RSA 加密算法

RSA

刘卓

1 现代密码学

现代密码学的假设:

1. 对手知道正在使用的系统;
2. 对手可以访问任意数量的相应的明文-密文对;
3. 对手可以访问加密转换 $E_k(M) = C$ 中使用的密钥;
4. 安全性是对手不能构造解密变换 $D_k(C) = M$。

定义 1: 如果反映射 D_k 的构造在理论上非常复杂，以至于我们现在的计算工具无法实现，那么映射 E_k 就是一个 *trapdoor* 函数。

例 1

设两个因子 p,q，其中

$$\begin{aligned} p &= 163473364580925384844313388386509085984178367003309231218111085238933310010450815121211816751157 9 \\ q &= 190087128166482211312685157393541397547189678996851549366663853908802710380210449895719126146557 1 \\ p \cdot q &= 310741824049004372135075003588856793003734602284272754572016194882320644051808150455634682967172328678243791627283803341547107310850191954852900733772482278352574238645401469173660247765234660 9 \end{aligned}$$

尝试因式分解。 □

2 数论

定义 2: *Euler* φ *(n)* 函数计算在封区间 *[1,n]* 内与 n 没有公因数的整数的个数。也就是说,

$$\varphi(n) = \#\{m \in \mathbb{Z} : 1 \leq m \leq n, gcd(m,n) = 1\}$$

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\varphi(n)$	1	1	2	2	4	2	6	4	6	4	10	4	12	6	8

引理 *1*：如果 *m,n* 为正整数，使得 *gcd(m,n)= 1,* 则：

$$\varphi(mn) = \varphi(m)\varphi(n)$$

例 2

$$\varphi 55) = \varphi(5 \times 11) = \varphi(5)\varphi(11) = 4 \times 10$$

□

定义 *3*：莫比斯公式：

$$\mu(d) = \begin{cases} (-1)^k \text{如果 } d \text{ 是 } k \text{ 个不同质数的乘积} \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad \text{其他} \end{cases}$$

d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\mu(d)$	1	−1	−1	0	−1	1	−1	0	0	1	−1	0	−1	1	1

$4 = 2 \cdot 2$，有相同的质数所得，所以是 0，$12 = 2 \cdot 2 \cdot 3$ 也有相同质数所得，所以是 0。$15 = 3 \cdot 5$ 由不同质数所得，所以是 $(-1)^2 = 1$

引理 *2*：假设 m 和 n 没有公因数。进一步假设 m 是 k 个不同素数的乘积 n 是 r 个不同素数的乘积。那么 mn 就是 $k + r$ 不同质数的乘积，即：

$$\mu(mn) = (-1)^{k+r} = (-1)^k(-1)^r = \mu(m)\mu(n)$$

定理 *1*：如果 n 是一个正整数，则可以被质因数分解：

$$n = p_1^{a_1} p_2^{a_2} \cdots p_r^{a_r}$$

进一步表示为：

$$\varphi(n) = n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\cdots\left(1 - \frac{1}{p_r}\right)$$

例 3

计算 $\varphi(360)$:

$$\begin{aligned} 360 &= 2 \cdot 180 \\ &= 2 \cdot 2 \cdot 90 \\ &= 2 \cdot 2 \cdot 2 \cdot 45 \\ &= 2 \cdot 2 \cdot 2 \cdot 3 \cdot 3 \cdot 5 \\ &= 2^3 \cdot 3^2 \cdot 5 \end{aligned}$$

$$\begin{aligned}\varphi(360) &= n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\cdots\left(1-\frac{1}{p_r}\right)\\ &= 360\left(1-\frac{1}{2}\right)\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right)\\ &= 96\end{aligned}$$

```
# 计算欧拉公式值
def gcd(p,q):
    while q != 0:
        p, q = q, p%q
    return p

def is_coprime(x, y):
    return gcd(x, y) == 1

def phi(x):
    if x == 1:
        return 1
    else:
        n = [y for y in range(1,x) if is_coprime(x,y)]
        return len(n)
print(phi(360))
```

□

那么 $\varphi(n)$ 和 $\mu(d)$ 有什么关系呢?

$$\varphi(n)=\sum_{d|n}\mu(d)\frac{n}{d}$$

例 4

让 $n=p_1^{a_1}p_2^{a_2}$

则 $p_1^i p_2^j$, $0\le i\le a_1$,$0\le j\le a_2$

- $d=p_1^0p_2^0=1\Rightarrow\mu(1)=(-1)^0=1$
- $d=p_1^1p_2^0=p_1\Rightarrow\mu(p_1)=(-1)^1=-1$
- $d=p_1^0p_2^1=p_2\Rightarrow\mu(p_2)=(-1)^1=-1$
- $d=p_1^0p_2^1=p_1p_2\Rightarrow\mu(p_1p_2)=(-1)^2=1$

$$\varphi(n) = \mu(1)\cdot\frac{n}{1}+\mu(p_1)\cdot\frac{n}{p_1}+\mu(p_2)\cdot\frac{n}{p_2}+\mu(p_1p_2)\cdot\frac{n}{p_1p_2}$$

$$
\begin{aligned}
&= n - \frac{n}{p_1} - \frac{n}{p_2} + \frac{n}{p_1 p_2} \\
&= n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)
\end{aligned}
$$

□

模运算的两个基本结果:

1. **定理 *2(费马 Fermat)***: 如果 p 是一个素数 a 是一个不能被 p 整除的整数，那么

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

2. **定理 *3(欧拉 Euler)***: 如果 a 和 n 是相对素数的非零整数，那么

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

例 5

$$115 = 5 \cdot 23 \Rightarrow \varphi(115) = \varphi(5)\varphi(23) = 4 \cdot 22 = 88$$

$$\underbrace{12659}_{a}^{\overbrace{88}^{\varphi(n)}} \equiv 1 \pmod{\underbrace{115}_{n}}$$

□

3 RSA

RSA 加密算法是一种非对称加密算法，在公开密钥加密和电子商业中被广泛使用。RSA 是由罗纳德 · 李维斯特（Ron Rivest）、阿迪 · 萨莫尔（Adi Shamir）和伦纳德 · 阿德曼（Leonard Adleman）在 1977 年一起提出的。当时他们三人都在麻省理工学院工作。RSA 就是他们三人姓氏开头字母拼在一起组成的。

1973 年，在英国政府通讯总部工作的数学家克利福德 · 柯克斯（Clifford Cocks）在一个内部文件中提出了一个与之等效的算法，但该算法被列入机密，直到 1997 年才得到公开。

对极大整数做因数分解的难度决定了 RSA 算法的可靠性。换言之，对一极大整数做因数分解愈困难，RSA 算法愈可靠。假如有人找到一种快速因数分解的算法的话，那么用 RSA 加密的信息的可靠性就会极度下降。但找到这样的算法的可能性是非常小的。今天只有短的 RSA 钥匙才可能被强力方式破解。到目前为止，世界上还没有任何可靠的攻击 RSA 算法的方式。只要其钥匙的长度足够长，用 RSA 加密的信息实际上是不能被破解的。