

公開鍵暗号概要

森 隆 Takashi Mori

〔要 旨〕

盗聴されうる通信路を用いて秘密情報を安全にやりとりするための暗号は、古くは第一次世界大戦の軍事外交用途に用いられ、長い歴史をもつ。暗号通信では鍵に基づいて暗号化および復号を行うが、暗号化用の鍵と復号用の鍵が等しい共通鍵暗号では、どのようにして安全に共通鍵を送受信者で共有するかが課題となる。つまり、盗聴されうる通信路を用いて共通鍵を送信することはできないため、安全な別の手段で共通鍵を配送する必要があった。この課題を解決するのが公開鍵暗号であり、暗号化と復号で異なる鍵を用意して、暗号化用の鍵は公開鍵として公開し、復号用の鍵は秘密鍵として秘密にする方式である。また、盗聴されうる通信路を用いて安全に秘密鍵を共有する鍵共有方式もあり、これも広義の公開鍵暗号である。公開鍵暗号は、インターネット上の暗号通信に広く用いられ、現代の情報化社会において必要不可欠な技術となっている。本稿では、現在の公開鍵暗号として RSA 暗号、DH 鍵共有、楕円曲線暗号の基本原理解説し、次世代の公開鍵暗号として量子コンピュータを用いても解読が困難な耐量子計算機暗号の概要について紹介する。

1 はじめに

通信により秘密情報を安全にやりとりしたいという要求に対して、実際の通信では第三者に盗聴される可能性がある。特に無線通信は第三者による傍受が容易で、かつ傍受されたことが正規の送受信者には分かりにくいという問題がある。このため、第三者に通信を盗聴されても秘密情報が分からないようにする暗号が必要となり、古くは第一次世界大戦の軍事外交用途に用いられ、長い歴史をもつ¹⁾。つまり、暗号の目的は盗聴されうる通信路を用いて安全に秘密情報を送信することである。

例えば古典的な暗号では、図 1 のように送信者(Alice)は平文「hello」の各アルファベットを 4 文字ずらして暗号文「lipps」を受信者(Bob)へ送信する。盗聴者(Eve)は暗号文「lipps」を盗聴しても意味が分からないのに対して、受信者(Bob)は受信した暗号文「lipps」の各アルファベットを 4 文字戻して平文「hello」を得ることができる。この暗号を「方式」と「鍵」に分けて考えると、方式はアルファベットを n 文字ずらすことであり、鍵は $n = 4$ となる。暗号を解読するには方式と鍵の両方が必要となるが、古典的な暗号では方式が分かると暗号の解読が容易になるため、方式と鍵の両方を秘密にしていた。上記の例では、アルファベットを n 文字ずらす方式が分かれば、26 通りの n を調べると必ず解けてしまう。

一方、現代の暗号では、方式が既知でも鍵が未知であれば暗号の安全性が保たれるようにしており、鍵のみを秘密とし方式は公開している。方式を秘密にするより公開する方が広く研究され、むしろ安全性が高くなると考えられている。



図 1 古典的な暗号の例

前述の古典的な暗号では、暗号化でずらす文字数 n と復号で戻す文字数 n' は等しくする($n = n'$)必要がある。このように、暗号化用の鍵と復号用の鍵が等しいものを共通鍵暗号と言う。共通鍵暗号では、どのようにして安全に共通鍵を送受信者で共有するかが課題となる。つまり、盗聴されうる通信路を用いて共通鍵を送信することはできないため、安全な別の手段を用意して共通鍵を配送する必要があった。この課題を解決するのが公開鍵暗号である。図 2 に示すように、受信者(Bob)は暗号化と復号で異なる鍵を用意し、暗号化用の鍵は公開鍵として公開し、復号用の鍵は秘密鍵として秘密にする。送信者(Alice)は公開鍵を用いて平文を暗号化して暗号文を送信し、受信者(Bob)は秘密鍵を用いて暗号文を復号して平文を得る。暗号化用の公開鍵では暗号文を復号できないため、盗聴者(Eve)が公開鍵と暗号文を入手しても平文を得ることはできない。公開鍵暗号の発明により、盗聴されうる通信路のみを用いて安全に秘密情報を伝送できるようになった。本稿では、現在広く用いられている主な公開鍵暗号の基本について解説するとともに、次世代の公開鍵暗号である耐量子計算機暗号の概要について紹介する。

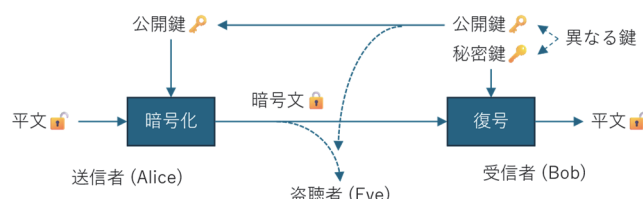


図 2 公開鍵暗号の手順

2 公開鍵暗号の特徴

多対多の暗号通信に必要な鍵の数を考える。共通鍵暗号では、鍵を共用すると共用した鍵で暗号文を復号できるため、暗号通信のすべての組み合わせに対してそれぞれ異なる鍵が必要となる。これに対して公開鍵暗号では、複数の送信者が暗号化用の公開鍵を共用しても、公開鍵では暗号文を復号できないため安全性が保たれる。このため多対多の暗号通信を行う場合でも、必要な鍵の組は受信者の数だけで良い。例えば8対8の通信では、**図3**に示すように、共通鍵暗号では28個の鍵が必要となるのに対して、公開鍵暗号では8組の鍵で良い。

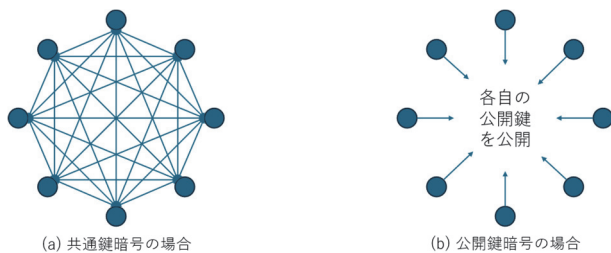


図3 8対8の暗号通信に必要な鍵の数

公開鍵暗号は、秘密鍵の共有が不要で多対多の暗号通信も容易であるという利点を持つが、暗号化および復号の計算量が多いという欠点がある。よって、大量のデータを公開鍵暗号で送信するのは負荷が大きいという課題がある。一方、共通鍵暗号は、共通鍵を安全に共有する必要があり、多対多の暗号通信には多数の共通鍵が必要であるという欠点があるが、暗号化および復号の計算量が少ないという利点を持つ。このため、最初に公開鍵暗号を用いて送受信者で鍵を共有し、次にその鍵を用いて共通鍵暗号でデータを送信する手法が、インターネット上の暗号通信に広く用いられている。公開鍵暗号を用いた鍵の共有はデータ量が少ないため計算量は大きな問題にならず、大量のデータは共通鍵暗号により効率的に送信されるとともに、共通鍵の共有は公開鍵暗号によって安全に行われるため、両暗号の欠点を巧妙に回避した手法である。

3 現在の公開鍵暗号²⁾

3.1 RSA 暗号

RSA 暗号は、1978年に Rivest, Shamir, Adleman の3者によって発表された公開鍵暗号であり(RSA は3者の頭文字)、大きな数の素因数分解が難しいことに基づいている。つまり、2つの素数 p, q が与えられたとき、それらの積 $n = p \times q$ を求めるのは容易であるが、積 n が与えられたときに p, q を求める素因数分解

は計算量が膨大になる。

RSA 暗号の手順を図示すると**図4**のようになる。

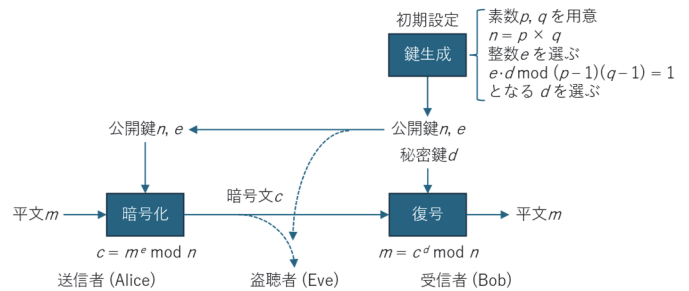


図4 RSA 暗号の手順

まず受信側にて以下の初期設定を行う。

- 2つの素数 p, q を用意する
 - 素数の積 $n = p \times q$ を求める
 - $(p-1)(q-1)$ と互いに素な、 $(p-1)(q-1)$ より小さい正の整数 e を選ぶ
 - $e \cdot d \bmod (p-1)(q-1) = 1$ となる d を選ぶ(mod は除算の余り)
 - n と e を公開鍵として公開する
 - d は秘密鍵として秘密にしておく
- 小さな数で数値例を示すと、
- 2つの素数 $p = 3, q = 5$ を用意する
 - 素数の積 $n = 3 \times 5 = 15$ を求める
 - $(3-1)(5-1) = 8$ と互いに素な、8より小さい正の整数 $e = 3$ を選ぶ
 - $e \cdot d \bmod (3-1)(5-1) = 1$ となる $d = 3$ を選ぶ($e \cdot d = 3 \times 3 = 9, 9 \bmod 8 = 1$)
 - $n = 15, e = 3$ を公開鍵として公開する
 - $d = 3$ は秘密鍵として秘密にしておく(この数値例では e と d が等しいが、実際には異なる値)

RSA 暗号では、 $\bmod n$ におけるべき乗計算の性質、具体的には整数 m を $\bmod n$ で $(p-1)(q-1) + 1$ 乗すると元の m に戻るという性質を利用している。例えば**表1**の数値例のように、いずれの m も $\bmod 15$ で $(p-1)(q-1) + 1 = 9$ 乗すると元の m に戻ることが分かる。

表1 mod 15 におけるべき乗計算

m	べき乗数								
	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	2	4	8	1	2	4	8	1	2
3	3	9	12	6	3	9	12	6	3
4	4	1	4	1	4	1	4	1	4
5	5	10	5	10	5	10	5	10	5
6	6	6	6	6	6	6	6	6	6
7	7	4	13	1	7	4	13	1	7
8	8	4	2	1	8	4	2	1	8
9	9	6	9	6	9	6	9	6	9
10	10	10	10	10	10	10	10	10	10
11	11	1	11	1	11	1	11	1	11
12	12	9	3	6	12	9	3	6	12
13	13	4	7	1	13	4	7	1	13
14	14	1	14	1	14	1	14	1	14

送信したいメッセージを m とすると、送信側では公開鍵 n, e を用いて次式によりメッセージ m を暗号化し、暗号文 c を送信する。

$$c = m^e \bmod n \quad (1)$$

受信側では秘密鍵 d を用いて次式により受信した暗号文 c を復号する。

$$m = c^d \bmod n \quad (2)$$

暗号化と復号により、メッセージ m を $\bmod n$ で $e \cdot d$ 乗することになるが、初期設定にて $e \cdot d \bmod (p-1)(q-1) = 1$ となるように e, d を選んでおり、 m を $\bmod n$ で $(p-1)(q-1) + 1$ 乗すると元の m に戻るから、復号により元のメッセージ m が得られる。例えば表2の数値例のように、 m を $e = 3$ 乗して暗号化し、暗号文を $d = 3$ 乗して復号すると元の m になる。

表2 べき乗による暗号化と復号の例

m	暗号化 べき乗数			復号 べき乗数		
	1	2	3	1	2	3
1	1	1	1	1	1	1
2	2	4	8	8	4	2
3	3	9	12	12	9	3
4	4	1	4	4	1	4
5	5	10	5	5	10	5
6	6	6	6	6	6	6
7	7	4	13	13	4	7
8	8	4	2	2	4	8
9	9	6	9	9	6	9
10	10	10	10	10	10	10
11	11	1	11	11	1	11
12	12	9	3	3	9	12
13	13	4	7	7	4	13
14	14	1	14	14	1	14

このように RSA 暗号では、 n と e が公開されているが、復号には d が必要となる。もし p と q が分かれば、 $e \cdot d \bmod (p-1)(q-1) = 1$ より d を求めることができ、解読されることになる。

n が分かっているときに $n = p \times q$ となる素数 p, q を求めるのは素因数分解であり、 p と q を非常に大きな数にすると素因数分解が困難になることから、RSA 暗号の解読は困難となる。ただし計算機の能力向上により、素因数分解の計算速度も年々速くなるため、定期的に p と q を大きくする必要がある。なお、受信側での初期設定では、2つの素数 p, q を用意した後に積 $n = p \times q$ を算出するため、大きな数の素因数分解を行う必要はない。

RSA 暗号の問題点として、鍵長増加に伴う問題と鍵生成の運用面の問題が挙げられる³⁾。鍵長増加に伴う問題は、安全性を確保するため計算機性能の向上に応じて鍵長を長くする必要があるのに対して、IC カードや組込機器など処理性能に制限がある環境では、長い鍵の処理が難しい点である。鍵生成の運用面の問題は、RSA 暗号では2つの大きな素数を使用するが、一方の素数が異なる利用者で同じ場合、秘密鍵を比較的簡単に導出できる点である。無作為に素数を選択すると異なる利用者が同じ素数を使用する確率は無視できるほど小さいが、乱数発生器の不具合や管理者のミスなどにより、同じ素数を使用する可能性があり、一方の素数が同じでも2つの素数の積である公開鍵は異なるため、重複を見つけにくい点が問題となる。

3.2 DH 鍵共有

DH 鍵共有は、1976年に Diffie と Hellman の2者によって発表された鍵共有方式であり(DH は2者の頭文字)、大きな数の離散対数問題が難しいことに基づいている。つまり、整数 g 、素数 p 、整数 a が与えられたとき、べき乗 $g^a \bmod p$ を求めるのは容易であるが、整数 g 、素数 p 、べき乗 $g^a \bmod p$ が与えられたときに整数 a を求める離散対数問題は計算量が膨大になる。

べき乗 $g^a \bmod p$ の計算は、 a 回の乗算を行うのではなく、次のようにして少ない乗算回数で計算することができる。 $x_2 = g^2 \bmod p$ 、 $x_4 = x_2^2 \bmod p = g^4 \bmod p$ 、 $x_8 = x_4^2 \bmod p = g^8 \bmod p$ 、 $x_{16} = x_8^2 \bmod p = g^{16} \bmod p$ 、 $\dots$ を計算し、例えば22乗を求めたいときは $x_{16} \cdot x_4 \cdot x_2 = g^{16} \cdot g^4 \cdot g^2 \bmod p = g^{22} \bmod p$ のようにして、大きな数のべき乗も効率的に求めることができる。同様に、 $y_a = g^a$ の y_a と g が分かっているときに a を求める対数問題は、次のようにして容易に解くことができる。 $x_2 = g^2$ 、 $x_4 = x_2^2 = g^4$ 、 $x_8 = x_4^2 = g^8$ 、 $x_{16} = x_8^2 = g^{16}$ 、 $\dots$ を計算し、例えば $x_{16} \cdot x_4 > y_a \geq x_{16}$ のように範囲を絞り込んでいくことで、効率的に a を求めることができる。このように $\bmod p$ なしの対数問題は効率的に解けるものの、 $\bmod p$ ありの離散対数問題ではこの計算方法が使えないため、 a を求めることは困難となる。

DH 鍵共有の手順を図 5 および以下に示す。

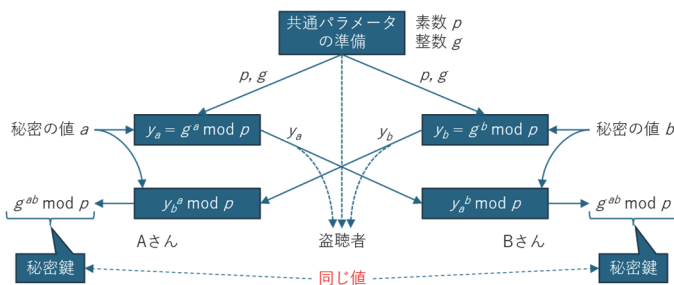


図 5 DH 鍵共有の手順

- ・ 共通パラメータとして素数 p と整数 g を公開する
- ・ Aさんは秘密の値 a を決めて、 $y_a = g^a \bmod p$ を計算して公開する
- ・ Bさんは秘密の値 b を決めて、 $y_b = g^b \bmod p$ を計算して公開する
- ・ Aさんは自分の秘密の値 a と Bさんの公開情報 y_b を用いて秘密鍵 $y_b^a \bmod p$ を計算する
- ・ Bさんは自分の秘密の値 b と Aさんの公開情報 y_a を用いて秘密鍵 $y_a^b \bmod p$ を計算する
- ・ 両者の秘密鍵は共に $g^{ab} \bmod p$ となる

小さな数で数値例を示すと、

- ・ 共通パラメータとして素数 $p = 23$ 、整数 $g = 9$ を公開する
- ・ Aさんの秘密の値は $a = 17$ とし、 $y_a = 9^{17} \bmod 23 = 3$ を公開する
- ・ Bさんの秘密の値は $b = 5$ とし、 $y_b = 9^5 \bmod 23 = 8$ を公開する
- ・ Aさんは秘密の値 $a = 17$ と Bさんの公開情報 $y_b = 8$ を用いて秘密鍵 $8^{17} \bmod 23 = 13$ を計算する
- ・ Bさんは秘密の値 $b = 5$ と Aさんの公開情報 $y_a = 3$ を用いて秘密鍵 $3^5 \bmod 23 = 13$ を計算する
- ・ 両者の秘密鍵は共に 13 となる

このように、盗聴されうる通信路で秘密鍵を送信することなく、両者で秘密鍵を共有することができる。盗聴者においては、素数 p 、整数 g 、 $y_a = g^a \bmod p$ 、 $y_b = g^b \bmod p$ が公開されているため、もし a の値が分かれば $y_b^a \bmod p$ により秘密鍵を算出可能であり、もし b の値が分かれば $y_a^b \bmod p$ により秘密鍵を算出可能である。しかし、 $y_a = g^a \bmod p$ から a の値を求める、もしくは $y_b = g^b \bmod p$ から b の値を求めるのは離散対数問題であり、 p を非常に大きな素数にすると離散対数問題を解くのが困難になることから、DH 鍵共有の解読は困難となる。ただし計算機能力向上により離散対数問題を解く速度も年々速くなるため、定期的に p を大きくする必要がある。なお、DH 鍵共有の手順では秘密の値 a 、 b を用意した後にべき乗 $y_a = g^a \bmod p$ 、 $y_b = g^b \bmod p$ を算出するため、離散対数問題を解く必要はない。

前節の RSA 暗号は任意のメッセージを伝達することができるため、秘密鍵をメッセージとして送信することにより鍵共有も可能であり、データの暗号化と鍵共有の両方に使用可能である。それに対して DH 鍵共有は、両者で何らかの秘密鍵を共有することができるが、任意のメッセージを伝達することはできないため、鍵共有専用の方式である。

3.3 楕円曲線暗号

楕円曲線暗号は、1985 年頃に Miller と Koblitz によってそれぞれ独立に提案された公開鍵暗号であり、楕円曲線上の離散対数問題が難しいことに基づいている。つまり、楕円曲線上の点 G と自然数 s が与えられたとき、楕円曲線上の乗算 $T = s \times G$ を求めるのは容易であるが、楕円曲線上の点 G 、楕円曲線上の乗算 $T = s \times G$ が与えられたときに自然数 s を求める楕円曲線離散対数問題は計算量が膨大になる。

楕円曲線は $y^2 = x^3 + ax + b$ で表される曲線であり、例えば $a = -1$ 、 $b = 1$ の場合の楕円曲線は図 6 のようになる。楕円曲線上の点を $G = (0.5, 0.790569)$ とし、図 7 に示すように点 G における接線と楕円曲線の交点 A を求め、点 A の y の符号を反転した点を B とする。これを楕円曲線上の加算および乗算 $B = G + G = 2 \times G$ とする。同様に図 8 に示すように、点 G と点 B を結ぶ直線と楕円曲線の交点 C を求め、点 C の y の符号を反転した点を D とする。これを楕円曲線上の加算および乗算 $D = B + G = 3 \times G$ とする。また図 9 に示すように、点 A と点 B を結ぶ直線と楕円曲線の交点が無い場合 $A + B = O$ とし、楕円曲線上の減算を $B = O - A = -A$ とする。楕円曲線上の乗算 $T = s \times G$ の計算は、 s 回の加算を行うのではなく、次のようにして少ない加算回数で計算することができる。前節のべき乗の計算と同様に、 $A_2 = G + G = 2 \times G$ 、 $A_4 = A_2 + A_2 = 4 \times G$ 、 $A_8 = A_4 + A_4 = 8 \times G$ 、 $A_{16} = A_8 + A_8 = 16 \times G$ 、 $\dots$ を計算し、例えば 22 倍を求めたいときは $A_{16} + A_4 + A_2 = 16 \times G + 4 \times G + 2 \times G = 22 \times G$ のようにして、大きな数の乗算も効率的に求めることができる。

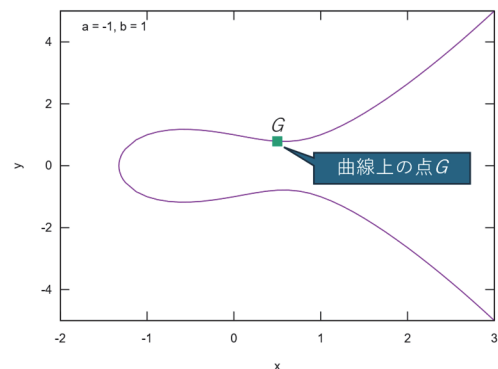


図 6 楕円曲線の例

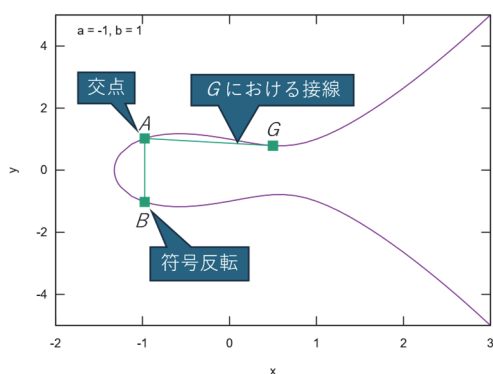


図 7 楕円曲線上の加算の例 1

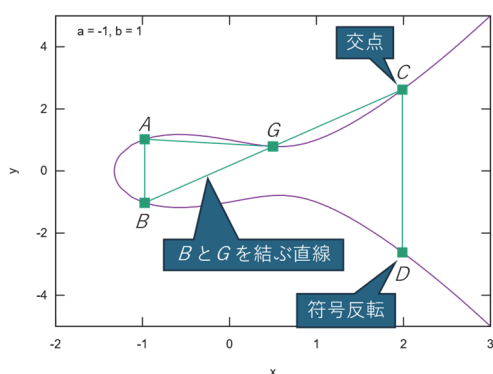


図 8 楕円曲線上の加算の例 2

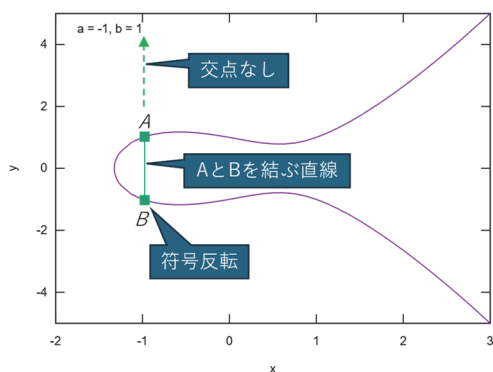


図 9 楕円曲線上の減算の例

公開鍵暗号に用いられる離散的楕円曲線は、上記の楕円曲線を整数化したものである。素数を p とし、 x, y, a, b は 0 から $p-1$ までの整数とする。離散的楕円曲線は $y^2 = x^3 + ax + b \pmod{p}$ で表される離散的な整数値である。離散的楕円曲線上の点 G を決めると、自然数 s が既知の場合、離散的楕円曲線上の乗算 $T = s \times G$ は前述と同様に効率的に求めることができる。逆に $T = s \times G$ から自然数 s を求める離散対数問題は解くことが難しい。

楕円曲線暗号の手順を図 10 および以下に示す。

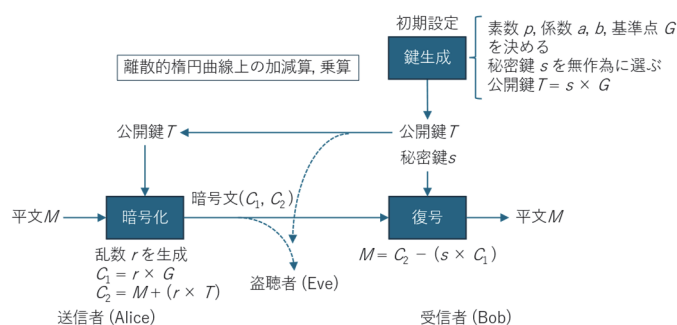


図 10 楕円曲線暗号の手順

鍵生成:

- ・ 共通パラメータとして素数 p , 係数 a, b , 基準点 G を公開する
- ・ 受信側で秘密鍵 s を無作為に選び、 $T = s \times G$ を公開鍵として公開する

暗号化:

- ・ 送信側で乱数 r を生成し、 $C_1 = r \times G$ を計算する
- ・ 送信するメッセージ M に対して $C_2 = M + (r \times T)$ を計算し、 (C_1, C_2) を暗号文として送信する

復号:

- ・ 受信側では、暗号文 (C_1, C_2) と秘密鍵 s を用いて $C_2 - (s \times C_1)$ を計算する
- ・ その結果、 $C_2 - (s \times C_1) = M + (r \times s \times G) - (s \times r \times G) = M$ となり、メッセージ M が得られる

小さな数の数値例を示すと、

鍵生成:

- ・ 共通パラメータとして素数 $p = 263$, 係数 $a = 2, b = 3$, 基準点 $G = (200, 39)$ を公開する
- ・ 受信側で秘密鍵 $s = 13$ を選び、 $T = s \times G = (114, 104)$ を公開鍵として公開する

暗号化:

- ・ 送信側で乱数 $r = 35$ を生成し、 $C_1 = r \times G = (219, 39)$ を計算する
- ・ 送信するメッセージ $M = (88, 84)$ に対して $C_2 = M + (r \times T) = (250, 161)$ を計算し、 $(C_1, C_2) = ((219, 39), (250, 161))$ を暗号文として送信する

復号:

- ・ 受信側では、暗号文 (C_1, C_2) と秘密鍵 $s = 13$ を用いて $C_2 - (s \times C_1)$ を計算する
- ・ その結果、 $(250, 161) - (13 \times (219, 39)) = (250, 161) - (201, 155) = (88, 84)$ となり、メッセージ M が得られる

楕円曲線暗号の利点として、短い鍵長と鍵生成の運用面の利点

が挙げられる³⁾。楕円曲線暗号は、RSA 暗号と比較して 1/10 程度の鍵長で同等の安全性が得られるため、IC カードや組込機器など処理性能に制限がある環境での利用にも適している。楕円曲線暗号では、秘密鍵として利用者ごとに異なる自然数 s を選択するが、もし異なる利用者間で同じ自然数 s を選択すると、公開鍵 $T = s \times G$ も同じになるため、重複を見つけやすい。これにより、脆弱な鍵を発行する問題が生じにくいという鍵生成の運用面の利点を持つ。

3.4 楕円曲線 DH 鍵共有

楕円曲線 DH 鍵共有は、楕円曲線を用いて DH 鍵共有を行う方式であり、略称は ECDH (Elliptic curve Diffie-Hellman key exchange) である。楕円曲線暗号と同様に、楕円曲線上の離散対数問題が難しいことに基づいている。

楕円曲線 DH 鍵共有の手順を図 11 および以下に示す。

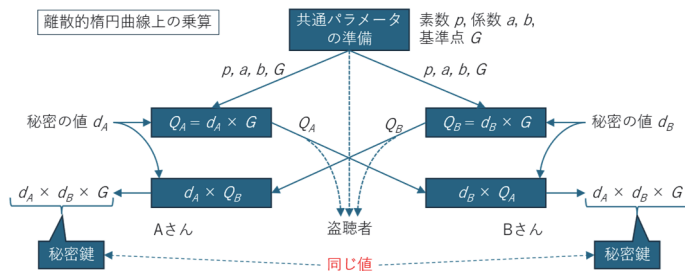


図 11 楕円曲線 DH 鍵共有の手順

- ・ 共通パラメータとして素数 p , 係数 a, b , 基準点 G を公開する
- ・ A さんは秘密の値 d_A を決めて、 $Q_A = d_A \times G$ を計算して公開する
- ・ B さんは秘密の値 d_B を決めて、 $Q_B = d_B \times G$ を計算して公開する
- ・ A さんは自分の秘密の値 d_A と B さんの公開情報 Q_B を用いて秘密鍵 $d_A \times Q_B$ を計算する
- ・ B さんは自分の秘密の値 d_B と A さんの公開情報 Q_A を用いて秘密鍵 $d_B \times Q_A$ を計算する
- ・ 両者の秘密鍵は共に $d_A \times d_B \times G$ となる

小さい数の数値例を示すと、

- ・ 共通パラメータとして素数 $p = 263$, 係数 $a = 2, b = 3$, 基準点 $G = (200, 39)$ を公開する
- ・ A さんは秘密の値 $d_A = 23$ を決めて、 $Q_A = d_A \times G = (198, 95)$ を公開する
- ・ B さんは秘密の値 $d_B = 79$ を決めて、 $Q_B = d_B \times G = (129, 9)$ を公開する
- ・ A さんは自分の秘密の値 $d_A = 23$ と B さんの公開情報 $Q_B = (129, 9)$ を用いて秘密鍵 $d_A \times Q_B = 23 \times (129, 9) = (133, 138)$ を計算する

- ・ B さんは自分の秘密の値 $d_B = 79$ と A さんの公開情報 $Q_A = (198, 95)$ を用いて秘密鍵 $d_B \times Q_A = 79 \times (198, 95) = (133, 138)$ を計算する

- ・ 両者の秘密鍵は共に $d_A \times d_B \times G = (133, 138)$ となる

このように、盗聴されうる通信路で秘密鍵を送信することなく、両方で秘密鍵を共有することができる。

前節の楕円曲線暗号は任意のメッセージを伝達することができるため、秘密鍵をメッセージとして送信することにより鍵共有も可能であり、データの暗号化と鍵共有の両方に使用可能である。それに対して楕円曲線 DH 鍵共有は、両方で何らかの秘密鍵を共有することができるが、任意のメッセージを伝達することはできないため、鍵共有専用の方式である。

3.5 公開鍵暗号の強度

前述の公開鍵暗号の安全性は、大きな数の素因数分解や離散対数問題を解く計算量が膨大になることに基づいている。世界 1 位のスーパーコンピュータの演算性能と、素因数分解および楕円曲線上の離散対数問題を 1 年間で解くのに必要な演算性能の比較が CRYPTREC から報告されており⁴⁾、現在使用されている 2048 ビット RSA や 256 ビット ECDH を解読するためには、1000 年以上の非現実的な時間を要することになる。ただし、スーパーコンピュータの演算性能はムーアの法則に従って年々向上しているため、定期的に暗号の鍵長を長くして解読に要する計算量を増やす必要がある。逆に言うと、定期的に鍵長を長くすることにより、ムーアの法則に従ったスーパーコンピュータの演算性能向上に対抗して、公開鍵暗号の安全性を維持することができる。

一方、量子コンピュータを用いて素因数分解および離散対数問題を効率的に解く「ショアのアルゴリズム」が存在する。このため、RSA 暗号と楕円曲線暗号は理論的には解読されたことになる。しかし、実際の暗号を解読するには大規模な量子コンピュータが必要であり、現在の量子コンピュータでは規模が全然足りず、実際には解読されていない。ただし、攻撃者が盗聴した暗号文を保存しておき、将来量子コンピュータの性能が向上した頃に暗号文を解読する「Harvest now, decrypt later 攻撃」が知られている。時間経過により陳腐化しない機密データにおいては脅威となるため、事前の対策が必要となる。

4 耐量子計算機暗号

4.1 CRYSTALS-Kyber

現在の RSA 暗号および楕円曲線暗号の後継として、大規模な量子コンピュータを用いても解読が困難な耐量子計算機暗号

(PQC: Post-Quantum Cryptography)の研究開発が活発に行われている。米国 NIST は 2016 年から PQC 標準化の公募を開始し、鍵交換アルゴリズムとして格子問題に基づいた CRYSTALS-Kyber が採択され、標準化文書 FIPS 203 として公開されている⁵⁾。耐量子計算機暗号で用いられている格子問題は、

$$\begin{cases} a_{11}s_1 + a_{12}s_2 + \cdots + a_{1n}s_n \approx t_1 \pmod{q} \\ a_{21}s_1 + a_{22}s_2 + \cdots + a_{2n}s_n \approx t_2 \pmod{q} \\ \vdots \\ a_{m1}s_1 + a_{m2}s_2 + \cdots + a_{mn}s_n \approx t_m \pmod{q} \end{cases} \quad (3)$$

上式の素数 q の剰余の連立線形近似方程式において、 $\mathbf{A} = (a_{ij})$ と $\mathbf{t} = (t_i)^T$ から $\mathbf{s} = (s_1, s_2, \dots, s_n)^T$ を求める問題である。誤差が無い“=”の連立方程式の場合は効率的に解けるが、誤差がある“ $\approx$ ”の連立方程式の場合は解くのが難しい。

CRYSTALS-Kyber の手順を図 12 および以下に示す⁶⁾。

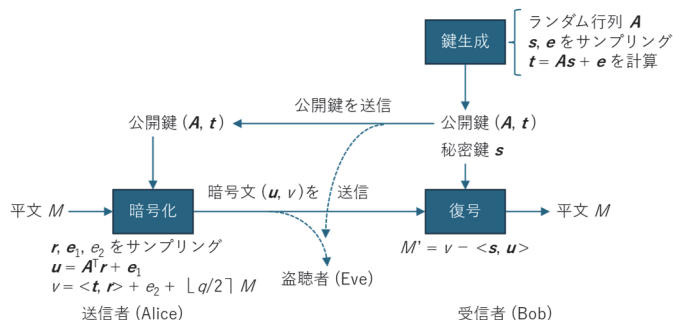


図 12 CRYSTALS-Kyber の手順

鍵生成:

- ・ ランダムに行列 $\mathbf{A}$ を生成する
- ・ q に比べて十分小さい $\mathbf{s}, \mathbf{e}$ を 2 項分布からサンプリングする
- ・ $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}$ を計算し、 $(\mathbf{A}, \mathbf{t})$ を公開鍵、 $\mathbf{s}$ を秘密鍵とする

暗号化:

- ・ q に比べて十分小さい $\mathbf{r}, \mathbf{e}_1, \mathbf{e}_2$ をサンプリングする
- ・ 平文 M に対して $\mathbf{u} = \mathbf{A}^T \mathbf{r} + \mathbf{e}_1$, $\mathbf{v} = \langle \mathbf{t}, \mathbf{r} \rangle + e_2 + \lfloor q/2 \rfloor M$ を計算し、 $(\mathbf{u}, \mathbf{v})$ を暗号文とする ($\lfloor \cdot \rfloor$ は整数への丸め、 $\langle \mathbf{t}, \mathbf{r} \rangle$ は $\mathbf{t}$ と $\mathbf{r}$ の内積)

復号:

- ・ 暗号文 $(\mathbf{u}, \mathbf{v})$ と秘密鍵 $\mathbf{s}$ を用いて $M' = \mathbf{v} - \langle \mathbf{s}, \mathbf{u} \rangle$ を計算し、 $|M'|$ が $q/4$ より小さければ 0、それ以外は 1 が復号文となる

4.2 耐量子計算機暗号への移行

耐量子計算機暗号への移行において、データ保護期間(暗号による保護が期待される期間)を X 年、インフラ移行期間(暗号処理の置き換えに要する期間)を Y 年、量子技術による暗号解読までの期間を Z 年とすると、図 13 に示すように $X + Y > Z$ の場合は

暗号が解読される。現代社会において公開鍵暗号は、さまざまな情報を保護するために広く利用されており、耐量子計算機暗号への移行は長い期間(Y 年)と労力を要する。暗号解読可能な量子コンピュータが近い将来に登場する可能性は低いと考えられているが、量子技術の革新的な発展が起こり得ることから、量子技術による暗号解読までの期間(Z 年)を見積もることが困難である。このため、耐量子計算機暗号への移行を可能な限り早期に開始する、インフラ移行期間(Y 年)とデータ保護期間(X 年)を短縮するような施策を行う、量子技術の開発動向を見守る等の対応が求められる⁷⁾。

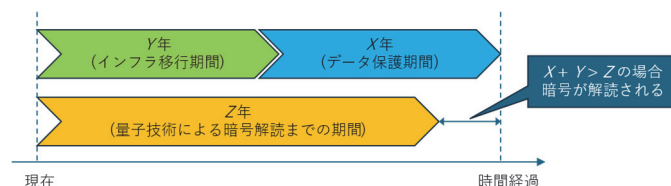


図 13 Mosca の定理

近年、耐量子計算機暗号の社会実装が進んでおり、例えば Apple 社はメッセージングアプリ iMessage に耐量子計算機暗号を導入することを発表している⁸⁾。iMessage では、従来の楕円曲線暗号と耐量子計算機暗号を併用することにより、もしどちらか一方の暗号が解読されても安全性が保たれるようにしている。

5 まとめ

盗聴されうる通信路を用いて安全に秘密情報をやりとりするために暗号が必要となる。しかし、共通鍵暗号では送信者と受信者の間で共通鍵を共有する必要がある、どのようにして安全に共通鍵を配送するかが課題である。この課題を解決したのが公開鍵暗号であり、暗号化と復号で異なる鍵を使用することにより、盗聴されうる通信路を用いて暗号化用の公開鍵を配送することができる。公開鍵暗号は多対多の通信にも適していることから、現在のインターネット上の暗号通信に広く用いられており、本稿では現在の公開鍵暗号として RSA 暗号、DH 鍵共有、楕円曲線暗号の基本原理解説した。ただし、これらの暗号は将来、大規模な量子コンピュータが実現すると効率的に解読できることが知られており、時間経過により陳腐化しない機密データにおいて脅威となる。このため、現在の RSA 暗号および楕円曲線暗号の後継として耐量子計算機暗号(PQC)の研究開発が行われており、鍵交換アルゴリズムとして格子問題に基づいた手法が NIST の PQC 標準に採択された。現代の情報化社会において公開鍵暗号は広く利用されており、インフラ移行には多大な労力を要するが、すでに PQC の社会実装が始まっており、今後 PQC への置き換えが進むものと考えられる。

参考文献

- 1) 辻井重男, “暗号の発展——軍事外交から社会基盤へ・隠す暗号から認証安号へ——,” 信学誌, vol.106, no.11, pp.971–976, Nov. 2023.
- 2) D.R. Stinson, Cryptography Theory and Practice, second edition, Chapman & Hall/CRC, 2002.
- 3) 清藤武暢, 四方順司, “公開鍵暗号を巡る新しい動き:RSA から楕円曲線暗号へ,” 日本銀行金融研究所/金融研究, July 2013.
<https://www.imes.boj.or.jp/research/papers/japanese/kk32-3-2.pdf>
- 4) CRYPTREC Report 2024, “暗号技術評価委員会報告,” March 2025.
- 5) National Institute of Standards and Technology, “Module lattice-based key-encapsulation mechanism standard,” Aug. 2024.
<https://doi.org/10.6028/NIST.FIPS.203>
- 6) 安田雅哉, “NIST 標準化の格子暗号方式の紹介,” 信学誌, vol.106, no.11, pp.982–987, Nov. 2023.
- 7) 伊藤忠彦, “耐量子計算機暗号への移行へ向けた課題と社会実装への論点整理,” 信学誌, vol.106, no.11, pp.1026–1030, Nov. 2023.
- 8) Apple Security Engineering and Architecture, “iMessage with PQ3: The new state of the art in quantum-secure messaging at scale,” Feb. 2024.
<https://security.apple.com/blog/imessage-pq3/>

執筆者



森 隆
先端技術研究所
技術企画部