   このとき、ひら文のファイルと暗号化されたファイルのサイズを比較せよ。

   本資料の 15 ページの 6.2 節 参考プログラム の 1 「RSA 暗号の暗号化（符号化）と復号化プログラム（rsa.c）」の実装内容を参考にせよ.

2. 1 から 10000 までの間に存在する素数をすべて求めるプログラムを作成し、実装せよ。

3. 異なる素数 p と q に対し、$p-1$ と $q-1$ の最小公倍数 $L=\mathrm{lcm}(p-1, q-1)$ を求めるプログラムを作成し、実装せよ。

4. 整数 L に対し、$\gcd(e, L) = 1$ かつ $1 < e < L$ を満たす正整数 e を求めるプログラムを作成し、実装せよ。

5. 整数 L と e に対し、$ed \equiv 1 \pmod{L}$ を満たす正整数 d を求めるプログラムを作成し、実装せよ。

6　プログラミング

6.1　プログラミングの準備

計算機上でプログラミングを行なうことで RSA 暗号の暗号化・復号化を実装させることを考える。ここでは、32 ビット計算機と C 言語を利用するという立場で、プログラミングの方針および注意事項などを箇条書の形で列挙する。また、多倍長計算に関してはここでは特に扱わないとする。

1. ここでの目標は、次の 2 つに大別される。一つ目は、暗号化鍵を受理した後、”foo” というファイル名の平文のデータを RSA 暗号で暗号化し、その暗号文を”goo” というファイル名のファイルに書き出す暗号化プログラムを作成すること。二つ目は、復号化鍵を受理した後、暗号化されたデータのファイル”goo” を復号化し、もとの平文のデータに戻し、 “hoo” というファイル名のファイルに書き出す復号化プログラムを作成すること。このとき、二つのファイル foo と hoo が一致することを diff コマンドで確認する。ここで、ファイル名に意味はない。

2. 暗号化の具体的な方法としては、foo というファイルから１バイト毎にデータを読み取り、暗号化し、goo というファイルに暗号文を書き出す。これをファイル foo の EOF を読み取るまで繰り返し、終了する。このとき、１バイトは８ビットの０と１の列である。そこで、１バイトのデータを文字として処理するのではなく、それらを 0 から $255 = 2^8 - 1$ までの整数の平文として処理することで暗号化を行なうことにする。

3. したがって、平文 M は $0 \leq M \leq 255$ であるから、$255 < n = pq$ を満たすような素数 p, q を選択する必要がある。

4. もし、$n > 255$ ならば、暗号文 C は 256 の以上の整数値をとることになる。すると、1バイトの平文 M を暗号化すると 1バイトでは表現できない整数になり、暗号文 C を記録するには2バイト以上の枠を考える必要がある。

   例えば、先の例 4.1 において、文字 “C” に対応する 67 は、$e_K(67) = 288$ となる。整数 288 を表すには、2 バイトが必要である。

5. 具体的にプログラミングを行なうには、利用する 32 ビット計算機で一般的に扱える整数の大きさを考慮する必要がある。C 言語での整数を扱う型 int, long では、正の整数は $2^{31} - 1$ までしか正しく扱うことができない。つまり、32 ビットのうち上位 1 ビットを符号用に利用するため符号以外の部分で利用できるのは 31 ビットである。

6. 復号化のことを考えた場合、少なくとも n^2 以下の数を正しく扱う（計算する）必要がある。大雑把に計算すると、$n \leq 2^{15} - 1$, すなわち、n が 15 ビット以内で表現できる正の整数ならば、$n^2 < 2^{30} < 2^{31} - 1$ を満たす。

7. したがって、$2^8 = 256 \leq n \leq 2^{15} - 1$ を満たすように p, q を選択すればよさそうである。

   たとえば、$(p, q) = (29, 31)$ とすると、$2^4 < 29 < 2^5$ かつ $2^4 < 31 < 2^5$ であるから、$2^8 < 2^9 = 512 < n = pq = 899 < 1024 = 2^{10}$ となる。したがって、プログラミングの具体的な (p, q) の例として $(29, 31)$ を利用することができる。一方、$(p, q) = (241, 251)$ とすると、$2^7 < 241 < 2^8$ かつ $2^7 < 251 < 2^8$ であるから、$2^{14} < 2^{15} = 32768 < n = pq = 60491 < 65536 = 2^{16}$ となる。ゆえに、プログラミングの具体的な (p, q) の例として $(241, 251)$ を利用することは適切ではない。

8. たとえば、$(p, q) = (29, 31)$ を選択した場合、$n = pq = 899 > 2^9$ であるから1バイトの平文 M に対する暗号文 C は少なくとも2バイトないと表現できない数である。ここでは、プログラミングを容易にするためにデータの取扱いをビット単位ではなくバイト単位で扱うことを考えている。一方、2バイトあれば十分でもある。このとき、暗号化プログラムは1バイトの平文を読み込み、暗号化する毎に常に2バイトの暗号文を書き出すようにすると考える。このように考えた場合、復号化プログラムは、暗号文のファイルから2バイト単位でデータを処理するように工夫し、それを暗号文 C として処理し、復号化を実行すればよい。

9. C 言語での fgetc や fputc の関数では、データを1バイト単位でしか処理することができない。そこで、前記のように2バイト以上で表現されるデータをファイルの入出力で扱うアイデアとして B 進展開するという方法が考えられる。B 進展開については数学的準備の 3.3 節を参照。

   例えば、先の例 4.1 において、文字 “C” に対応する 67 は、$e_K(67) = 288$ となる。この整数 288 を表すには、2 バイトが必要であるが、これを次のように分解し、1バイト単位で処理できるようにする。すなわち、$B = 256$ とする。$288 = c_1 \times (256)^1 + c_0 \times (256)^0$ を満たす $(c_1, c_0) = (1, 32)$ の各 c_i は、 255 以下の整数である。したがって、それらは1バイトで表現でき、288 に対応するデータとしては、1 と 32 を暗号文のファイルに保存すればよい。復号する場合は、暗号文のファイルから 1 と 32 を得てから、逆の計算をし、288 を求めた後に、$e_K(288) = 67$ を計算すればよい。

6.2 参考プログラム

本節では、本課題のプログラムを作成するに当たり参考になると思われるプログラムを示す。プログラムの実行ファイル (ただし、ied でコンパイルしたもの) およびソースファイルの PS または PDF ファイルを
`http://www.lit.ice.uec.ac.jp/kuri/C3/`
に置くので自由に利用して構わない。以下に, プログラムの実行方法とその内容を記す。同時にコンパイルの様子も記載しておく。“ IED: ” はコマンドラインでのプロンプトである。

1. RSA 暗号の暗号化（符号化）と復号化プログラム（rsa.c）

   前節のプログラミングの準備で挙げた各事項に従って作成したプログラムである。

まず、暗号化するために「1」を選択し、公開鍵「$(5, 323)$」を入力する。続いて、平文のファイル名「foo」と暗号文を出力するファイル名「goo」を入力する。この入力により、foo の中身は暗号化され、その暗号文が goo に書き出される。ここでは、$(p, q) = (17, 19)$ として、$n = 323$ を得ている。

次に、復号化するために「2」を選択し、復号化鍵「$(29, 323)$」を入力する。続いて、暗号文のファイル名「goo」と復号化後の平文を出力するファイル名「hoo」を入力する。この入力により、goo の中身は復号化され、その内容が hoo に書き出される。

最後に、diff コマンドでファイルが一致することを確認する。さらに、ファイルサイズも確認する。

```
IED: gcc -o rsa rsa.c
IED: more foo
WEWILLMEETATCHOFUSTATION
IED: rsa
     This is RSA-cipher program.
     Now would you please select 1.encoding or 2.decoding?
     Select the number 1 or 2. : 1

...start encoding for RSA-cipher.
 1. public-key (e,n)?  Input e and n such as 5 323. : 5 323
 2. input  file name? : foo
 3. output file name? : goo
... end of encoding for RSA-cipher.
IED: more goo
SgScgg2^L2 W2^L2cl
IED: rsa
     This is RSA-cipher program.
     Now would you please select 1.encoding or 2.decoding?
     Select the number 1 or 2. : 2

...start decoding for RSA-cipher.
 1. secret-key (d,n)?  Input e and n such as 29 323.: 29 323
 2. input  file name? : goo
 3. output file name? : hoo
... end of decoding for RSA-cipher.
IED: more hoo
WEWILLMEETATCHOFUSTATION
IED: diff foo hoo
IED: ls -la | grep oo
-rw-r--r--   1 kuri     teacher       25  5月 13日  12:20 foo
-rw-r--r--   1 kuri     teacher       50  5月 13日  12:21 goo
-rw-r--r--   1 kuri     teacher       25  5月 13日  12:21 hoo
IED:
```

2. ファイルをコピーするプログラム (fcopy.c)

   このプログラムは、参考文献に示した 椋田實 著の「はじめての C」 p.247 に記載されているものを変数名などのみを修正したものである。

   このプログラムは、入力ファイルから1バイト単位でデータを読み取り、その読み取ったデータを1バイト単位で出力ファイルに書き出すプログラムである。

```
IED: gcc -o fcp fcopy.c
IED: more foo
WEWILLMEETATCHOFUSTATION
IED: fcp
...start fcopy.
input file name? foo
output file name? goo
...end of fcopy.
IED: diff foo goo
IED: more goo
```

```
WEWILLMEETATCHOFUSTATION
IED:
```

3. ファイルデータの表示プログラム (fdisp.c)

   このプログラムは、入力ファイルから 1 バイト単位でデータを読み取り、そのデータを「文字型」と「整数型 (10 進数)」として、"文字 $<-->$ 10 進数" のように表示するプログラム。つまり、文字とその ASCHII コードを表示する。

```
IED: gcc -o fdisp fdisp.c
IED: more foo
WEWILLMEETATCHOFUSTATION
IED: fdisp
...start fdisp.
 input file name? foo
----------------
W <--> 87
E <--> 69
W <--> 87
I <--> 73
L <--> 76
L <--> 76
M <--> 77
E <--> 69
E <--> 69
T <--> 84
A <--> 65
T <--> 84
C <--> 67
H <--> 72
O <--> 79
F <--> 70
U <--> 85
S <--> 83
T <--> 84
A <--> 65
T <--> 84
I <--> 73
O <--> 79
N <--> 78

 <--> 10
----------------
...end of fdisp.
IED:
```

4. 文字の変換プログラム (fchg.c)

   このプログラムは、入力ファイル中のアルファベットについて、大文字を小文字に、小文字を大文字に変換（交換）し、出力ファイルにその結果を書き出すプログラム。ここで、アルファベット以外のものは変換しない。

```
IED: gcc -o fchg fchg.c
IED: more title
A Method for Obtaining Digital Signatures and Publick-Key Cryptosystems
IED: fchg
...start fchg.
 input  file name? : title
 output file name? : goo
.. end of fchg.
IED: more goo
a mETHOD FOR oBTAINING dIGITAL sIGNATURES AND pUBLICK-kEY cRYPTOSYSTEMS
IED:
```

5. 拡張 Euclid 法のプログラム (euclid2.c)

下記に示すように、入力された二つの整数 $(a, b) = (144, 5)$ に対する最大公約数 d を求めると同時に式 $fa + gb = d$ を満たす (f, g) も求める。RSA 暗号に置き換えると、暗号化用の鍵の一部 e に対応させ、$(L, e) = (144, 5)$ を入力すると、復号化用の秘密鍵が $d_e = 29$ と求まることが分かる。

```
IED: gcc -o euc euclid2.c
IED: euc
...start Euclid algorithm for finding gcd(a,b).
 Input two numbers a and b where a > b such as 144 5 : 144 5
-----------------------------------------------------------
  i :        f_i         g_i         r_i         q_i
-----------------------------------------------------------
 -1 :          1           0         144          -
-----------------------------------------------------------
  0 :          0           1           5          -
-----------------------------------------------------------
  1 :          1         -28           4          28
-----------------------------------------------------------
  2 :         -1          29           1           1
-----------------------------------------------------------
  3 :          5        -144           0           4
-----------------------------------------------------------

          d = gcd(a,b) = gcd(       144 ,          5) =          1

          ( f * a) + (g * b) = d

  (        -1 *        144)+(        29 *          5) =          1

          (f,g)=(        -1,         29)

...end of Euclid algorithm.
IED:
```

6. シフト暗号の暗号化・復号化プログラム (shift.c)

下記に示すように、暗号化として、鍵となる k の値を入力し、入力ファイル foo を暗号化し暗号文をファイル goo に書き出す。また、復号化として、復号鍵となる $-k$ の値を入力し、入力ファイル goo を復号化し元の平文と同じ内容をファイル hoo に書き出す。

```
IED: gcc -o sft shift.c
IED: more foo
WEWILLMEETATCHOFUSTATION
IED: sft
...start SHIFT-cipher.
 1. If you want to encode input-file then input k, otherwise( decoding ) -k.?
    shift size k ?    : 5
 2.  input-file name? : foo
 3. output-file name? : goo
.. end of SHIFT-cipher.
IED: more goo
BJBNQQRJJYFYHMTKZXYFYNTS
IED: sft
...start SHIFT-cipher.
 1. If you want to encode input-file then input k, otherwise( decoding ) -k.?
    shift size k ?    : -5
 2.  input-file name? : goo
 3. output-file name? : hoo
.. end of SHIFT-cipher.
IED: diff foo hoo
IED: more hoo
WEWILLMEETATCHOFUSTATION
IED:
```

参考文献

[1] Douglas R. Stinson, 暗号理論の基礎，櫻井幸一監訳、共立出版，1996.

[2] 池野信一, 小山謙二, 現代暗号理論（電子情報通信学会）、コロナ社，1986.

[3] 松坂和夫、代数系入門（第 28 刷発行）、岩波書店、2003.

[4] 石田信、代数系入門（第 12 刷発行）、実教出版、1990.

[5] R.L.Rivest, A. Shamir and L. Adleman, "A Method for Obtaining Digital Signatures and Publick-Key Cryptosystems," *Communications of the ACM*, 21(1978), pp.120–126, 1978.

[6] 椋田 (むくだ) 實, はじめての C (改訂第三版), 技術評論社, 1995(平成 7 年).

[7] 平林雅英，ANSI C 言語辞典 (初版 第 10 刷発行), 技術評論社, 2000(平成 12 年).

ASCHII

ASCII(アスキー) とは、American Standard Code for Information Interchange(情報交換用米国標準符号) の略。その中身は、1963 年にアメリカ規格協会 (ANSI: American National Standard Institute) が定めた、7/8 ビット英数字のコード体系の 1 つ。7 ビット版は、128 種類のローマ字、数字、記号、制御コードで構成されている。実際にはコンピュータは 1 文字を 8 ビット (1 バイト) で表現するため、256 種類の文字を扱うことができるが、ASCII が定めていない 128 文字分の拡張領域には、コンピュータメーカーや国によって異なる文字が収録されている。日本では、拡張領域にカナ文字を収録したコード体系が JIS X 0201 として規格化されている。

下記に、7 ビットコードを上位 3 ビットと下位 4 ビットに分けて 16 進表示したものを示す。たとえば、大文字の「Z」は、16 進表示では 5A、2 進表示では 1011010 となる。

ASCHII

下位 \ 上位	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	‘	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	”	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAC	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	’	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF/NL	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

[2] ©電気通信大学 情報通信工学科 栗原正純 (e-mail: kuri@ice.uec.ac.jp), 2005.(2005/05/10)
本資料は参考文献欄に挙げた文献をもとに作成したものである。

制御符号の説明

制御符号	コード	制御符号名	意味
NUL	00	Null	空
SOH	01	Start of Heading	ヘディング開始
STX	02	Start of Text	テキスト開始
ETX	03	End of Text	テキスト終了
EOT	04	End of Transmission	伝送終了
ENQ	05	Enquiry	問い合わせ
ACK	06	Acknowledge	肯定応答
BEL	07	Bell	ベル
BS	08	Backspace	バックスペース（1 文字後退する）
HT	09	Horizontal Tablation	水平タブ
LF/NL	0A	Line Feed/New Line	改行／復行（復帰・改行）
VT	0B	Vertical Tablation	垂直タブ
FF	0C	Form Feed	改頁
CR	0D	Carriage Return	復帰
SO	0E	Shift Out	シフト・アウト
SI	0F	Shift In	シフト・イン
DLE	10	Data Link Escape	データ・リンクでの拡張
DC1	11	Device Control 1(X-ON)	装置制御 1（送信を開始する要求に使用）
DC2	12	Device Control 2	装置制御 2
DC3	13	Device Control 3(X-OFF)	装置制御 3（送信を止める要求に使用）
DC4	14	Device Control 4	装置制御 4
NAC	15	Negative Acknowledge	否定応答
SYN	16	Synchronous Idle	同期文字
ETB	17	End of Transmission Block	伝送ブロック終了
CAN	18	Cancel	取り消し
EM	19	End of Medium	媒体終端
SUB	1A	Substitute Character	（CP/M でファイルのデータ終了記号に使用している）
ESC	1B	Escape	拡張（画面やグラフィックなどの制御コードの拡張に使用している）
FS	1C	File Separator	ファイル・セパレイタ
GS	1D	Group Separator	グループ・セパレイタ
RS	1E	Record Separator	レコード・セパレイタ
US	1F	Unit Separator	ユニット・セパレイタ
SP	20	Space	空白、ブランク、スペース
DEL	7F	Delete	抹消

参考文献

[1] http://e-words.jp/w/ASCII.html

[2] http://www.psl.ne.jp/perl/pdojo00b.html

[3] http://www.komonet.ne.jp/~perl/chap13.htm

[4] 椋田 (むくだ) 實, はじめての C (改訂第三版), 技術評論社, 1995(平成 7 年).

[5] 平林雅英，ANSI C 言語辞典 (初版 第 10 刷発行), 技術評論社, 2000(平成 12 年).