Lec 07. 公钥密码体系 (1) RSA

公钥密码：非对称密码或双钥密码，为简化密钥分配管理并实现签名等功能出现，由 Diffie 和 Hellman 提出，当时只是一种想法，并没有给出实际例子。

- 原理：每个用户拥有一对公私钥对 (P_k, S_k)，公钥 P_k 公开，私钥 S_k 保密；已知公钥算法与 P_k，得不到 S_k 的值（计算上不可行）
  - 用于加密解密：$C = E_{P_k}[M], M = D_{S_k}[C]$；
  - 用于数字签名：签名 $M||Sig_{S_k}[M]$；验证 $M = Ver_{P_k}[Sig_{S_k}[m]]$。
- 不对称性是公钥最重要的性质，可以用来保证**真实性、不可否认性**。

公钥密码系统的要求：

- 每个用户可以方便快捷地产生自己的公私钥对 (P_k, S_k)；
- 可以方便快捷地利用公钥 P_k 对 M 加密 $C = E_{P_k}[M]$；
- 如果拥有私钥 S_k 可以方便快捷地对某个密文 C 解密 $M = D_{S_k}[C]$；
- 对于其他人，已知公钥 P_k 无法得到 S_k；已知公钥 P_k 与密文 C，无法得到 S_k。

单向函数：给定 x，计算 $y = f(x)$ 是容易的，而计算 $x = f^{-1}(y)$ 是困难的，即计算上不可行。并且求解问题的困难与否应该是客观的，不应该依赖当事人的知识和能力，考虑抽象计算模型 Turing 机的计算复杂性。

- 单向函数的例子：离散对数问题、大整数分解问题等；
- **陷门单向函数**：对于单向函数 $y = f(x)$，若存在 δ 使得已知 δ 时对任何给定的 y 只要相应的 x 存在，则计算 $x = f^{-1}(y)$ 是容易的，则 $y = f(x)$ 为陷门单向函数，δ 是陷门信息。

MH scheme（Merkle-Hellman）：背包问题 KP $(a_1, a_2, \cdots, a_n)$ 属于 NPC 问题，超递增背包问题 SKP（一个数大于前面所有数之和）$(a'_1, a'_2, \cdots, a'_n)$容易；进行 $a_i \equiv wa'_i \pmod n$ 可以将 KP 转化为 SKP，则用户公钥为 PK，私钥为超递增背包 SKP。

- 加密：比特 $(x_1, x_2, \cdots, x_T)$ 加密为 $c \equiv \sum_i a_i x_i \pmod n$；
- 解密：令 $c' = w^{-1}c$ 于是得到 $\sum_i a'_i x_i \pmod n$ 为 SKP 问题，可以解出 $(x_1, x_2, \cdots, x_T)$。
- 已被攻破，现有体系下不安全。

RSA算法

- 评价：把纯数学用简单的方式解决现实问题，并且解决地如此之好。
- 明文空间 M 与密文空间 C 均为 $\mathbb{Z}_n$；分组为 $\log_2 n$ 比特。
- **密钥生成**：
  - 选择不同的素数 p, q 并计算 $n = pq, \varphi(n) = (p-1)(q-1)$；
  - 选择整数 e 使得 $(\varphi(n), e) = 1$，其中 $1 < e < \varphi(n)$;
  - 计算 d 使得 $d = e^{-1} \bmod \varphi(n)$。
- **公开参数**：n，公钥 e；
- **保密参数**：p, q，私钥 d。
- **加密**：$C = M^e \pmod n$；
- **解密**：$M = C^d \pmod n$。

RSA 算法的正确性验证：由算法有 $de \equiv 1 \pmod{\varphi(n)}$，又根据欧拉定理推论（即使 $(M,n) \neq 1$ 也成立的推论）有

$$M^{ed} \equiv M \pmod n$$

（证明分情况，若 $p \nmid M$ 且 $q \nmid M$，则 $(M,n)=1$ 显然成立；若 $p \mid M, q \nmid M$，则令 $M=kp$ 有 $(k,q)=1$ 与 $(M,q)=1$，则必有 $M^{q-1} \equiv 1 (\bmod m)$，从而 $M^{ed} \equiv M \pmod q$。即 $(kp)^{ed} = kp + tq$，则 $p \mid tq$，又由于 $(p,q)=1$ 有 $tq = t'pq$，从而 $(kp)^{ed} = kp + t'pq$，即 $M^{ed} \equiv M \pmod{pq}$；其余情况类似）

于是由 $C \equiv M^e \pmod n$ 有 $C^d = M^{ed} \equiv M \pmod n$。

RSA 的安全性：如果 n 的素数分解是已知的，那么 RSA 问题被攻破；但"攻破 RSA 与分解 n 是多项式等价的"猜想没有被证明！

- 针对 RSA 的攻击非常丰富：数学攻击、物理攻击、基于运行错误的攻击、基于系统使用错误的攻击。
- **强力攻击**：尝试分解 n（RSA 问题的困难性不会比大整数分解问题的困难度高）。
- **参数选择**
  - n 的长度至少是 1024 比特，或 2048 比特或更大；
  - 建议 p,q 需要大，且 $|p-q|$ 也需要大，且 p,q 为强素数（早期建议），即 $p=2p_1+1, q=2q_1+1$；
  - EDI 国际标准规定 $e=2^{16}+1$；e 应足够大，虽然 e 小时运算快，但存在问题；
  - $d > n^{1/4}$。
- **参数不当攻击**：选择 p,q 时，应注意 p,q 为随机素数且不包含在素数表中，且两个素数不能太接近。如果两个素数太接近，则可以如下攻击：$n = pq = \frac{1}{4}[(p+q)^2 - (p-q)^2]$，则后项很小，$(p+q)/2$ 只比 $\sqrt{n}$ 大一点，然后逐个检查大于根号的整数 x，直到 $x^2 - n$ 也是平方数 y^2，于是 $p = x+y, q = x-y$。
  - 解决方法：选择 p,q 使得其二进制表示长度有几个比特不同。
- **共模攻击**：指通信系统中使用相同的 n 且存在两个用户的公钥 e_1, e_2 互素，则可以由这两个用户对同一条明文的不同加密结果恢复出原始明文，即 $c_1 = m^{e_1} \pmod n$，$c_2 = m^{e_2} \pmod n$，攻击者（系统外）知道 e_1, e_2, n, c_1, c_2，根据扩展欧几里得算法，存在 s,t 使得 $te_1 + se_2 = 1$，于是 $c_1^t c_2^s = m^{e_1 t + e_2 s} \equiv m \pmod n$。
  - 解决方法：不能用同一个 n 生成密钥。
- **小 e 攻击**：e 很小时，若使用广播加密（用同一个 e 对同一个消息 m 加密再发给多个人）下遭遇的攻击。设 $e=3$ 时，设有三个成员公钥 $e=3$ 但 n 不同，分别为 n_1, n_2, n_3，则 $c_i = m^3 \pmod{n_i} \quad (i=1,2,3)$，因为 n_1, n_2, n_3 一般是互素的，因此由中国剩余定理有存在 C 使得 $m^3 \equiv C \pmod{n_1 n_2 n_3}$，由于 $m < n_1, m < n_2, m < n_3$，从而 $m^3 < n_1 n_2 n_3$，则 $m = C^{1/3}$。
  - 解决方法：选择大的 e，且 RSA 的 message 应该足够长。
- **计时攻击**：利用指数中某一位为 0 或 1 时硬件加密速度不同进行分析。
  - 解决方法：不变的幂运算时间，保证所有过程返回结果前执行时间相同；
  - 随机延时；
  - 隐蔽：在密文上乘随机数。

RSA 公钥算法的实现：利用快速幂即可，可选用 $e = 3, 17, 65537$ 等，$x^c \pmod n$ 的复杂度为 $O(kl^2)$，其中 $k = \log_2 c, l = |n|$。

RSA-OAEP：假设 RSA 算法模 N，且 $|N|=n$，选择 k_0,k_1 且 message 长度仅有 $n-k_0-k_1$ 的长度，设 r 为 k_0 长度的随机数，则

$$
\begin{gathered}
L_0=(m||0^{k_1}),\quad R_0=r\\
L_1=L_0\oplus H_1(r),\quad R_1=R_0\\
L_2=L_1,\quad R_2=H_2(L_1)\oplus R_1\\
X=L_2,\quad Y=R_2
\end{gathered}
$$

为两轮 Feistel 结构，最后将 $X||Y$ 进行 RSA 加密输出，同时返回随机数 r 以及 k_1 等信息。解密时只需要先解密得到 X,Y，然后使用 Feistel 结构进行反向计算即可。

- **优点**：非确定性（r 随机），验证性（选择密文攻击下，攻击者随机生成可以通过有效性验证的密文的概率为 $\frac{1}{2^{k_1}}$。
- 目前具有可证安全性，进入工业标准的为 OAEP+。

对 RSA 的模数攻击：理想情况为每个 n 对应唯一 p,q（每个素数的度为 1），实际情况，找到许多至少 2 条边的组件，甚至出现了 K_9 完全图……

结论：安全的 $n=pq$ 值需要：

- p,q 从未出现过；
- 产生素数的随机种子长度必须是目标安全标准长度的两倍，防止素数的 regeneration；
- 存在许多产生安全素数的方法，但在实际应用中常常使用没有新鲜信息的低熵种子导致问题，例如
    - 同一随机数发生器在第一次运行时输出同一素数，第二次运行时输出不同素数；
    - 出现公共素因子等等。
    - 不过，这可能可以解释深度为 1 的树的出现，无法解释其他情况。

关于公钥密码的误解

- 公钥密码比对称密码更安全；
- 对称密码已经过时；
- 对称密码中用户与 KDC 握手异常麻烦，而在公钥中很简单。

Lec 08. 公钥密码体系 (2) Diffie-Hellman 密钥交换与 ElGamal

基于离散对数问题的加密体制：此处仅研究有限域 F_{p^n} 上乘法群的离散对数问题，p 是素数且 $n\in\mathbb{N}_+$。给定 a,b 且 a,b,p^n 公开，目标是找 s 使得 $a^s=b$；当 $n=1$ 时，如果找 $\{g_i\}^{i=0,1,\cdots,p-1}=F_p-\{0\}$，则 g 为原根；F_p 的原根可以有效计算。当 $p-1$ 只含有小素数因子时，计算满足的 $y\equiv g^x\pmod{p}$ 的 x 是容易的；当 $p-1$ 有大素数因子时则困难。故令 $q=2p+1$ 且 p 为大素数，则在 F_q 上离散对数问题难解。

Diffie-Hellman密钥交换算法：是公钥体制的思想来源，但本身并不是公钥加密算法，是基于公钥加密的密钥分配算法。不可以被用于交换任何消息（非加密算法），而是用于建立一对仅交互双方知道的密钥。密钥的值依赖于双方参与者的公开信息和似有信息，安全性依赖于离散对数问题的困难性。

- **初始化**：用户得到全局参数：大素数 p 与模 p 的原根 g，且 $p-1$ 含有大素数因子；各个用户生成各自公私钥对，选择私钥 $x<p$ 并计算公钥 $y=g^x \bmod p$；然后各个用户公开自己的公钥。

- **密钥交换**：适用于双方密钥交换。通信双方的会话密钥 $K \equiv y_B^{x_A} = y_A^{x_B} = g^{x_A x_B} \pmod p$。除非某一方更新公钥，否则会话密钥保持不变；攻击者必须解离散对数问题以求出会话密钥 K。具体来说，过程如下：

  ```
  (1) A generate xA, yA, and send yA to B;
  (2) B generate xB, yB, calculate K and send yB to A;
  (3) A generate K
  ```

- **陷门单向函数**：给定 g^a, g^b 求 g^{ab} 是困难的；但给定 a 或 b 会让问题变得简单。这个问题被称为计算性 Diffie-Hellman 问题。
- **中间人攻击**：攻击者 Malice 装作 B 与 A 通信并装作 A 与 B 通信，即

  ```
  (1) A send yA to B, and intercepted by M;
  (2) M send yM to B and generate Ka;
  (3) B generate Kb and send yB to M;
  (4) M send yM to A and generate Kb;
  (5) A generate Ka.
  ```

  于是，`M` 知道 `Ka, Kb`，可以同时与两人通信，“中间人”攻击。

ElGamal 加密算法：基于 Diffie-Hellman 陷门单向函数，是概率加密。

- **初始化**：生成系统参数 p, g 且 $p-1$ 含有大素数因子，g 是群 F_p^* 的乘法生成元。Alice 选择私钥 $x_A \in F_p^*$ 并公开公钥 $y_A = g^{x_A} \bmod p$。
- **加密**：需要加密消息 $M < p$ 给 Alice，发送方随机选择 $k \in F_p^*$，密文 $C = (C_1, C_2)$，其中 $C_1 = g^k \bmod p$，$C_2 = y_A^k M \bmod p$。
- **解密**：Alice 用自己的私钥 x_A 计算 $M = \frac{C_2}{C_1^{x_A}} = M$。
- **CPA（选择明文） 攻击**（*）：由 Euler 判别条件，一定有 $g \in QNR_p$，即 g 是模 p 的二次非剩余；攻击者选择 m_0, m_1 使得 $m_0 \in QR_p, m_1 \in QNR_p$。如果 $y \in QR_p$，则由 C_2 是否 $\in QR_p$ 可以确定 m_0 或 m_1；如果 $y \in QNR_p$；若 $C_1 \in QR_p$，则 k 为偶数，由 C_2 是否属于 QR_p 可以确定 m_0 或 m_1；如果 $C_1 \in QNR_p$，则k 为奇数，再若 $y_A \in QNR_p$，则由 C_2 是否 $\in QR_p$ 可以确定 m_0 或 m_1。
- **CCA （选择密文）攻击**：攻击者提供 $C' = (C_1, rC_2)$，请求解密，则 $M = \frac{M'}{r}$。这种性质称为密文可扩展性：可以根据一个明文密文对得到另一个具有相同明文的明文密文对，其密文由一个函数定义。
- ElGamal 算法具有乘法同态性，也可以经过修改使其具有加法同态性（M 修改为 g^M 并将 M 设得很小，并用穷搜索解决解密时出现的离散对数问题）；
  - 但不可能同时具有加法同态和乘法同态特性——半同态公钥加密算法；包括 RSA 与 ElGamal。
  - 既能够实现加法同态和乘法同态可行——全同态公钥加密算法。
- 实际中使用的 ElGamal 类算法可以抵御上述攻击。

椭圆曲线密码学：大部分的公钥密码体制参数长度很长，给密钥存储及加解密运算带来巨大负担，一种替代方法是使用椭圆曲线。椭圆曲线可以用较短的参数长度实现同样的安全级别。由于椭圆曲线密码学研究时间相对短，因此对其密码分析尚不足够；属于基于离散对数问题的公钥加密体制。

- 动因：ElGamal 算法基于循环群，需要一个与 $GF(q)$ 差异较大的循环群，使得已知关于 $GF(q)$ 的对数算法不适用。
- **椭圆曲线**：代数几何曲线（椭圆曲线的形状并不是椭圆的，指示方程类似于计算椭圆周长的方程而得名），满足：
  - 形如 $E(x,y): y^2 = x^3 + ax + b$ 且 $4a^4 + 27b^2 \neq 0$（确保方程三个解不同，非奇异椭圆曲线）；
  - 外加无穷远点 O。

  可以证明 $E(x,y)$ 结合所定义的加法构成一个“加群”。
- **困难问题**：给定点 P，从倍点 kP 求 k。
- **实数上的椭圆曲线上的运算（加法）**：E 为非奇异椭圆曲线，在 E 上定义二元运算，使其成为一个 Abel 群，这个二元运算通常用加法表示。
  - 单位元是无穷远点 O，满足 $O + P = P + O = P$；
  - 对于任意 $P, Q \in E$，设 $P = (x_1, y_1), Q = (x_2, y_2)$，$PQ$ 两点连线交 E 于 R'，则 $P + Q$ 定义为 $R = -R'$。

  性质

  - 在 E 上封闭，可交换；
  - 存在单位元 O；
  - E 上每个点都存在加法逆元；
  - 加法满足结合律。

  计算：一般来说有 $P + Q = R = (x_3, y_3)$，其中 $x_3 = -x_1 - x_2 + \lambda^2, y_3 = -y_1 + \lambda(x_1 - x_3)$，其中 $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$ $(P \neq Q)$ 且 $\lambda = \frac{3x_1^2 + \alpha}{2y_1}$ $(P = Q)$。
- **无穷远点**：平行直线的交点。
  - 直线 L 上无穷远点只有一个；
  - 平面上一组相互平行的直线有公共的无穷远点；
  - 平面上任何香蕉的两直线有不同的无穷远点；
  - 平面上全体无穷远点构成一条无穷远直线；
  - 平面上全体无穷远点与全体平常点构成射影平面。
- **模素数的椭圆曲线** (ECC)：与实数椭圆曲线类似定义，只不过在模素数意义下进行。

ECC 上相应方法的变化：原来模乘法变为 ECC 加法，模除法变为 ECC 减法；原来模指数运算相当于 ECC 累加（“乘法”）；原来模加法运算变为 ECC 指数傻姑娘加法。原来模ECC 上困难问题等价于离散对数问题，称为椭圆曲线离散对数问题。

ECC 上 Diffie-Hellman 密钥交换协议

- **步骤**：
  - 用户选定合适的椭圆曲线 $E_p(a,b)$ 并选择基点 $G = (x_1, y_1)$，该点的阶为含有大素数因子的值 n 满足 $nG = O$；
  - A 与 B 格子选定自己的私钥 $n_A < n, n_B < n$，并计算自己的公钥 $P_A = n_A G, p_B = n_B G$；
  - 计算共享密钥：$K = n_A P_B = n_B P_A = n_A n_B G$。

ECC 上 ElGamal 公钥加密

- **建立**：选择基点 $G \in E_p(a,b)$，且 G 的阶足够大并含有大素数因子，设 $nG = O$；
- **私钥**：x；
- **公钥**：$Y = xG$

- **明文**：$P(m)$ 由 m 编码得到；
- **加密**：$C = (C_1, C_2) = (rG, rY + P(m))$，其中 r 为随机数；
- **解密**：$P(m) = C_2 - xC_1 = (P(m) + rY) - xrG$。

椭圆曲线密码学结论：

- 同等安全条件下，椭圆曲线密码学的密钥长度短。
- 可以以较短的安全参数获得高安全特性，常被用在一些运算能力很低的环境中，如 Smart Cards 和 Sensor Network 等。

其他公钥加密体制 (PKC)：概率 PKC、基于二次背包的 PKC、基于有限状态自动机的 PKC、基于超奇异椭圆曲线的 PKC 等。