暗号理論概説

＜第 1 回＞
暗号の歴史と現状について（理論と現実との中間点から）

上野正樹

暗号という言葉を聞いてまず思い浮かべることといえば，探偵小説のような謎解きのイメージ，もしくはエニグマ暗号やむらさき暗号（パープル暗号）といった戦時における機密通信手段を思い浮かべるのではないだろうか．

2015 年に『イミテーション・ゲーム』という映画が公開された．第 2 次世界大戦時におけるドイツのエニグマ暗号解読をめぐるドラマなのだが，その主人公はアラン・チューリングという数学者である．アカデミー脚本賞を受賞するなど話題を集めた．詳しい内容の記述はさけるが，数学者アラン・チューリングは現代におけるコンピュータの原型を作成し，エニグマ暗号の解読に成功する．コンピュータの登場は人類にとってはもちろん，暗号の世界にとっても，非常に意味のある出来事である．暗号解読に数学的な理論を利用する．それは解読のみならず暗号を作成することにも大きな働きかけがあった．現代の暗号（理論）を理解するためには，数学とコンピュータは切り離せないものになっているからだ．そこで第 1 章では，エニグマ暗号以外の過去の暗号システムと，それらの暗号のどこに問題があるのかを，コンピュータや数学の視点から記述してみたい．そして第 2 章では，簡単にエニグマ暗号の仕組みなどを記述していく．

ここ最近のセキュリティに関する報道を見ると，2015 年 6 月に起きた年金情報漏えい問題，ベネッセコーポレーションにおける顧客情報漏えい問題，アメリカにおいて FBI が Apple 社に対して iPhone に関するセキュリティロックの解除を要請，拒否されたというような報道など，多数あった．スマートフォンの普及や日本におけるマイナンバー制度の運用などにより，今後ますますセキュリティの重要性は高まっていくことが予想される．九州大学では『先進数理暗号デザイン室』という暗号理論を中心に研究する部門を開室[*1] し，目覚ましい成果をあげている．また大学のみならず，暗号理論を研究している民間企業も多く，もちろん Cryptrec などの国の機関でも調査されている．そこで第 3 章では，今現在使われている暗号の仕組み（理論）や安全性評価などの概略を，RSA 暗号を中心に説明したい．数学的な記述もあるが，全体としてその暗号のイメージがわかるようにしていく．そして第 4 章では，第 3 章とはまた別に，現在の暗号全般の対する攻撃手段について，い

[*1] http://imi.kyushu-u.ac.jp/~lmdac/index.html

くつか選んで記述してみたい.

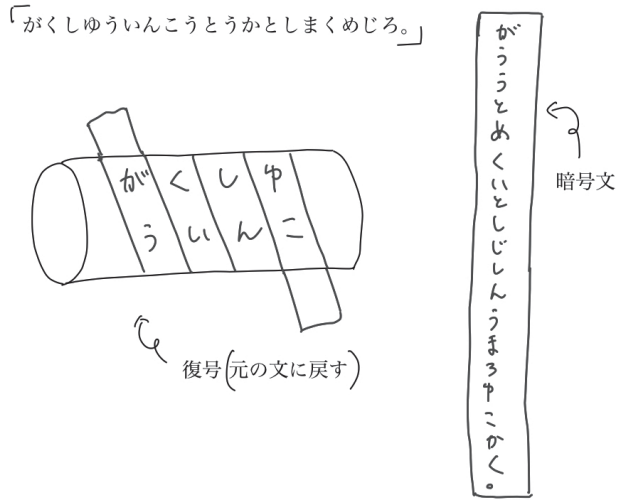
そして第 5 章では，現在の暗号がいつまで使えると考えられているか，将来的に暗号はどのようになっていくのかを，現状から推察される範囲で記述してみたい.

全 3 回を予定しているうちの本稿では，参考文献もなるべく新しいものを中心に選び，現在の状況をなるべくわかりやすく記述することを目的とした．次回以降は暗号に対する数学的な理論を中心に論じる予定である.

1　暗号の起源と探偵小説における暗号

1.1　換字式暗号とコンピュータの関係

紀元前 19 世紀の古代ギリシャの神聖象形文字『ヒエログリフ』の中にも標準以外の文字列が発見されており，暗号なのではないかと言われているが，はっきり暗号とわかる形では紀元前 5 世紀の古代スパルタにおける『スキュタレー暗号』がある．当時は革紐などを棒に巻き付け，文字を書き，ほどいて暗号文にしていた．受け取った解読者も同じ太さの棒に巻き付けて，元の文を得ることになる.



上のように，文字の位置を変える暗号のことを『転置式暗号』と呼ぶ．また，暗号の世界では，暗号化する･暗号を解読するために必要な仕組みや道具のことを**鍵**という．スキュタレー暗号では，現代人の感覚では仮に暗号解読のための鍵（この場合は『棒』）が無かったとしても，容易に解読されてしまうだろう（何文字か置きに読めばいい）．もう少し複雑な暗号として有名なものは，紀元前 2 世紀の『シーザー暗号』がある．これは古代ローマの将軍ジュリアス・シーザー（カエサル）が使用したことからこの名前で呼ばれる，の

ちに記述する**換字式暗号**の代表例である．

平アルファベット	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
暗号アルファベット	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 a

上のように，いくつかアルファベットをずらして，a を b，b を c，などのようにしていく．例えば上の変換であれば，

$$\text{night} \quad \Longrightarrow \quad \text{ojhiu}$$

と暗号化されるし，もし eph という暗号文を受け取れば，変換を逆から見れば dog と元の文に戻すことができる．いくつずらすかは任意に設定できるが，このように単にいくつかずらす方法は 26 通りしか存在しないため，多かれ少なかれ短時間で解読されてしまうことになるだろう．

次に考えることは，アルファベットをランダムに変換していく考えで，例えば

平アルファベット	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
暗号アルファベット	j l p a　w i q b　c t r z　y d s k　e g f x　h u o n v m

のように設定する．このように，各文字を別の一文字に対応させる暗号方式を『換字式暗号』という．例えば上の表の変換であれば，

$$\text{night} \quad \Longrightarrow \quad \text{dcqbx}$$

という暗号文になる．pjx という暗号文を受け取れば，表をやはり逆から見て cat と元の文に戻すことができる．このような文字の変換の仕方は単純に考えて $26! = 26 \times 25 \times 24 \times \cdots \times 2 \times 1$ 通りある．現代のコンピュータを用いたとしても，階乗（上のような 26!）計算を総当たりしていくことは難しい．というのも，$26! = 403291461126605635584000000$ であり，これは 27 桁の数で 1 兆の 1 兆倍よりも大きい数である．次のセクションでも紹介するドイツのエニグマ暗号も暗号方式としてはこのような仕組みによって暗号化されている．水面下の研究を除けば，1977 年に **RSA 暗号**という新しいシステムが出てくるまでは，この暗号方式が主流であった．

さて，私は今，文字をいくつかずらすシーザー暗号は解読が簡単であり，ランダムにずらす場合は 26! 通りあるのでコンピュータを用いても解読が難しいと書いたが，その意味を少し説明したい．コンピュータは文字を文字としては認識しておらず，数値として認識している．代表的なものとして，アスキーコード（ASCII CODE[*2]）がある．これに従うと，

コンピュータはAを65，Bを66，…，Zを90のように読みかえている．なのでシーザー暗号であれば，コンピュータは暗号文をまずアスキーコードで数値にし，その数値から1から26までの数字を引いた値を計算して，再びアスキーコードによって文字として出力すれば，必ずどれかは正解の文章のはずである．この意味で26通りならコンピュータですぐに解読することができる．次にランダムの場合，一般的な換字式暗号の場合は，27桁通りの場合があるわけだが，例えば理化学研究所と富士通が共同で開発した日本のスーパーコンピュータ『京』は，1秒間に約10京回 = 10^{17} = 100000000000000000回と，18桁の回数計算ができる（！）．10年間は約3.2億秒で9桁，約60 * 60 * 24 * 365 * 10 = 315360000秒であるから10年間で27桁の回数計算が可能である．つまり京であっても約11年かけねば計算は終わらない（これでたった1通りの変換である）．

とはいえ，単純に総当たりしていくのはあまりに無策であるとも言える．そこで換字式暗号の実際例を探偵小説の中からいくつか紹介してみよう．

1.2 探偵小説における暗号

まずはエドガー・アラン・ポーの『黄金虫』に出てくる暗号から紹介したい．

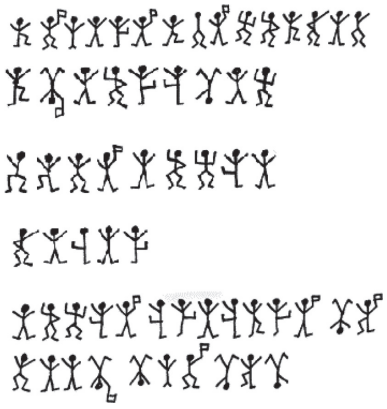
53 ‡‡† 305))6*;4826)4 ‡ .)4 ‡);806*;48 † 8¶
60))85;1 ‡ (;: ‡ *8 † 83(88)5* † ;46(;88*96
-;8) ‡ (;485);5* † 2:* ‡ (;4956*2(5*-4)8
¶8*;4069285);)6 † 8)4 ‡‡ ;1(‡ 9;48081;8:8
‡ 1;48 † 85;4)485 † 528806*81(‡ 9;48;(88;4
(‡ -34;48)4 ‡ ;161;:188; ‡ -;

このままでも良いのだが，これはとても見にくい．解読する側も大変だ．文字と記号の間にきちんと一対一の対応があればいいのだから，‡や¶を適当にアルファベットに書き換えても構わないはずで，その作業を行うと

FHXXGHYFVVICKNOWIVNXMVNXVKOYICKNOGOQ
IYVVOFKKDOCKRXCOGOHAOOVFCGKNIAKOOCJI
CZKOVCXAKNOFVKFCGWRCXAKNJFICWAFCPNVO
QOCKNYIJWOFVKVIGOVNXXKUAXJKNOYOUKORO
XUKNOGOFKNVNOFGFWOOYICOUAXJKNOKAOOKN
AXZHNKNOVNXKUIUKRUOOKXZK

[*2] 小説『ダ・ヴィンチ・コード』のように，暗号の意味としてコード（code）という単語は現在では一般的ではない．暗号の場合はcryptographyをあてる．

のようになる（数理科学 1975 年 12 月号，長田順行の記事によった）．これでずっと見やすくなった．解読法はまとめて後で説明することにして，次にコナン･ドイル作シャーロック・ホームズシリーズ『踊る人形』に登場する暗号をみてみよう．



さて，これらはどう解読したら良いだろうか．先に述べたように，単純に各アルファベットがどのアルファベットになるかを探索すると，26! 通りあり，解読はコンピュータを用いても困難なのであった．しかし実際のところ，『黄金虫』の暗号は簡単に解読することが可能である．そこで使われるアイデアがアラブの哲学者キンディによる**統計学的なアルファベットの偏り**を利用するものである．このアイデアは西暦 850 年ころには着目され，実用されている．当時分析したアルファベットは総計 100,362 個で，のちに 30 万個のデータベースに拡大された（Beker and Piper）がほとんど数値には変更はないものであった．

表 1　アルファベットの出現頻度（単位は％）

a	b	c	d	e	f	g	h	i	j	k	l	m
8.2	1.5	2.8	4.3	12.7	2.2	2.0	6.1	7.0	0.2	0.8	4.0	2.4
n	o	p	q	r	s	t	u	v	w	x	y	z
6.7	7.5	1.9	0.1	6.0	6.3	9.1	2.8	1.0	2.4	0.2	2.0	0.1

上の表を見てもわかるように，英文中では，e がもっとも出現頻度が高く，q や z はほとんど現れていない．この表を見ながら暗号文の中に現れているアルファベットの頻度を見て，ある程度どの文字がどの文字に変換されているのかの予想がつけられることになる．また，例えば 3 文字ずつアルファベットを見て，その 3 文字の並びに偏りがあれば，それは the なのではないか？というようなことも予測がつくだろう．実際，黄金虫の暗号を上の表を見ながら解読してもらいたい．少しコツをつかめば，15 分程度で解読できるので

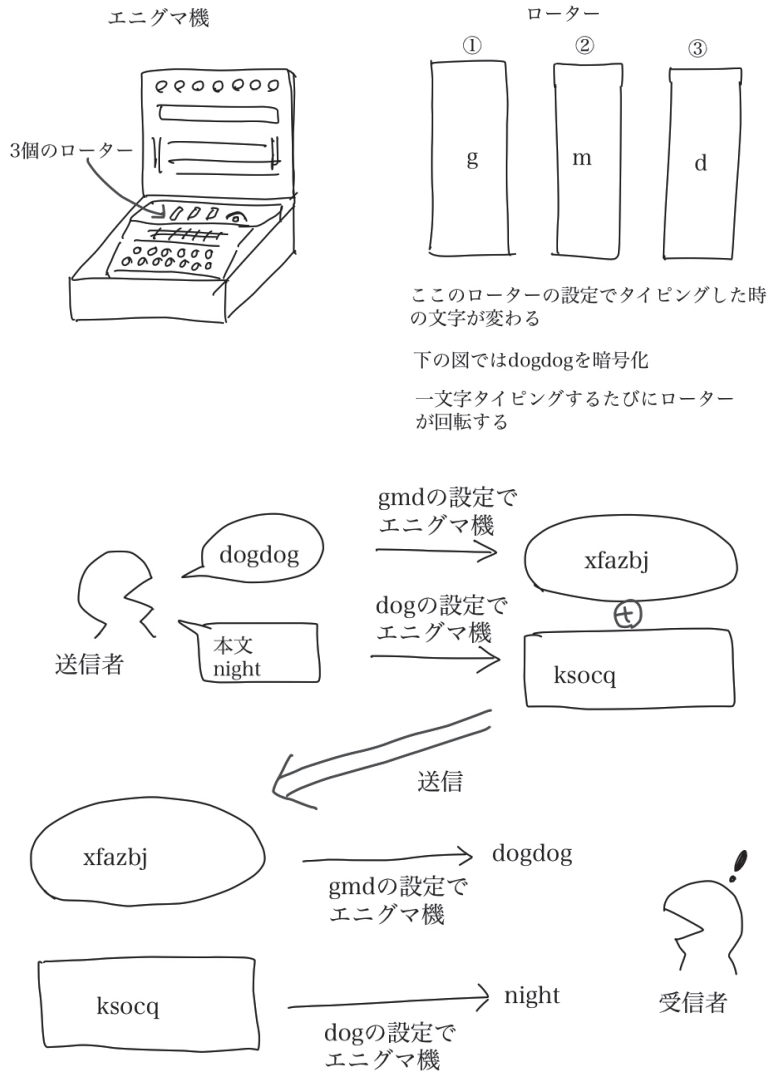
はないだろうか．今現在も利用されている『モールス符号（信号）』にも，このアルファベット頻度のアイデアが利用されている．アルファベット頻度による暗号解読は古い手法ではあるもの，それによってマイクロソフトの office2003 年版までの暗号化が解読されたというようなこともあった．

1.3　暗号鍵と復号鍵

暗号を考える際に重要な要素はたくさんあるのだが，ここでは 2 つ説明しておきたい．まず暗号化のアルゴリズム，暗号（化）鍵がある．これはつまり，元の文をどのようにして暗号文にするか？のその方法である．例えばシーザー暗号であれば『いくつアルファベットをずらすか』であり，一般の換字式暗号であれば，各アルファベットをどのアルファベットに変換するかといった変換表によって，暗号化される．一方で暗号文を元の文章に戻すことを暗号では『復号』と呼んでいて，復号（化）鍵というのは，暗号文を受け取った人がどのようにして元の文章に戻すか？という方法のことになる．スキュタレー暗号であれば，それは紙を巻き付ける棒であり，シーザー暗号，換字式暗号の場合は『暗号化鍵＝復号化鍵』である．スキュタレー暗号も多かれ少なかれ暗号鍵と復号鍵は同じものと言っていいだろう．このような暗号のことを『共通鍵暗号』または『対称（鍵）暗号』という．この方式の場合，とりわけその暗号化・復号化するための方法，すなわち鍵の管理を厳重にやっていく必要がある．

2　エニグマ暗号

1 章で述べたように，換字式暗号は復号化鍵を知らずともアルファベットの出現頻度によって解読されてしまう．実際，黄金虫は簡単に解読ができてしまうのだが，シャーロック・ホームズの踊る人形のほうはこれだけでは実は難しい．なぜならば，全体の文章がとても短く，出現頻度でアルファベットの予想をするのが困難だからである．**第 2 次世界大戦時，1918 年に誕生したエニグマ暗号**もシステムとしては換字式暗号なのだが，毎日アルファベットの置き換え方を変えることで出現頻度に着目できないようにし，暗号の安全を保障しようと考えていた．以下にエニグマ暗号のシステムを簡略化したものを図解したい．まず，『エニグマ機』という暗号・復号するための機械と，配布される『日替わり暗号鍵表』が必要になる．



上の流れを文章でも説明を加えたい．エニグマ機を使って暗号文を作るためには，まず 3 枚のローターと呼ばれる部分を設定する必要がある．ローターは歯車のようにグルグル回転し，アルファベットを決めることができる．このローターは 1 文字タイピングするたびに，少し回転する機構になっていた．これによって同じ a というアルファベットも 1 回目と 2 回目のタイピングの結果は異なるアルファベットが出力されるようになっている．その最初にローターに設定するアルファベットは『日替わり暗号鍵表』という本に書かれている．上の図は，当日の日替わり暗号表が『gmd』であったときの図になっている．また，送りたい文は『night』（実際にはもっと長い文章を送るが簡略化のため 5 文字だけにしている）だ．が，実際にすぐに night を暗号化するのではなく，間に一工程を入れる工

夫をしていた．それが暗号文を送信する人が night を暗号文にするときの暗号化鍵を設定しなおして良い，というものだ．新しい暗号化鍵は，送信者の判断で好きにして良いことになっていた．上の図では送信者が暗号化鍵を『dog』と決めたときの図になっている．よってその日の暗号鍵が gmd から dog に変更したことを受信者に伝えなければならないので，まず送信者は日替わり暗号鍵表の『gmd』を使って，dog をエニグマ機で暗号化する．しかしこの dog は送信者が勝手に決めたものなので，受信者にうまく伝わらない可能性を考慮し，念のため dogdog と同じ 3 文字を繰り返して 6 文字にしたものを送信していた．これで受信者は復号したときに，同じ単語が 2 回繰り返されていることを確認して，新しい暗号鍵を無事に受け取れたことを確認することにしていた．暗号表による暗号化と実際に暗号にしたい文を暗号化するための暗号化鍵を分けているのは，先にも述べたように，アルファベットの頻度による解析をさけるためである．そして送信者はローターの設定を『dog』にし直して，night を暗号化し送信する．これで送信は完了する．受信者は図のように，送信者の逆の手順を踏めば良い．

エニグマ機によるエニグマ暗号は 26! 通りには及ばないものの，全通りを調べると 1 兆通り以上あり，だれにも解読できない暗号なのではないかとも考えられていたようだ．しかしエニグマ暗号は『日替わり暗号鍵表』無しで解読されてしまう．その大きな理由は

（1）日替わり暗号表は無事だったが，エニグマ機は敵に渡ってしまった
（2）送信者が勝手に決めた暗号化鍵（上の図の dog のほう）が，何度も同じものを使いまわしていた
（3）送信者が勝手に決めた暗号化鍵（上の図の dog のほう）は念のため 2 回繰り返して送信していた（上の図では dogdog を暗号化していた部分のこと）

が挙げられる．(1)のエニグマ機の構造と(3)の新しい暗号化鍵を 2 回繰り返したことによって，探索する通りを減らすことができたからだ．もちろん大幅に減らすというだけで，即解読されるわけではないが，最終的にはアラン・チューリングという数学者がエニグマ暗号を解読するための機械『ボンブ』を開発することになる．

今現在，実際にエニグマ機を用いて暗号文を作ることは難しいが，web やアプリによって，容易にエニグマ暗号の真似をすることができる．例えばアプリとしては iPhone Developers East 社が提供している『The Enigma』がある（若干使いにくいアプリではあるが）．iPhone と名前は入っているが，アプリ自体は Android 端末用にも公開されている．またエニグマ機のレプリカを東京工業大学名誉教授の辻井重雄先生が所蔵されており，2015 年に映画『イミテーション・ゲーム』が公開された際には，何度かテレビ上でそのレプリカが公開されていた．

3　現状の暗号技術について（RSA 暗号を中心に）

第 1 章，第 2 章で見てきた暗号は暗号鍵と復号鍵が同じであったため，どのように暗号

化しているか？の管理を厳重にしなければならない．しかしながら暗号文を渡した人には解読してもらわねばならない以上，事前に解読方法（暗号化の方法と同じ）を渡しておく必要がある．戦時のように，少数の特定の人たちだけを対象としたやり取りであれば大きな問題ではないかもしれないが，スマートフォンや PC によって，自分の資産などの大切な情報を web サイトに入力しなければならない場合や，盗聴を含むサイバー攻撃などのリスクもあって，先に利用者に暗号鍵を渡しておくというのは非常に困難で危険である[*3]．そこで実際に通信を行う際に利用可能な暗号として考え出されたのが，**1977 年に登場した RSA 暗号**[*4] である．RSA 暗号は当時マサチューセッツ工科大学（MIT）の学生であった Rivest, Shamir, Adleman という 3 人の名前の頭文字からとられている．RSA 暗号は『公開鍵暗号』と呼ばれる（実質）最初の暗号手段であり，数年前の調査でも公開鍵暗号の中のほとんどが RSA 方式であるという結果もある[*5]．当然，公開鍵暗号のすべてが RSA 暗号ではないが，大部分を占めているのは事実であるし，実際この暗号の良いところ・悪いところ･もっている性質，などに着目すると，その他の暗号や今後の暗号がどうあるべきか？というものも見えてくる．今後いくつか数学的な記述は出てくるが，高校 1 年生程度の数学がわかっていれば，複雑なところは読み飛ばしても概略には影響がないような記述を心がけている．40 年もの間，現役であり続ける RSA 暗号のアイデアは，いたってシンプルで，必ずしも難しいことから大きな結果が得られるわけではないことの一つの例になっていると言えるだろう．

なお，今後『平文』などのように文とあっても，それはアスキーコードなどで数値化された数値であるものとする．

3.1 公開鍵暗号とは何か

公開鍵というのは，その名前の通り『暗号化するための鍵，つまり暗号化鍵を全員に公開しておく』方式の暗号のことである．まず手順は以下のようになる；

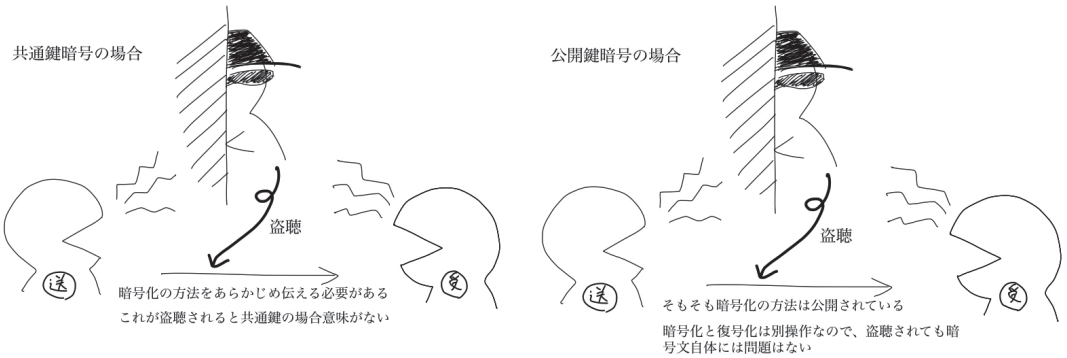
1. A さんは暗号化するための鍵を公開（誰でも見られるように）する
2. B さんは A さんに送りたい平文 M を，A さんが公開している暗号化鍵で暗号文 M' にする
3. 暗号文を受け取った A さんは，自分だけが知っている秘密鍵（復号化鍵）で M' を M にする．

[*3] サイバー攻撃というと大げさに聞こえるかもしれないが，現実に攻撃は活発になされている．大学などには 1 秒間に数回以上の攻撃を受けている例も珍しいことではない．どのような攻撃が考えられているのか？は第 4 章でとりあげてみたい．

[*4] 事前に暗号化・復号化するための方法（鍵）を知らせる問題を『鍵配送（共有）問題』という．RSA 暗号はそれを解決する一つの方法であり，RSA だけではもちろんない．

[*5] ここ数年で急激に楕円曲線暗号の利用が増えているが，まだ RSA 暗号のほうが多いだろうと考えられる．

これまでの共通鍵暗号とこの公開鍵暗号の一番の違いは下の図にあるような，単純な盗聴者が居ても，そもそも暗号鍵は公開されているわけだから，通信に問題はないことだ．なぜこのようなことが可能になるのかと言えば，第 2 章まで見ていた共通鍵暗号（対称鍵暗号）は『暗号鍵＝復号鍵』であったが，公開鍵暗号は『暗号鍵≠復号鍵』だからである．当然そのような良い性質（誰でも文章に鍵をかけられるが，鍵を開けることができるのは受け取った者だけ）をもった仕組みを作るには数学の力が必要になり，また安全性についても，なんらかの数学的な根拠[*6] が必要になる．



3.2　公開鍵暗号の代表例その 1：RSA 暗号

まず，合同式の定義だけ行いたい．『余り』に着目している関係式だ．

定義 1．a, b を整数，n を自然数とする．a を n で割った余りと b を n で割ったときの余りが等しいとき，

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

と書く．これは $a - b$ が n の倍数ということと同値である．

例 $27 \equiv 3 \pmod{12}$

27 を 12 で割ると商が 2 で余りは 3 になっている．この式は時刻でいうところの，27 時というのは 3 時と同じ，というようなニュアンスを数式で表現したものになっている．

実際には，合同式の場合，必ずしも割り算ができるのかはわからないので，通常の『=』とは意味が違うが，以下の記述については基本的に加法 +，減法 −，乗法 ×, ∗ のみの記述なので，ほとんど = のように読んでもらって構わない[*7]．

[*6] 素因数分解の困難性，離散対数問題，衝突（発見）困難性などがある．
[*7] 数学的には，この『できない』ところに注目すると，いろいろ面白い結果が出てくる．

そこで，RSA 暗号がどのように暗号化・復号化されるのかをみていきたい．実状よりも少し簡素な形で紹介しているが，概ね以下の通りである[*8]．

＜ **RSA 暗号による暗号化・復号化**＞

1-1. A さんは 2 つの素数 p, q を用意し，$n = p \times q$ を計算する．

1-2. A さんは $(p-1) \times (q-1)$ の値と互いに素な自然数 e を用意する．

1-3. A さんは n と e の 2 つを暗号化鍵として公開する（p と q は公開しない）．

2. B さんは A さんに送りたい平文 M（数値）を A さんが公開している e と n を用いて，M^e を計算して，それを n で割った余りを計算し，暗号文 M' にする．すなわち $M' \equiv M^e \pmod{n}$ が成立している．

3-1. 暗号文 M' を受け取った A さんは，以下のようにする．

3-2. $ed \equiv 1 \pmod{(p-1)(q-1)}$ となる d を求める．

3-3. $(M')^d$ を計算すると，$M = (M')^d \pmod{n}$ となり，元の文（平文）に戻る．

少し説明を加えていく．1-1 については，2030 年程度までは n は 2048bit（617 桁）の大きさであれば安全であろうと考えられている．

1-2 は基本的に e は $(p-1)(q-1)$ と互いに素であればなんでも良いのだが，$e = 2^{16} + 1 = 65537$ とすることが多い[*9]．

1-3 は，p と q を公開しないことがポイントで，3-2，3-3 で特に重要になる．

3-2 の d は高校 1 年で習うユークリッドの互除法を用いると，すぐに求めることができる．合同式だとわかりにくいのではあれば，

$$ax + by = c \quad (a = (p-1)(q-1), b = e, c = 1)$$

という 1 次不定方程式を解いていると考えて良い（$y = d$）．

3-3 を証明するためには，フェルマの小定理が必要になるので，紹介する．

[*8] 例えば中国の剰余定理から派生される多くの整数論を用いるほうが，計算が早く行える．これらのことは次回に回したい．

[*9] ちなみに $(p-1)(q-1) = \varphi(n)$（オイラー関数）である．

＜フェルマの小定理＞
a を整数，p を素数とするとき，

$$a^p \equiv a \pmod{p}$$

が成立する．
PROOF. まず $2 \leqq r \leqq p-1$ ならば ${}_pC_r = \dfrac{p!}{r!(p-r)!} \equiv 0 \pmod{p}$ である（分子の p は約分されないので，p の倍数である）．この事実と整数 a による数学的帰納法でフェルマの小定理を証明する．
$a = 1$ のときは明らかなので，a のときにフェルマの小定理が成り立つと仮定し，$a+1$ でも成立することをみてみる．2 項定理と先の結果より

$$(a+1)^p = a^p + {}_pC_1\, a^{p-1} + \cdots + {}_pC_{p-1}\, a + 1 \equiv a^p + 0 + \cdots + 0 + 1 = a + 1 \pmod{p}$$

となる．よって $(a+1)^p \equiv a+1$ より，$a+1$ でも成立している．a が負の整数であれば，合同式の性質で自然数に書き直してしまえば良い． □

＜フェルマの小定理の書き換え（系）＞
p を素数，a を p の倍数ではない整数とするとき，

$$a^{p-1} \equiv 1 \pmod{p}$$

が成立する．
PROOF. フェルマの小定理から $a^p - a = a\,(a^{p-1} - 1) \equiv 0 \pmod{p}$ なので，$a\,(a^{p-1}-1)$ は p の倍数である．ここで p は素数であり，a が p の倍数でないことから，$a^{p-1}-1$ が p の倍数である． □

3-3 の説明に戻ろう．$(M')^d \equiv (M^e)^d = M^{ed} \pmod{p}$ に対して，e, d は $ed \equiv 1 \pmod{(p-1)(q-1)}$ を満たすから，$ed = 1 + k\,(p-1)(q-1)$ となるような整数 k が存在する．よって，

$$M^{ed} = M^{1+k(p-1)(q-1)} = M \times (M^{p-1})^{k(q-1)}$$

となる．ここで，フェルマの小定理の書き換え（系）から p は素数なので $M^{p-1} \equiv 1 \pmod{p}$ である．よって

$$M^{ed} = M \times (M^{p-1})^{k(q-1)} \equiv M \times 1 = M \pmod{p}$$

となる．よって $M^{ed} - M$ は p の倍数である．同様のことを素数 q にも行えば，$M^{ed} - M$ は q の倍数であることもわかる．以上から $M^{ed} - M$ は異なる素数 p, q の倍数なので，$n = pq$ の倍数である．以上から目的であった

$$(M')^d \equiv M^{ed} \equiv M \pmod{n}$$

が示された．よって $M < n$ としておけば，これまでの計算を行えば，受け取った M' から M に戻すことができる．

例 非常に小さい数字で RSA 暗号の実験をしてみよう；
$p = 2, q = 17$ のとき，暗号化したい文章 $M = 26$（文を数値にした結果が $M = 26$ のときの意味）を暗号文にし，解読してみる．

（1-1） n を計算する．$n = 2 \times 17 = 34$ である．

（1-2） $(p-1)(q-1) = 16$ となるので，これと互いに素な数として，$e = 3$ としてみる．

（2） 公開されている $n = 34, e = 3$ を用いて，$M^e = 26^3 = 17576$ を $n = 34$ で割った余りを計算すると $M' = 32$ になる．これが暗号文になる．

（3-2） $ed \equiv 1 \pmod{(p-1)(q-1)}$ となる d を求める．この場合は $3d \equiv 1 \pmod{16}$ である．ユークリッドの互除法を用いれば容易だが，これくらいなら d にいくつか 1 から 15 までの適当な数値を代入して確かめるのも大変ではない．例えば $d = 11$ のとき，この等式を満たしているので $d = 11$ とする．

（3-3） 受け取った $M' = 32$ を $d = 11$ 乗したものを $n = 34$ で割った余りを計算する．計算機を使うと $32^{11} = 36028797018963968$ であり，これを 34 で割ると，商が 1059670500557763 で，余りが $26 = M$ となる．つまり

$$M = (M')^d \pmod{n}$$

であり，確かに元の文 M と一致した．

上の計算では計算機などで露骨に数値を計算したが，実際はユークリッドの互除法を含めた様々な整数論のアイデアで高速に計算できる方法があることは断っておきたい．

3.3 RSA 暗号の安全性と疑問

3.3.1 RSA 暗号の安全性の数学的根拠

セクション 3.2 の＜RSA 暗号の暗号化・復号化＞と RSA 暗号の例などをみるとわかるように，（3-2），（3-3）における $ed \equiv 1 \pmod{(p-1)(q-1)}$ で求める d が知られてしまうと，RSA 暗号は解読されてしまうことがわかる．何度か書いているが，実際にはユークリッドの互除法を行うと，$ed \equiv 1 \pmod{(p-1)(q-1)}$ を満たす d はすぐにわかってしまうので，$(p-1)(q-1)$ の値が計算できてしまえば，d は求まる．つまり RSA 暗号の安全性は

p, q を A さんは公開していないので，$(p-1)(q-1)$ の計算が行えるのは，A さんだけ

ということに依存している．それはつまり

n の値が大きければ，$n = p \times q$ の値を公開しても p, q の値はわからない

という，『**素因数分解の困難性**』を数学的な安全性の拠り所としている．その『n の値が大きければ』というのが，現在では n を 2048bit（617 桁）以上と定めている．第 1 章で非常に大雑把ではあるものの，26! という 27 桁の数であっても，スーパーコンピュータを用いて単純計算では 10 年かけても総当たりすることができないと書いた．今回は素因数分解なので，2048bit までのすべての数字で試してみる必要はないが，617 桁という数はあまりにも大きいと思われるかもしれない．しかし現在は様々な素因数分解の手法が発見されており，安全に使うという意味ではこれくらいが必要とみられている．新規採用の場合には 3072bit にしているくらいだ．

例えば 127451 を素因数分解せよ，と言われたらどうだろうか．たかが 6 桁の数と思うが，それでも手計算ではかなり困難な分解になると思われる．この場合は 50 番目の素数 233 と 100 番目の素数 547 の積なので，計算機を用いてバカバカしく思っても 1, 2, 3, …, と割っていくと 233 回目の計算によって素因数 233 が発見できる．しかし 233 × 547 を計算せよ，と言われれば手計算でも大したことはないし，コンピュータを用いて良いなら一瞬で計算が終わる．暗号の場合，このような『**一方向性関数**』という性質を用いて安全性を担保する．すなわち RSA 暗号であれば，$p \times q$ の計算は簡単だが，n の値を見て p × q という素数の組 p, q を得ることは難しいという一方向性である．

3.3.2　暗号の安全性評価

暗号が数学的に安全と言っても，時間を無限に使って良いのならいつかは解読されてしまうわけだから，なにをもって安全と言うか？ということも重要である．いろいろな方針があるが，ひとつその指標をあげるならば

そのときのスーパーコンピュータを用いて 1 年間計算し続けても解読できない程度

というのがある．1 年という時間を長いとみるか短いとみるかはあるが，一つの暗号を解くためにスーパーコンピュータを一年間独占するなどということはあり得ないし（費用も莫大），普通のコンピュータというのをどう定義するかは難しいが，普通のコンピュータを 1 万倍速くしたものがスーパーコンピュータの計算力と大雑把に見積もることができるので，充分安全性は確保されているとみて良い．

3.3.3　RSA 暗号の理論と現実

ここまでは RSA 暗号で用いられている理論や定理を（簡易的な形ではあるが）紹介してきたが，現実にそれを運用するとなるといくつか問題となることがある．それに答えていってみよう．

(1) RSA 暗号には 2 つの素数が使われているが，各々が素数を使っていったら素数はなくならないか？
⇒ 素数は無限にあるので問題はない（ユークリッドの結果）.

(2) n が 617 桁以上ということであれば，素数 p, q は 600 桁程度の素数しか候補にならないはずで，それでもコンピュータを用いても素因数分解は難しいのか？
⇒ とても難しい．もう少し正確に答えると，600 桁程度までの素数の個数は 1792 年に 15 歳のガウスが予想し，1896 年に解決された『素数定理』

$$\pi(x) \sim \frac{x}{\log x} \qquad (\pi(x) \text{ は } x \text{ までの素数の個数を表す記号})$$

によって 10^{597} 個以上あることがわかる．ただし対数の底は e だ．簡単に言えば，自然数 n が素数である確率は $\frac{1}{\log n}$ くらいということを表している．ガウスがどのようにしてこのアイデアにたどり着いたのかを考えるのは面白い．この素数定理によると 600 桁くらいの素数はめちゃくちゃたくさんあるということがわかる．一度素数定理と実際の素数の個数を比較してみると面白い.

表 2　素数定理による素数の個数と実際の素数の個数との対比

x	x までの実際の素数の個数 $=\pi(x)$	素数定理による素数の個数近似
10	4	5.12
100	25	29.08
1000	168	176.56
10000	1229	1245.09
100000	9592	9628.76
1000000	78498	78626.50
10^{21}	21127269486018731928	21117412262909985552.2

10^{21} までの素数の個数と素数定理の誤差は 9857223108746376 個，約 17 桁分で，10^{21} が 22 桁であることを考えれば驚異的な精確さだと言っていいだろう．素数定理になぜ対数 log が出てくるのか？などはそれほど難しくはないので，調べてみても面白い．例えば参考文献[8]は参考になるだろう.

(3) (2)の補足．リーマン予想との関連
(2) で 617 桁までの素数はたくさんあって，表によれば $x = 10^{21}$ までの素数は 21127269486018731928 個あることがわかる．ところでどうやってこの個数を求め

たのだろう？ 実際一つ一つの数字が素数かを確かめたのだろうか？ 10^{21} であればまだしも，10^{600} などという巨大な数までを一つ一つ確かめることは不可能なので，解析的な方法と組み合わせ論を用いる方法によって計算されている．また素数定理の精度は最も有名な未解決問題の一つ『リーマン（Riemann）予想』が証明されれば，さらに精度があがる．すなわち暗号理論は整数論と強い強い関連があり，それは素数定理やリーマン予想といった数学のとてもメジャな定理や未解決問題と繋がっている，とても面白い分野だと言えるだろう．ちなみに 2016 年 8 月 10 日現在発見されている最も大きな素数は 2016 年 1 月 7 日に証明された $2^{74207281}-1$ である．これは 2000 万桁以上ある．一年間にいくつか最大の素数は更新されていくことが多い．素数 p に対する 2^p-1 の形をした素数のことをメルソンヌ素数という．近年実際に証明された最大の素数はほとんどすべてメルソンヌ素数で，メルソンヌ素数でない最大の素数の発見となると 1989 年まで遡る．メルソンヌ素数に有効なアルゴリズムがあることが理由だ．素数に関する話題は web サイト http://primes.utm.edu/ が詳しい．

（4）RSA 暗号に必要な巨大な素数はどうやって用意するのか？

今までのことからわかるように，巨大な数になると，そこまでの素数の個数を調べたり因数分解することはとても難しい．すると RSA に必要な巨大な素数を『どうやってそれが素数であると確かめるのか』という問題に突き当たる．実際何百桁にもおよぶ数を実際に素数であると証明するのはとても大変であるし，それこそ素数であることが証明されている数だけを用いて RSA 暗号を作ってしまったら，安全性に問題がある（素因数分解されてしまう）．実際の RSA 暗号では，一般には素数だと証明されていない数を用いるほうが良いに決まっているが，それをどうするのか？という問題の解決には『ミラー・ラビンの素数判定法』が使われていることが多い．この判定法を用いると『ほぼ 100％』その数は素数か合成数か判定できる．このあたりの話も次回話したい．

3.4 RSA 暗号における電子署名

公開鍵暗号を用いると，だれでも暗号文を送ることができるメリットがある一方，例えば B さんが A さんに暗号文を送った際に，受け取った A さんは『本当にこれは B さんが送ってきたものなのだろうか』という疑問をもつことになる．それを解決するのが電子署名というもので，日本でいうところの印鑑，海外におけるサインのような役割を果たすことができる便利な機能がある．ここでは詳しく見せるというより，実際に使われている形とは異なるがイメージがつきやすい簡易的な方法を書いておきたい．もう一度 RSA 暗号に暗号化の方法をおさらいすると，

＜**RSA 暗号による暗号化・復号化**＞
1-1. A さんは 2 つの素数 p, q を用意し，$n = p \times q$ を計算する．
1-2. A さんは $(p-1) \times (q-1)$ の値と互いに素な自然数 e を用意する．
1-3. A さんは n と e の 2 つを暗号化鍵として公開する（p と q は公開しない）．
2. B さんは A さんに送りたい平文 M（数値）を A さんが公開している e と n を用いて，M^e を計算して，それを n で割った余りを計算し，暗号文 M' にする．すなわち $M' \equiv M^e \pmod{n}$ が成立している．
3-1. 暗号文 M' を受け取った A さんは，以下のようにする．
3-2. $ed \equiv 1 \pmod{(p-1)(q-1)}$ となる d を求める．
3-3. $(M')^d$ を計算すると，$M = (M')^d \pmod{n}$ となり，元の文（平文）に戻る．

であった．n は公開するが，その n の素因数 p, q を知っているのは自分（A さん）だけというのがポイントで，このことを指紋のように考えて，署名で用いることができる．

＜**RSA 暗号による簡易的署名方法**＞
1-1. 暗号を送る B さんも 2 つの素数 p', q' を用意し，$n' = p' \times q'$ を計算する．
1-2. B さんは $(p'-1) \times (q'-1)$ の値と互いに素な自然数 e' を用意する．
1-3. B さんは n' と e' の 2 つを暗号化鍵として公開する（p' と q' は公開しない）．
1-4. B さんは $e'd' \equiv 1 \pmod{(p'-1)(q'-1)}$ となる d'（復号鍵）を求める．
2. B さんは A さんに送りたい平文 M（数値）を A さんが公開している e と n を用いて，M^e を計算して，それを n で割った余りを計算し，暗号文 M' にする．そしてさらに B さんが用意した非公開の d' を用いて自分の名前の数値 b を d' 乗，$b^{d'} \pmod{n'}$ する．
3. 受け取った A さんは暗号文は自分の d を用いて復号する．そして最後の署名のところは B さんが公開している e' を用いて，$(b^{d'})^{e'}$ を求める．これを同じく B さんが公開している n' で割れば，RSA 暗号の仕組みから $(b^{d'})^{e'} \equiv b \pmod{n'}$ のはずなので，B さんの名前 b が出てくる．

つまりイメージとしては『B さんも RSA 暗号を用意する』というものに近い．ポイントは上の 1-4 で出てくる d' は素因数分解の困難性から B さんにしか計算できない．それを用いて A さんに名前を送り，A さんに B さんが公開している e', n' を用いてもらって復号してもらい，B さんの名前が出てくる．このような仕掛けができるのは d' を知っている B さんだけのはずだ．公開している部分と非公開にしている部分をうまく用いて署名を実現している．

3.5 RSA 暗号の問題点

当然 RSA 暗号は良いところばかりあるわけではないし，問題点もたくさんある．ここでは 2 つ紹介したい．

1. 計算に（ある程度）時間がかかる．
   RSA 暗号を暗号化，復号化するには巨大な素数があり n は 2048bit にもなる．実際に暗号化するときには，元々の平文 M も巨大な数になるから例えば暗号化のときには

$$1234467899884736^{28940392349439483493748399483934 83939}$$

   のような計算が必要になる（上の例でも小さすぎるくらい）．いくらコンピュータが計算が早いとはいえ，一回一回の処理でこの計算をするのは（ある程度）大変なことと言える[*10]．実際他の公開鍵暗号である 1985 年に登場した『楕円曲線暗号』における公開鍵は 224bit で RSA 暗号の 2048bit と同程度の安全性を保てることが知られている．IC カードのような小さなカードに埋め込み，多くの人が行き来する改札機などでは非常に速い計算が必要になることが多いので，計算は早いほうが良いのは当然である．ちなみに共通鍵暗号（対称暗号）であれば，あらかじめ暗号化・復号化の鍵を渡しておく必要はあるものの，112bit で RSA 暗号 2048bit と同程度の安全性が保てる．しかしながら共通鍵暗号の場合には n 人とやり取りがあると $\frac{n(n-1)}{2}$ 個の鍵を用意しなければならないので，$n = 1000$ であっても 499500 個，人類 60 億人とすれば 1.8×10^{17} 個の鍵は必要で，やはり人数が多い通信のようなときには不向きだ．しかしながら『共通鍵暗号は計算が速く済む』という事実は重要で，のちに紹介する『ハイブリッド暗号』のアイデアに繋がる．
2. 同じ文章は同じ暗号文になってしまう（決定性アルゴリズム）．
   暗号には決定性アルゴリズムと確率的アルゴリズムという概念がある．決定性というのはつまり，同じ文章が同じ暗号文になってしまうということで，このことは非常に困る．例えば電子入札（オークションでもいい）を考えてみよう．事前にいくらで入札したかがわかってしまうと困るが，今のままの RSA 暗号では暗号化の際に行われる計算は決まっているので平文 M に対する暗号文 M' は常に M' になる．すると一つの額に対する暗号文は常に同じなので，暗号文をみていくらかわかってしまうことが考えられる．また yes か no かで答えるようなアンケートでは RSA 暗号のままでは秘密を保ったまま投票することができない．これを回避する方法は現在では確立されている．すなわち確率的アルゴリズムであるような RSA 暗号のシステムも確立されている．

3.6　その他の公開鍵暗号とその応用

簡単に RSA 暗号以外の公開鍵暗号と RSA 暗号を含めた暗号の応用について書いておきたい．

[*10] NAF 変換，モンゴメリ高速乗算などの高速にべき乗を求めることのできるアルゴリズムがあり，それによって可能になっている．

3.6.1 ElGamal 暗号（エルガマル暗号）

鍵生成 大きい素数 p とある性質を満たす自然数 g を用意し[*11]，ランダムな数 x を決める．そして $y \equiv g^x \pmod{p}$ となる y を計算し，p, g, y を公開し，公開鍵とする．x が秘密の復号鍵になる．

暗号化 ランダムに数 r を決めて，元の文 M に対して

$$M' = (M_1, M_2) = (g^r, M \times y^r)$$

を暗号文とする．ただしすべて mod p，すなわち p で割った余りにしておく．ElGamal 暗号の場合，暗号文は (M_1, M_2) という二つの組（ベクトル）の形で送られてくる．

復号化

$$M \equiv \frac{M_2}{(M_1)^x} \pmod{p}$$

が成立している[*12]．

安全性 ここまでのことで，暗号化，復号化されていることを確かめることは簡単だが，なぜこれで安全と考えられているのかについては少し説明が必要だ．RSA 暗号の場合は，素因数分解の困難性によって安全を保っていたが，ElGamal 暗号の場合は**離散対数問題**という問題によってそれを保障しようとしている．離散対数問題とは鍵生成の部分における記号をもう一度用いて説明すると，

p, g, y がわかって，p が大きい素数のときに $y \equiv g^x \pmod{p}$ となる x を求めること

ということになる．例えば $3^x \equiv 1 \pmod{5}$ なら 5 で割った余りなので $x = 1, 2, 3, 4$ で確かめれば充分であるが $3^x \equiv 225 \pmod{233}$ となる x がいくつかを答えることは難しい（この例の場合 $x = 100$ である）．ElGamal 暗号の場合，p として 2048bit 程度であれば安全が保たれると考えられている．もちろん余りの世界でなければ，つまり mod p がついていなければ単なる対数の問題だということも付け加えておく[*13]．

ElGamal 暗号は暗号化の際に自分でランダムな数字 r を選ぶことができるので，元々の文が同じであっても，r を変えることによって暗号文を変えることができる．すなわち確率的アルゴリズムである．

[*11] g は数学の言葉における巡回群 $(\mathbb{Z}/p\mathbb{Z})^*$ の生成元とする．

[*12] mod p での割り算があるので，実際は $(M_1)^x$ の逆元を掛け算することになる．

[*13] つまり対数という『連続な関数』であれば容易だが，余りという『離散』なものを考えることは難しいということになる．

3.6.2 SSL 通信とハイブリッド暗号

インターネットサイトを見ているとき，例えば自分の個人情報などを入力する場面などのときに，画面上に『セキュリティで保護された接続（SSL)』といった文章や，サイトのアドレスが表示されているバーが緑色になり鍵のマークが現れ，サイトアドレスが『https:』と http → https に変化したところを見たことがあるかもしれない．

SSL とは Secure Socket Layer という世界で標準的に使われている暗号通信で，現在はそれを新しくした TLS（Transport Layer Security）という通信もある．現実に使われている技術なので，きちんと流れを追っていくのは難しい部分もあるので，署名のときと同様に，簡易的な形で表現したい．SSL でない通常の通信の場合，

（1）相手の http サーバに接続リクエストを出す

（2）相手が自分にレスポンス（web ページ）を返す

それを SSL では

（1）相手の https サーバに接続リクエストを出す

（2）相手から自分にレスポンスとして公開鍵証明書（ここでは単に暗号の公開鍵と思っていても良い）という電子証明書を返してくれる

（3）自分のブラウザは受け取った証明書の検討を始め[*14]，問題なければ乱数を使ってこの通信で使うための『共通鍵（対称鍵）暗号』を作り，その共通鍵暗号の鍵を相手からもらった公開鍵を用いて暗号化して，相手に送信する[*15]

（4）相手は送られてきた公開鍵で暗号化された共通鍵暗号の鍵を，サーバが用意していた（公開鍵の）秘密鍵を用いて復号化し，共通鍵暗号の鍵を手に入れる．今後はこれでやりとりする[*16]

（5）もし検討した結果，失敗だった場合には，通信を続けるかの確認をとる

という流れで行う．このように，**公開鍵を用いてその後使う共通鍵暗号の鍵の受け渡しを行い，そのあとの暗号化は共通鍵暗号で行う仕組みを『ハイブリッド暗号』**という．

上の(3)における『証明書の検討』というところでは，相手が本当に自分がやりとりをしようとしているところなのだろうか？といった疑問を解決する必要があるので，電子署名の技術などが利用されていることは想像できるだろう．問題は SSL 通信で必要な『証明書』たちをどのように用意するかであるが，それには PKI（公開鍵基盤）という仕組み

[*14] 検討に必要な『ルート証明書』は最初からブラウザに埋め込まれているのでユーザはほとんど意識しなくて良い．

[*15] 最後まで公開鍵暗号で通信をしないのは，RSA 暗号の問題点 1 で書いたように計算に時間がかかるためである．鍵の受け渡しの問題さえ安全に済んでしまえば，共通鍵暗号であれば公開鍵よりもはるかに速く計算が可能である．

[*16] SSL の場合，公開鍵として楕円曲線暗号，共通鍵暗号として AES 暗号を用いることが多い．

がとられている．簡単に言えば，必要な公開鍵を信頼のおける第 3 者機関に保証してもらうことする．そうでないと署名云々の前に，そもそも自分の思い描いている人と違う人と通信をしようとしているかもしれないからだ．その第 3 者機関が本当に安全かという問題はもちろんあるが，暗号というのはより多くの人の目にさらされているほうがむしろ良いという考えがある．そういう厳しい目をくぐり抜けてきているほうが安全だと考えているからだ．このような PKI の基礎的な考えは MIT の学生が作ったが，その指導教授は RSA 暗号の生みの親のアデルマンである．

3.7　暗号の準同型性

第 3 章の最後に，今も今後も重要な性質になるであろう，暗号の準同型性について話したい．『準同型』という言葉はなじみの薄い言葉かもしれないので，その意味ではこの節は少し数学色が強いかもしれないが，重要な性質であるので簡単ではあるが触れておきたい．ここでいう準同型性というのは『暗号文を暗号文のまま，平文（元の文）の計算（加工）ができる』と考えてもらっても良いかもしれない．情報がもれて危険なのは，暗号文を平文に直し，その時に何か攻撃をうけたり，管理を誤ると情報漏えいにつながることになる．準同型性は平文に戻すことなく計算ができるので，ある種その問題を解決する一つの方法になっているし，個人対個人ではなく，大勢で多くの暗号文を処理することができる．ただし逆に暗号文のまま意味のある改ざんを行うことができるため，攻撃を受けた際に改ざんされるリスクがあることは忘れてはならない．これらの性質上，準同型暗号というのは基本的に公開鍵暗号で意味をもつことになる．

＜**定義 2**：準同型暗号＞
平文 x に対して，暗号化した暗号文を $f(x)$ と表したとき，

$$f(x_1) * f(x_2) = f(x_1 * x_2)$$

が成立しているとき，（乗法的）準同型暗号という．

上の定義における掛け算（乗法）のところは +（加法，足し算）であっても良い．その場合は加法的準同型と呼ぶ．また 2009 年には加法的と乗法的の両方で準同型性をもつ暗号も実現されている．加法的であり乗法的でもある準同型暗号を完全準同型暗号とも呼ぶ．実用レベルで使えるような完全準同型暗号はまだ構成できていないようで，活発に研究がされている．

3.7.1　準同型暗号の例

RSA 暗号は乗法的な準同型暗号であることが，合同式の性質をもちいた計算によってわかる．なお e, n などの文字は RSA 暗号の暗号化，復号化の部分と合わせている．

RSA 暗号は

$$\text{平文 } M \longrightarrow \text{暗号文 } M' \equiv M^e \pmod{n}$$

という計算によって暗号化されていた．すなわち，$f(M) = M^e \pmod{n}$ である．よって

$$f(M_1) * f(M_2) = (M_1)^e * (M_2)^e = (M_1 * M_2)^e = f(M_1 * M_2) \pmod{n}$$

が成立している．よって準同型性をもつ． □

また，ElGamal 暗号の暗号化を

$$M' = (M_1, M_2) = (g^r, M \times y^r)$$

から

$$M' = (M_1, M_2) = (g^r, g^M \times y^r)$$

にすることで，準同型暗号になる．値は ElGamal 暗号と同様，mod p しておく．これで準同型性をもつことの証明は，RSA 暗号とほぼ同じなので興味があれば確かめてもらいたい．

3.7.2 準同型暗号の応用例：電子投票

例えば国民投票のような，賛成か反対かは秘密に投票したいが，最終結果を決めたいときに，準同型暗号は利用できる．yes に投票したならば 1，no なら 0 を投票というルールにしておくと $f(1)$ というのが，1 の暗号文になっているので元々が 1 か 0 かはわからないことに注目する（ただしこの場合の暗号は 3.5 節で説明した確率的アルゴリズムであることが必要になる）．わかりやすく加法的準同型暗号であったとすると

$$f(1) + f(0) + f(1) + f(1) + f(0) + \cdots = f(1 + 0 + 1 + 1 + 0 + \cdots) = f(\text{1 の個数})$$

とできる．もう少し複雑にはなるが極簡易的な電子投票も考えてみよう．選挙のような場合に A さんに投票なら a，B さんに投票なら b，C さんなら c，のように定めておくと

$$f(a) + f(b) + f(b) + f(c) + \cdots = f(a + b + b + c + \cdots) = f(\mathrm{l}_a a + \mathrm{l}_b b + \mathrm{l}_c c + \cdots)$$

ここで l_a などは a の係数，すなわち A に投票した個数（人数）になることがわかる．

3.7.3 補足：ペアリング暗号

正確にはペアリング暗号というのは準同型暗号ではないが，持っている性質としては近しいものがあるので簡単に触れておきたい．準同型暗号同様に，様々なことに応用が効くため研究が盛んに行われている．なお，下の定義は楕円曲線を用いたペアリングを意識しているため点の和と演算値における積でのペアリングで記述している．数学的には加法と

乗法は呼び方，記号の問題なので違和感はない．

＜**定義 3**（簡易的）：ペアリング暗号（双線型暗号）＞
2 つの点 P, Q に対して，写像（演算）$e(P, Q)$ が，

$$e(P_1 + P_2, Q) = e(P_1, Q) * e(P_2, Q), \quad e(P, Q_1 + Q_2) = e(P, Q_1) * e(P, Q_2)$$

の 2 式を満たすとき，双線型（ペアリング）写像（演算）という．この性質を満たす写像を用いて暗号化される暗号をペアリング暗号という．

2 つの点はどこにある点なのかといったことや，『この性質を満たす写像を用いて暗号化される』という表現はとてもずるい言い方ではあるが，きちんとしたペアリング暗号の定義には多くの準備が必要なのでこれくらいの言い回しにさせてもらいたいと思う．このペアリング写像の定義から

$$e(aP, Q) = e(P, aQ) = e(P, Q)^a$$

が成立する．このことが ElGamal 暗号のときのような『離散対数問題』へと繋がり，点をある楕円曲線からとってくることにより楕円曲線における離散対数問題へと繋がっていく．楕円曲線における離散対数問題は安全を保証する bit が小さくて済むこともあり，計算が速く行える利点がある．

4　暗号への攻撃

4.1　コンピュータ全般における攻撃あれこれ

暗号は情報セキュリティの一つである．暗号があればそれで安心，というものでは全くないことをまず確認するために，どのような脅威が考えられるのかを少し述べてみたい．

まず馴染みのある言葉として『ハッキング』があるかもしれない．正確な定義は難しいが，ハッキングとはネットワークやソフトウェアの脆弱性（弱点）をついて，様々な悪影響をもたらすものと考えられるかもしれない．例えば次のようなものが考えられる[*17]．

（1）盗聴
　ネットワークを利用してコンピュータ間でのやりとりを不正に傍受すること
（2）不正侵入
　権限を持たない第三者がネットワークを経由して不正アクセスすること
（3）不正使用
　権限を持たない第三者が，コンピュータやネットワークなどの情報資産を使用する

[*17] 参考文献[2]『サイバーセキュリティ入門』p 53 からの引用．一部表現は変えています．

こと

(4) なりすまし
ユーザ ID やパスワードを不正入手し，正当なユーザのふりをして行為を行うこと

(5) 改ざん
コンピュータやネットワークの情報を不正に書き換えること

(6) 妨害
サーバなどに大量のパケットを送信するなどして，サービスやサーバを停止させたりすること

上のようなものの多くは一部暗号が役立つともいえるが，暗号だけではどうしようもない部分が大きい．さらに言えば直接 PC を盗まれたりするようなこともあり得る．もちろん暗号によって中身を防御することはできるかもしれないが，情報セキュリティはいろいろなものが相互的なつながりをもって初めて意味をなすことが大きいことは確認しておこう．他にもトロイの木馬といったコンピュータウィルスもあるが，今回はそこには触れずに，次に暗号への直接的な攻撃を考えてみたい．

4.2 暗号に特化した攻撃の種類

一口に『安全な暗号』といってもそもそもの定義が難しく，第 3 章で例えば『スーパーコンピュータを 1 年間動かし続け』ても大丈夫であればという目安を書いたが，そもそもどうやって動かすか，ただ全通りを試していくのか[*18]（ブルート・フォース・アタック），そうでなければどんなアルゴリズムで動かすのが適当か，など難しい．攻撃についても同様で，前節での『盗聴』などの攻撃も相手がどういうアルゴリズムでどんな機械を用いるか？ どんな精度なのか？ ということも現実にはわからない．そこで暗号への攻撃といった場合には，どんな形の攻撃があったかということは考えず，ある攻撃によって以下のような状況（仮定）になった，ということを想定し，それに耐えうるように暗号を作っていこうという発想になっている．

(1) 暗号文単独攻撃（COA）
暗号文をみて，そこから元々の平文が解読されないこと．そもそも暗号という言葉の定義のようなもの．もちろんいつかは解読されるわけだから，そこにスーパーコンピュータを使っても，1 年以内で，などの条件が入る．いまやエニグマ暗号などはこの条件を満たしていないことになる．では例えば RSA 暗号はどうか？ということについては次の(2)で考えたい．

(2) 選択平文攻撃（CPA）
あらゆる平文に対する暗号文がわかっているという仮定のもと，目的の暗号文からは元々の文が解読されないこと．RSA 暗号をはじめとする公開鍵暗号はそもそも

[*18] 例えば復号鍵が n bit だとすれば 2 進法であるから 2^n 通り試せば必ず鍵はわかる．

暗号化鍵を公開しているので，任意の平文に対する暗号文を得ることは可能だ．よってCPAに対して安全でなければ公開鍵暗号とは呼べなくなる．RSA暗号の場合には，結局のところ『n, eを公開し，p, q, dを非公開にしたときに解読されないか』ということになり，3章で書いたように安全だと現時点では考えられている．しかし未知のアルゴリズムが存在し，それによって解読されることは考えられる．現在のところではまずnの素因数分解をうまくやるアルゴリズムは無いのか，ということを中心に議論は進んでいる．そして目覚ましく発展している．代表的なものとして『数体ふるい法』や『楕円曲線法』という手法がある．このことは次回述べたい．

（3）選択暗号文攻撃（CCA）

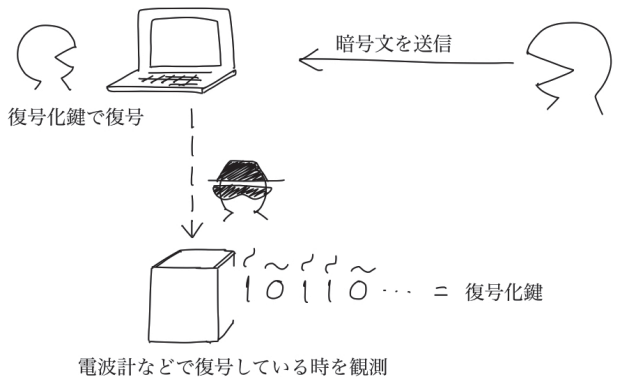
目的の暗号文を除く，あらゆる暗号文に対する平文がわかっているという仮定のもと，目的の暗号文からは元々の文が解読されないこと．例えばすごい機械があって，（目的の暗号文以外で）適当な暗号文を入力するとその平文を教えてくれるといったケースが考えられる．これはとても強い仮定だが，公開鍵暗号の場合には特に，自分であらゆる平文を打つことができてその暗号文が得られるし，盗聴などもされているかもしれないので，このような仮定のもとでの安全性は重要な要素になる．ではRSA暗号はどうだろうか．残念なことに第3章のままのRSA暗号ではこの選択暗号文攻撃によって解読はされないものの，平文の一部の情報がもれてしまうことがわかっている．そもそもRSA暗号は今のままでは決定性アルゴリズムでもあって，弱点が多い．このCCAに対する安全性も確率的アルゴリズムへの変貌も同時に改良しているものに『RSA-OAEP（Optimal Asymmetric Encryption Padding）暗号』がある．

他にも解読の技術的な方法として『差分解読法』や『man-in-the-middle攻撃』などもある．興味のある人は調べてもらいたい．

4.2.1 サイドチャンネル攻撃

最後にユニークな攻撃について書いてみたい．1990年代に考えられた攻撃で当時は技術的にそれほど脅威ではなかったものの，現在では熱心に対策が考えられている，比較的新しい攻撃方法になる．

暗号化という操作は現代ではコンピュータやスマートフォン・携帯電話などの機械を必ず使用している．機械を使用するには電気が必要になり，電気というのは波形をもっている．この形を読み取って，例えば暗号を平文に直すとき，すなわち復号鍵を使用しているときの電波や電気使用量などを観測することで，復号鍵を予測したり手に入れたりしようとする試みだ．



もちろん上のように単純にはいかないかもしれないが，ノイズなどによって多少うまく読み込みができなくとも，統計の考えを利用することで推察ができることもある．暗号技術そのものはもちろんだが，こうして破る側の技術もあがっていることは忘れてはならない．

5　量子コンピュータ登場以後の暗号について

ここまで暗号の過去と現在を考えてきたので，最後に未来について考えてみたい．

5.1　暗号の危殆化

暗号の危殆（きたい）化というのは，現在は安全であっても時間が経つにつれ，機械の性能の向上や新しい攻撃によって，利用している暗号の廃止や改良をしなければならない状況になること，または暗号の安全性が低下すること全般を指す言葉として使われている．例えば 2010 年には 1024bit の RSA 暗号は使わない，2048bit も 2030 年まで，のように考えている．暗号の危殆化の大きな要因をもう少し詳しく述べると

（1）コンピュータの性能向上予測（ムーアの法則）
ムーアの法則という 1965 年に発表された経験則に類する予測で，簡単に言うと『計算速度は 10 年で 100 倍速くなる』というもの．現実にそれにかなり近いことも確認できる．よって暗号も 10 年後には 100 倍速くなっているかもしれない，と考えていつまで使用するかを考えることになる．

（2）現状アルゴリズムの改良，新しい攻撃アルゴリズムの誕生
RSA 暗号ができた 1977 年に，2 つの素数の積である $n = p \times q$ が 129 桁のときの素因数分解を行え，という問題が雑誌に掲載された．4 京年かかると見積もる者もいたほど，それは難しい作業であると考えられていたが，実際には 1994 年に『2 次ふるい法』というアルゴリズムで分解に成功している．前に少し述べたが，現在は素因数分解のアルゴリズムとしては『数体ふるい法』（や『楕円曲線法』）が主役だ．現在は特殊な形をしていない数でも 170 桁程度の数が素因数分解可能である．また

素数であることの厳密な証明も 15000 桁程度であれば可能だ．しかしこれらの記録も新しいアルゴリズムの誕生によって劇的に現状が変化するかもしれない．

5.2 量子コンピュータ

量子コンピュータとは，1985 年に提唱された，量子計算という手法によって計算される『超並列計算機』（のようなもの）と考えられる．量子計算とは，粒子は同時に複数の状態を持つことができるので，1 つの粒子を使って 2 進法における 0 と 1 の状態を同時に処理できるようになれば，粒子を n 個並べれば 2^n 通りの状態をまとめて計算できる，という理論だ．まだ完成の目途は無いようではあるが，もし量子コンピュータができればRSA 暗号における素因数分解の困難性や楕円曲線暗号や ElGamal 暗号における離散対数問題は解かれ，暗号としては使えなくなるだろうと考えられている．また，完成の目途が無いとはいえ『耐量子コンピュータ暗号』の開発は熱心に行われている．というのも，理論ができてもそれを実装するには 10 年程度の時間が必要になるので，量子コンピュータができました，では暗号を変えましょう，とはすぐにはできない．その『耐量子コンピュータ暗号』として『（イデアル）格子暗号』がある．これは本稿で書いていない『衝突発見困難性』という問題によって暗号の安全性を保障しようとしているものだ[*19]．しかし 2016 年 7 月 19 日，九州大学と KDDI 研究所によって（60 次元以下の場合）解読に成功したとの発表がなされた．これは世界初のことである．このことによって，格子暗号であっても，どの程度高い次元にすれば解読されないか，さらなる高速解読アルゴリズムを作れないか，といった研究がされていくことになるだろう．

量子コンピュータの登場はまだまだ先のことになるだろう．今は公開鍵としては RSA 暗号と楕円曲線暗号が主流だと書いたが，徐々にペアリング暗号ならびに多重線型暗号に移っていき，そして耐量子コンピュータ暗号へと変遷する，というような流れになるのではないだろうか．実装に 10 年とも書いたが，一度実装するとそのシステムを変えるのは大変な作業だ．RSA 暗号よりも楕円曲線暗号のほうが計算が速く，短い鍵 bit で同程度の安全性が保てるとわかっていても，RSA 暗号のほうが主流になっているのは，そのようなシステムの変更に伴う混乱をさけるためでもあるだろう．しかしここ 2，3 年で楕円曲線暗号が Blu-ray やデジタルテレビにおける著作権保護の暗号に使用されるなど，利用も増えてきたようだ．

安全な暗号を作っても，実際にそれを動かしたり管理したりするのが人間である以上，ミスは起こり情報漏えいが全く無くなるというのは，無いかずっと先のことだと思う．最終的には人間の問題だとも言えるだろう．そういった人間側の問題で暗号が破られたことは何度かあり，国家的な問題になったこともある．このような事例を紹介するのも興味深

[*19] 故意に誤差を付加した多元連立 1 次方程式を解く問題，と言える．

いとは思うが，今回は割愛し暗号の紹介や機能のほうを中心に書いた．とにかく暗号は身近であることは間違いないし，RSA などのアイデアは単純なのに，奥が深い．一口に暗号といっても関連する分野は広大で自分の好きな分野でがんばることもできそうだ．技術的な分野もあれば純粋に数学として追い求めるにしても，整数論，関数論，リーマン予想との関わりもあり充分な魅力がある．本稿は多少の専門性があるにしても，例えば高校生のような若い人が暗号に対する興味をもってもらえないだろうか，ということを一番念頭にいれて書いたつもりである．本稿は平成 28 年度学習院安倍能成記念教育基金学術助成金による研究成果の一部である．

参考文献

［1］今井秀樹監修，“トコトンやさしい暗号の本”，日刊工業新聞社，2010
［2］猪俣敦夫，“サイバーセキュリティ入門”，共立出版，2016
［3］結城浩，“暗号技術入門（第 3 版）”，SB クリエイティブ，2015
［4］岡本栄司・西出隆志，“暗号と情報セキュリティ”，コロナ社，2016
［5］Richad Crandall・Carl Pomerance（監訳：和田秀雄），“素数全書”，朝倉書店，2010
［6］光成滋生，“クラウドを支えるこれからの暗号技術”，秀和システム，2015
［7］楫元，“工科系のため初等整数論入門”，培風館，2000
［8］木田祐司，“初等整数論”，朝倉書店，2001