応用楕円曲線論　プリント No.2

情報科学科　金子 晃

第２章　有限体の復習と公開鍵暗号のショートコース

この章では，楕円曲線の応用の最も重要なものの一つである，公開鍵暗号の概略を紹介し，目的をはっきりさせます．説明を効率的にするために，最初に有限体の復習をやっておきます．

§2.1 有限体

【体の定義】　集合 K が体とは，K に二つの２項演算 $+, \cdot$ が与えられており，

(a) $(K,+)$ は可換群，i.e.
  (1) 結合律 $\forall x,y,z \in K \quad (x+y)+z = x+(y+z)$
  (2) 単位元の存在 $\exists 0$ s.t. $\forall x \in K \quad x+0 = 0+x = x$
  (3) 逆元の存在 $\forall x \in K \quad \exists -x$ s.t. $\quad x+(-x) = (-x)+x = 0$
  (4) 可換律 $\forall x,y \in K \quad x+y = y+x$

(b) $(K^\times,\cdot)$ は可換群 (ここに $K^\times = K \setminus \{0\}$)，i.e.
  (1) 結合律 $\forall x,y,z \in K \quad (xy)z = x(yz)$
  (2) 単位元の存在 $\exists 1$ s.t. $\forall x \in K \quad x\cdot 1 = 1\cdot x = x$
  (3) 逆元の存在 $\forall x \in K^\times \quad \exists x^{-1}$ s.t. $\quad x\cdot x^{-1} = x^{-1}\cdot x = 1$
  (4) 可換律 $\forall x,y \in K \quad xy = yx$

(c) 分配律 $\forall x,y,z \in K \quad x(y+z) = xy+xz, \qquad (x+y)z = xz+yz$

群の一般論により，$0, 1$ がただ一つ, また，x に対して $-x$ や x^{-1} が一意に定まることが示せます．定義から，体は 0 と 1 の二つの異なる元を含みます．普通の数は $1,\ 2=1+1,\ 3=1+1+1,\ldots$ がすべて異なり，従って $\boldsymbol{N}, \boldsymbol{Z}, \boldsymbol{Q}$ を含むことが順に示せるので，有理数体 $\boldsymbol{Q}$ の拡大体となります．これを標数 0 の体と言います．しかし，体の公理からは $\exists p > 0$ について，

$$\underbrace{1+\cdots+1}_{p\text{ 個}} = 0$$

となる可能性が排除できません．このような p の最小のものは素数であることが容易に示せ，これを体の標数と呼びます．標数 p の体の例は，剰余環 $\boldsymbol{F}_p := \boldsymbol{Z}/p\boldsymbol{Z}$ で作れます．p が素数のとき，この剰余環で零元以外に乗法の逆元が常に存在することは，鳩の巣原理で初等的に示せます[1)]．

線型代数やアフィン幾何は，任意の体の上で議論できます．体 K 上の n 次元数ベクトル空間は K^n と書かれ，その点は $(a_1,\ldots,a_n)$ で表されます．原点の平行移動を許せば，同じ集合は体 K 上の n 次元アフィン空間となります．体の元を係数とする多項式は，体が何であっても意味を持つので，代数曲線論も同様に行えます．射影平面の定義も

$$(K^3 \setminus (0,0,0))/K^\times$$

で計算により行えば，問題なく一般の体に拡張できます．これが DesCartes さんのご利益です．

最も小さな体は 0 と 1 だけから成る $\boldsymbol{F}_2$ です．演算は，標数２の定義と 0 と 1 の定義から自然に定まり

$$0+0=0,\ 0+1=1+0=1,\ 1+1=0,\ 0\cdot 0 = 0\cdot 1 = 1\cdot 0 = 0,\ 1\cdot 1 = 1$$

となります．0 を偽，1 を真と解釈すると，この足し算は排他的論理和 XOR, 掛け算は論理積 AND と同じ規則です：

$$\text{偽}\,\mathrm{XOR}\,\text{真} = \text{真},\ \text{真}\,\mathrm{XOR}\,\text{真} = \text{偽},\ \text{偽}\,\mathrm{AND}\,\text{真} = \text{偽},\ \text{真}\,\mathrm{AND}\,\text{真} = \text{真}$$

[1)] 暗号に応用するには, 逆元を具体的に求める必要がありますが, それは拡張 Euclid 互除法で高速に計算できます.

代数学ではいろんな数が出て来るので，普通の整数 Z のことを有理整数と呼びます．これを二進法で表したものは，0 と 1 の文字列となります．このような二つの数 x, y の足し算は，各桁では上の規則による足し算ですが，繰り上がりがあるので複雑になります．情報科学では，これを F_2 上の数ベクトルのようにみなして，繰り上がりをせずに足し算を行う演算もよく用いられます．例えば，

$$10101 + 11011 = 01110$$

など．このような演算をビット毎の排他的論理和と呼び，$x \oplus y$ などで表します．

【有限拡大体の四則演算】　p を素数，$q = p^k$ とし，$\boldsymbol{F}_q$ を q 元の有限体とします．これは，素体 $\boldsymbol{F}_p = Z/pZ$ の上で既約な k 次多項式 $f(x)$ を一つ選べば，$\boldsymbol{F}_q = \boldsymbol{F}_p[x]/(f(x))$ として実現できます．この剰余環の元は，

$$c_{k-1}x^{k-1} + c_{k-2}x^{k-2} + \cdots + c_1x + c_0 \tag{2.1}$$

の形に一意的に表され，従って元の数が $q = p^k$ となることは，f の既約性から直ちに従います．これらの和は多項式の普通の和，積は多項式としての普通の積を計算した後で，$f(x)$ で割算した余りで置き換えるというものです．これらの演算で，この剰余環が体となることは，１変数多項式環において既約多項式が生成するイデアルが極大イデアルとなることから，一般論により自明ですが，0 以外の元に乗法の逆元があることを拡張 Euclid 互除法により直接確かめるのも，応用的な計算をするときは大切です．普通は x の代わりに $f(x) = 0$ の一つの根を α で表し，(2.1) の代わりに α の $k-1$ 次多項式として $\boldsymbol{F}_q$ の元を表示します．以上の議論で自明でないのは，$\boldsymbol{F}_p$ 上に k 次の既約多項式が必ず存在すること，それらのどれを用いても同型な体が得られること，ですが，これは信じて先に進みましょう (拙著：応用代数講義, 第８章参照). 後の便のため, 有限体に関して良く知られた事実をまとめておきましょう.

定理 2.1 (1) 有限体 $\boldsymbol{F}_q$ から零元を除いたもの $\boldsymbol{F}_q^\times$ は, 情報に関して巡回群を成す.
(2) 有限体 $\boldsymbol{F}_q$ の元は, ちょうど方程式 $x^q - x = 0$ のすべての根に対応する. この方程式は重根を持たない. (3) 任意の正整数 m に対し, 有限体 $\boldsymbol{F}_q$ 上で既約な m 次多項式 $f(x)$ が必ず存在する. その根 α に対し, Frobenius 置換 $\alpha \mapsto \alpha^q$ が推移的に作用し, $f(x)$ の分解体の Galois 群を生成する. (4) $q = p^m$, p 素数, とするとき, $x^q - x$ を $\boldsymbol{F}_p$ 上で既約分解すれば, $\boldsymbol{F}_p$ 上の m 次の既約多項式がすべて現れる.

実際, 多項式 $x^q - x$ は, 導函数が $-1 \neq 0$ なので重根を持ちません. (一般に, $f(x)$ の重根は $f(x)$ と $f'(x)$ の共通根でした.) さて, $\boldsymbol{F}_q$ の元のうち, 0 は明らかにこの多項式の単根ですが, それ以外の根は, 乗法群 $\boldsymbol{F}_q^\times$ に関する Lagrange の定理により $x^{q-1} - 1 = 0$ を満たします. よって重複するものが無ければ, 個数が一致するので, 根と体の元とは一対一に対応します. もし, $q-1$ の真の約数 r について $x^r = 1$ がすべての $x \in \boldsymbol{F}_q^\times$ について成り立ってしまうと, この方程式も重根を持たないので, 元の個数が不一致となります. よって $\boldsymbol{F}_q^\times$ は巡回群です. $\boldsymbol{F}_q$ を構成するときに使った $\boldsymbol{F}_p$ 上の既約多項式はもちろん $\boldsymbol{F}_q$ に根を持ちますが, それが $\boldsymbol{F}_q$ で１次因子の積に分解されることは, Frobenius 写像 $\alpha \mapsto \alpha^p$ がこの根に推移的に働くことから分かります. どの既約多項式を使っても同じ体が得られるなら, m 次の既約多項式がすべて $x^q - x$ の因子となる訳です.

【有限体の代数的閉包】　同じようにして $\boldsymbol{F}_q$ の拡大体 $\boldsymbol{F}_{q^m}$ が作れます．有限体の構造が元の個数だけで決まることから，$K = \boldsymbol{F}_q$ の代数的閉包，すなわち，$\boldsymbol{F}_q$ を含む最小の代数的閉体は，単に

$$\overline{K} = \bigcup_{m=0}^{\infty} \boldsymbol{F}_{q^m} \tag{2.2}$$

で定義できます．二つの体の和集合は必ずしも体にはならないので，この定義はちょっと心配かもしれませんが，一般に，$m \mid n$ なら，$\boldsymbol{F}_{q^m} \subset \boldsymbol{F}_{q^n}$ となるので，上は

$$\overline{K} = \bigcup_{m=0}^{\infty} \boldsymbol{F}_{q^{m!}}$$

と同じものです．$m \leq n$ というだけでは必ずしも $\boldsymbol{F}_{q^m} \subset \boldsymbol{F}_{q^n}$ とはならないことに注意しましょう．それは，包含関係があると，$\boldsymbol{F}_{q^n}$ が部分体 $\boldsymbol{F}_{q^m}$ 上の線型空間となり，その次元を l とすれば，元の個数の間に $q^n = (q^m)^l$ という関係が生じるからです．逆に，この関係があるとき必ず包含関係があることは，勝手に作った二つの体ではそう自明ではありませんが，同型な体の中で $\boldsymbol{F}_{q^n}$ が $\boldsymbol{F}_{q^m}$ の l 次拡大となっているものがあるので，いつでも包含関係にあるとみなせるのです．(2.2) は $\boldsymbol{F}_q$ の代数的閉包であるだけではなく，標数 p の任意の有限体に対する共通の代数的閉包となっています．

§2.2 公開鍵暗号

暗号とは，秘密裏に通信を行うことですが，諜報員が直接秘密文書を届けることができれば，暗号は不要です．暗号は，途中で盗まれても読まれない，あるいは，伝播などで傍受されても読まれないように，安全でない通信路を通して秘密通信を実現するときに必要となるものです．

Alice が Bob に送りたい秘密の平文を m とします．Alice と Bob は A, B (日本語で甲，乙に相当) というところを親しみを込めて洒落て言ったものです．以下 m は正整数とします．なぜ整数としてよいかは，あらゆるデータが最終的には正整数で表現できるからです．うそだと思う人は，数学科の２３年向けの僕の講義『数理逍遥４』の初日のレジュメを参照して下さい．暗号の最も基本的な形態は，m と同じ長さの乱数 r を Alice と Bob が共有し，Alice は $m \oplus r$ として Bob に送り，Bob は $(m \oplus r) \oplus r = m \oplus (r \oplus r) = m$ により，もとの文を得るというものです．ここで，$\oplus$ はビット毎の XOR 演算を表し，m や r を二進数で表したものを F_2 上のベクトルとみて成分毎の足し算をしたものと同等です．乱数 r が暗号の鍵です．この方法は，発明者の名を取って Vernam 暗号と呼ばれ，一回だけの通信では絶対の安全性を持っていますが，更に別の平文を暗号化しようとして，同じ r を用いたら，たちまち解読されてしまいます．新たに r を交換するくらいなら，m が直接送れる道理なので，あまり実用的ではありません．実用的な共通鍵暗号方式は，r を Alice とボブがある決まった方法で，より短い種 (シード) から同時に生成し，使うものです．このように，従来の暗号は，送信者と受信者が同じ鍵を共有することが，秘密通信の絶対条件でした．

公開鍵暗号は Diffie-Hellman 1976 が原理を発表したもので, 今日これが公開鍵暗号の公式の誕生年とされています. しかし, イギリス諜報部は秘密裏に同じ概念に到達していたという噂が以前から有り、最近ではその証拠資料も発表されたようです. が，もちろん発表順優先の学問の世界では，発見の権利を要求できません．そういうところで働いていた数学者は可哀想ですね．

Diffie-Hellman による公開鍵暗号発見の経緯は面白いものです．+++年にアメリカ政府は一般の商用目的で使う暗号の標準として DES (Data encryption Standard) を補修し，IBM の応募作品に手直しして制定されました．しかし，彼らは，このように政府が勧める標準暗号などというものには，裏があるに違いない．政府だけに知られた解読機構を含んでいるのではないかと疑い, その種の機構の可能性を調べているうちに, 副産物として公開鍵暗号の考えに到達したのでした.

現代暗号の理論では逆写像の計算が不可能な写像，一方向性函数 (one-way function) がいろいろなところで重要な役割を果たします[2)]. 数学的には, 有限集合の上の写像が一対一なら逆写像は確定しますが, 集合の元の個数が多いと逆写像を計算するのが実質的に計算不可能となるような函数が存在します. このようなものは UNIX システムのパスワードの暗号化などで使われており, これはスーパーユーザと言えども, 暗号化されたパスワードから元のパスワードを知ることはできません. クラッカーは, 通常, パスワードを推測しそれを暗号化してみて, パスワードファイルに記された暗号化パスワードと一致するかどうかを見て, 他人のパスワードを手に入れます. この操作は辞書を用いてコンピュータによりものすごいスピードで行えるので，他人に類推できるような単語をパスワードに使ってはいけないのです．

公開鍵暗号で使われる一方向性函数は, 一般の人には逆写像の計算が不可能だが, ある秘密の情報を持った人には逆写像が計算できる, というようなものでなければなりません. これを落とし戸 (trap door) 付き一方向性函数と呼びます. すなわち, 一方向性函数は暗号化鍵 K_E をパラメータとする写像の族 ENC_{K_E} で, これと対になった復号鍵 K_D を用いると, 逆写像 DEC_{K_D} が容易に計算できることが必要です.

[2)] 純粋数学の外の世界では，“写像” という言葉は難しいと思われているのか，あまり使われず，“函数” と呼ばれることが多いようです．

この際, 暗号化鍵 K_E を知ってもそれから復号鍵 K_D を計算するのが不可能ならば, 暗号化鍵を公開鍵 (public key) として公開し，誰にでも暗号文を作らせ, 受信者だけが秘密の復号鍵，秘密鍵 (private key) で復号できるようになります. これが公開鍵暗号の原理です. 古典的な暗号アルゴリズムでは, K_D は K_E と同じものであることが普通なので, このようなことを可能にするためには新しい暗号アルゴリズムが必要になります. 本当にそんなものが可能でしょうか？それが数学の力で実現されたのです！

公開鍵暗号は，今までにいろんなものが提案されていますが，広く使われているのは，RSA 暗号と ElGamal 暗号です．公開鍵暗号の原理の発明者自身は，公開鍵暗号の例を与えることができず，原論文では Diffie-Hellman の鍵交換法というものだけを提案しました．以下, この三つについて順に紹介します.

§2.3 RSA 暗号

RSA はこの暗号方式の発見者, Rivest, Shamir, Adleman の三人の頭文字です. 1977 年に発見された史上最初の公開鍵暗号で, 巨大整数の素因数分解の計算の困難さを利用しています.

RSA 暗号の原理

☆ p, q : 大きな素数とし, $n = pq$ をブロック長とする.
☆ $\varphi(n) = (p-1)(q-1)$: $1 \le x \le n-1$ のうち n と互いに素な数の個数,
☆ d, e を $de \equiv 1 \mod \varphi(n)$ に選ぶ.　　公開鍵： n, e. 秘密鍵： d
☆ 暗号化写像： $x \mapsto x^e \mod n$
☆ 復号写像: $y \mapsto y^d \mod n$

この暗号の数学的な説明としては公開鍵は e だけで, n は公開鍵というよりもむしろ共有データと言った方が分かりやすいと思われるかもしれませんが, RSA 暗号の場合は鍵のペアを作るのに n だけでは足りず, その因数分解も知る必要があるので, n を共通にしてみんなで使うということはできません. また, 誰か一人が n の因数分解情報を占有して, みんなのために鍵 d, e のペアを作ってあげることにした場合, 2組の鍵 (d_1, e_1), (d_2, e_2) から n の素因数分解が簡単にばれてしまうので, 2人が共謀する可能性がある現実世界では実用になりません. 以上のような理由で, 暗号学では n, e を公開鍵と呼ぶのが普通です.

RSA 暗号のアルゴリズムの正当性

$n = pq$ は素数の積とし, $de \equiv 1 \mod \mathrm{LCM}(p-1, q-1) \Longrightarrow \forall x$ について $x^{de} \equiv x \mod n$ となることをいいます. ここに LCM は最小公倍数を表します (後注参照). p, q は互いに素なので, $x^{de} \equiv x \mod p$, $x^{de} \equiv x \mod q$ を言えば $x^{de} \equiv x \mod pq$ が言えます. $\mathrm{LCM}\{p-1, q-1\} = (p-1)m$ と書けることに注意しましょう. 従って $\exists k$ s.t. $de = 1 + k\mathrm{LCM}\{p-1, q-1\}$ より

$$x^{de} = x \cdot (x^{\mathrm{LCM}\{p-1,q-1\}})^k = x \cdot (x^{p-1})^{mk}.$$

ここで $x \neq 0$ なら, Fermat の小定理により $x^{p-1} \equiv 1 \mod p$. 従って

$$x^{de} \equiv x \cdot 1^{mk} \equiv x \mod p.$$

また $x = 0$ なら最初から $x^{de} = 0 \equiv x \mod p$ が成り立っています. q についても同様です.

☺ オリジナルの RSA 暗号では $de \equiv 1 \mod (p-1)(q-1)$ という条件でしたが, その後 $de \equiv 1 \mod \mathrm{LCM}(p-1, q-1)$ で十分なことが分かり, 現在ではこちらが普通に用いられています. この方が計算サイズをいくらか小さくできます. $de \equiv 1 \mod (p-1)(q-1)$ ならもちろん $de \equiv 1 \mod \mathrm{LCM}(p-1, q-1)$ も成り立つので, 上の証明はオリジナルな設定にも通用します. $\varphi(n) = (p-1)(q-1)$ は Euler の函数ですが, $\mathrm{LCM}(p-1, q-1)$ は Carmichael 函数と呼ばれます. Euler の函数を使うのなら, Fermat の小定理の代わりに Euler の定理を使って, $x^{\varphi(n)} \equiv 1 \mod n$ で一発じゃないかと思われるかもしれませんが, 平文 x は n と互いに素にならない確率がものすごく小さいが残るので, これだけでは証明としては不完全です. (まあ，そんな平文が見付かったら暗号が破れてしまうので，実用的な証明としてはこれでも十分な訳ですが．(^^;)

RSA 暗号の安全性

この暗号の安全性の根拠は, “ n と e から d を計算するのは n の素因数分解と同程度に困難？”という RSA 仮説です. この仮説は広く信じられているので, 素因数分解が難しければ RSA 暗号は安全だろうと殆どの人が考えていることになります.

n の素因数分解については, 現在知られているものとしては準指数時間のアルゴリズムが最良で,

☆ 楕円曲線法. $O(e^{c(\log n)^{1/2}(\log\log n)^{1/2}})$

☆ 数体篩法. $O(e^{c(\log n)^{1/3}(\log\log n)^{2/3}})$

などがあります. RSA 暗号を作るときはこれらの方法で因数分解が不可能な大きさの n を選ばねばなりません. 現在のお勧めは 1024 ビットです.

RSA 暗号を実際にプログラムで実装するときに必要となることは

☆ 多倍長演算ルーチン, 特に冪乗計算の高速化,

☆ 巨大素数の生成アルゴリズム

などです. これらに興味の有る人は，後期に開講される情報科学科向けの学部の講義『符号理論』に出てみましょう！単なるおもちゃなら, 初等整数論の演習として, 普通の大きさの整数を使って初等代数学や離散数学の練習問題でやったかもしれませんね.

RSA 暗号の特許

RSA 暗号は発明直後に, 作者の Rivest, Shamir, Adleman が特許をとり, 三人の頭文字を取った RSA 社を興して製品化しました. しかしこの特許は 2000 年の 9 月に 20 年の期限が切れて, 今は誰でも自由に使えるようになっています. 従ってフリーの ssh や ssl, あるいは PC UNIX の組み込み暗号などでも, 今は RSA が選択可能です.

§2.4 Diffie-Hellman の鍵交換法

公開鍵暗号のアイデアの創始者 Diffie-Hellman が 1976 年に提唱したもので, 離れた二人が安全でない通信路を用いて，秘密鍵暗号の暗号化鍵 (= 復号鍵) を共有する方法を示したものです. 数学的には離散対数問題を用います. 離散対数問題とは

G: 有限群, $g \in G$: 位数が大きな元のとき,
$a \in \boldsymbol{N}$ に対し $x = g^a$ の計算の逆を計算, i.e. $x \mapsto a = \log_g x$ の計算をせよ

という問題で, 一般の群に対してはシラミ潰しに探すしか方法が無い？と予想されています. シラミ潰し探索はデータのビット長の指数時間の計算量 $O(e^{cn})$ になります.

Diffie-Hellman の秘密鍵共有方式：

☆ 共有データ : g

☆ A (Alice), B (Bob) はお互いに秘密鍵 $a, b \in \boldsymbol{N}$ を用意し, 相手に g^a および g^b を送る.

☆ それぞれは相手から送られてきたデータを元に

A : $(g^b)^a = g^{ab}$ を

B : $(g^a)^b = g^{ab}$ を

計算して同一の秘密鍵 g^{ab} を共有し, それを元に秘密鍵暗号を使う.

この方式の安全性の根拠は, g^a と g^b から g^{ab} を求めることは離散対数問題と同程度に困難？という Diffie-Hellman の予想です.

なお, 離散対数問題自身の困難さについては, 群 G が特殊だと比較的速い計算がいろいろ知られています. 最も基礎となる有限体の乗法群に対しては, 準指数時間 $O(e^{(1+o(1))\sqrt{\log p \log\log p}})$ のアルゴリズムが存在します (index calculus). 一般の注意深く選ばれた楕円曲線の加法群に対しては, 今のところ指数時間と予想されており, これが楕円曲線暗号の流行する根拠となっています.

§2.5 ElGamal 暗号

ElGamal が 1985 年に発表したもので, やはり有限群の離散対数問題を基礎にしています.

ElGamal 暗号のしくみ：

☆ 共有データは, G: 有限群, $g \in G$: 位数 n の大きな元 (生成元にとるのが普通)
A は $0 < a < n$ をランダムに選び, a を秘密鍵, g^a を公開鍵とする.
他人から A への暗号化メッセージの送信法：

$$\left.\begin{array}{l} P \in G: \text{ 平文のブロック} \\ k: \text{乱数} \end{array}\right\} \quad \Longrightarrow \quad \text{暗号文 } (g^k, Pg^{ak})$$

☆ 暗号文は a を知らなくても g^a だけから計算可能！
☆ 復号：$g^{ak} = (g^k)^a$ を計算し, $P = (Pg^{ak})/g^{ak}$ を復元.

これも Diffie-Hellman 予想が正しく，かつ離散対数問題が難しければ安全です.

Q 元祖 ElGamal 暗号は $G = \boldsymbol{F}_q^\times$ (有限体の乗法群) で, 平文の埋め込みは容易ですが, この場合の離散対数問題に対しては準指数時間のアルゴリズムが知られており, 現実にも鍵長を長めにしないと安全性が保たれません. G として楕円曲線上の F_q 有理点の成す加法群を取ったものを楕円 ElGamal 暗号と呼びます. なお, 演算 Pg^{ak} は逆が一意に計算可能な任意の演算で代用できます. 例えば, P と g^{ak} の XOR でも構いません. その方が計算時間は圧倒的に速くなります.

§2.6 電子署名と認証

現代暗号, 特に公開鍵暗号はインターネットの世界で多くの応用を持ちます. 特に，公開鍵暗号方式を用いると，電子署名 digital signature) やネットワークを通した認証 (authentication, 字義通りでは正当性証明) が可能になります．このことは，Diffie-Hellman の原論文に既に書かれており，彼らの寄与の最も重要な部分とさえ言われています．

電子署名と認証は，(秘匿目的の) 暗号と並び，暗号の要素技術 (暗号 primitive) と呼ばれているくらい重要なものです. これらを部品として用い, 電子投票, 電子オークション, 電子決済など, 更にいろいろなセキュリティの機能・方法が構築されます. それらの応用技術は一般に暗号プロトコルと総称されています.

【認証】 公開鍵暗号を利用して相手を確かに本人であると特定する方法です. ここでは公開鍵暗号を用いて暗号化し送信するメッセージに認証を付加する方法を説明します.

1) お互いに秘密鍵と公開鍵を持つ:
   A の秘密鍵 K_D^A, 公開鍵 K_E^A; B の秘密鍵 K_D^B, 公開鍵 K_E^B
2) 相手の公開鍵で暗号化し, それに自分の秘密鍵で更に暗号化を施して相手に送る.
   A から B へ $M \mapsto \mathrm{ENC}_{K_E^B}(M) \mapsto \mathrm{DEC}_{K_D^A}(\mathrm{ENC}_{K_E^B}(M)) = C$ が送られる
3) 受信者はまず相手の公開鍵で復号し, 得られた暗号文を自分の秘密鍵で復号する. i.e. B は $C \mapsto \mathrm{ENC}_{K_E^A}(C) \mapsto \mathrm{DEC}_{K_D^B}(\mathrm{ENC}_{K_E^A}(C)) = M$ を得る. これで平文が得られれば相手は公開鍵を発表した当人と推測できる.

暗号化写像も復号写像も, 数学的には互いに逆写像というだけなので, ひっくりかえして使っても問題無いところがミソです.

【電子署名】 デジタル文書は誰でも作成できてしまい, 筆跡の概念がありません. ある文書が本人の書いたものであることを証明できるでしょうか？またファイルは紙の文書と違い, 誰かが内容を改竄してもその痕跡が全く残りません. (ファイルの作成日付などは容易にごまかせます.) デジタル文書を改竄が不可能にできるでしょうか？もしこれらが不可能なら, 契約書などは永久に紙の上に書いて署名捺印 (複数葉より成るときは, 各ページに割印) したものしか通用しないことになり, 電子社会とは言えません.

普通の署名が持つ役割として次の条件があります：
① 本人にしか作れない.
② 検証は誰にでもできる.
③ 署名された文書の改竄ができない.

これを電子的に実現するのに公開鍵暗号が利用できます：
① 本人しか知らない情報を使って署名作成.

② 誰でも知っている情報を使って署名を検証.

それぞれ, これらを秘密鍵, 公開鍵で行えばよいのです.

署名生成：もとの文書 m を秘密鍵で暗号化したもの $m \mapsto \sigma = \mathrm{DEC}_{K_D}(m)$ をもとの文書とペアにし (m, σ) を署名付き文書とする.

署名検証：公開鍵で復号し $(m, \sigma) \mapsto \mathrm{Enc}_{K_E}(\sigma)$ と m が一致するかどうかを見る.

上の電子署名方式を実用化しようとした場合には, 以下のような問題点が浮かび上がります：

0) まず，商法上電子署名を施された電子文書が契約書として通用することを保証しなければなりません．(その昔，電線から勝手に引き込み線を作って電気を不正使用していた人が窃盗罪で検挙されたことがありました．法律の解釈に厳格なドイツでは，“ 窃盗罪は物を盗むものと書いてあるが，電気は物ではない ”と判断され無罪になったそうです．日本では法律を拡張解釈して有罪でした．

1) 本人の署名用公開鍵であることをどうやって認知するか？
A の名前をかたった第三者が“ これは A の公開鍵だ ”と言いふらしても, 本人の顔を知らないのが前提の電子社会では, なかなかうそが分かりません. 印鑑の場合でも他人が A だと偽って契約をしようとすることは有り得ますが, それを防ぐ仕組みとして, 重大な契約には住民票の所在地の役所に印影を登録した, いわゆる実印と呼ばれるもののみ使用が認められており, 契約時には役所が発行した印鑑証明書を一緒に提出する慣習ができています. そこで印鑑登録の代りとなる電子認証局を作り, 公開鍵をそこに登録する，という方法が考えられます.
これらの準備のため，アメリカの大多数の州と, ドイツ, シンガポールなどで次々に関連事項が法制化され，少し遅れて日本でも平成１２年５月に“ 電子署名及び認証業務に関する法律 ”が成立し平成１３年４月から施行されています. ただし日本では法律の題からも想像されるように，認証局は必ずしも国家が運営するとは限らず, 民間でやりたい者にやらせるという立場でそのための法的整備をしたのです．興味の有る人は次の URL を参照してください：
http://www.meti.go.jp/policy/netsecurity/digitalsign.htm

2) 文書を丸ごと暗号化したのでは, 本文よりも長い署名になってしまいます. 署名というのは短いに越したことはありませんが，文書の一部だけ使うと, その他をちぎって改竄される恐れがあります. そこでハッシュ函数 (hash function) というものを利用して，元データからその各部分が本質的に関与して，かつデータ量を減らしたようなものを作り, それを暗号化したものを添付するように改良します.
ハッシュ函数の利用は準同型を利用して第三の文書の作成を予防できるという効果もあります： もし単純な RSA による署名だと, 同一人が (m_1, σ_1), (m_2, σ_2) という二つの署名付き文書を公表していたとき, 第三者が $(m_1 m_2, \sigma_1 \sigma_2)$ という合成文書を作り出しても, 検証に通ってしまいます.
$\because\ \sigma_1 \sigma_2 = m_1^d m_2^d = (m_1 m_2)^d$
こうしてできた文書が契約等として意味のある (日本語として読める) ものになる確率は極めて小さいものの, 計算機による自動チェックなどでは見付からないので, 使われる状況によっては危険です.

以上に解説した方式, 実用化するときは更にいろいろな問題点を解決する必要が有りますが, 数学の講義としてはこの程度で十分でしょう.

DSS (標準電子署名)

標準暗号と同様, 電子署名も標準化しようとして NIST が制定した DSA (Digital Signature Algorithm) の標準が DSS (Digital Signature Standard) です. 制定当時は RSA 暗号の特許がまだ効いていたので, ElGamal 暗号を基礎にした方式が採用されました. ハッシュ函数には SHA-1 (シャーワン) を用いています. アルゴリズムの正確な定義は

http://www.itlnist.gov/fipspubs/fip186.htm

を見てください. 楕円曲線暗号を用いる変種などの他, 現在は特許の切れた RSA 暗号を用いる変種なども定義されています.