密码学导论

Introduction to Cryptography

密码导学团队

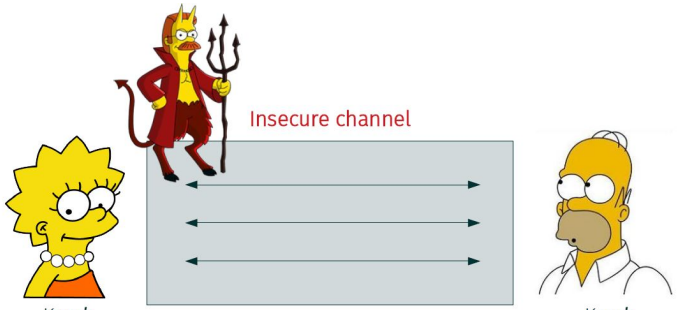
北京理工大学网络空间安全学院

17 octobre 2023

第六章　RSA密码体制

公钥密码学简介

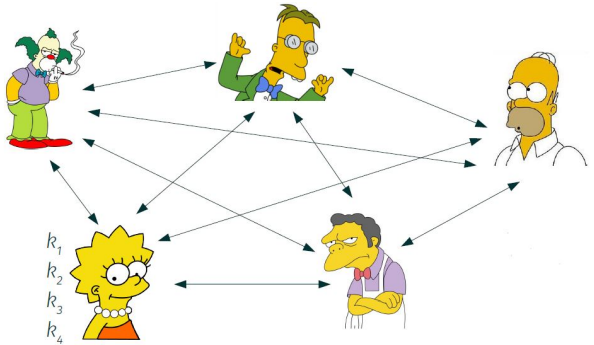
目前为止我们研究的密码学模型中，加密采用的密钥与解密采用的密钥是相同的。我们称这类密码体制为对称密钥密码体制。



对称密码体制的缺点是加密方和解密方必须在传输密文前使用一个安全信道交换密钥，这在实际中很难实现。（两人见面交换密钥？）

公钥密码学简介

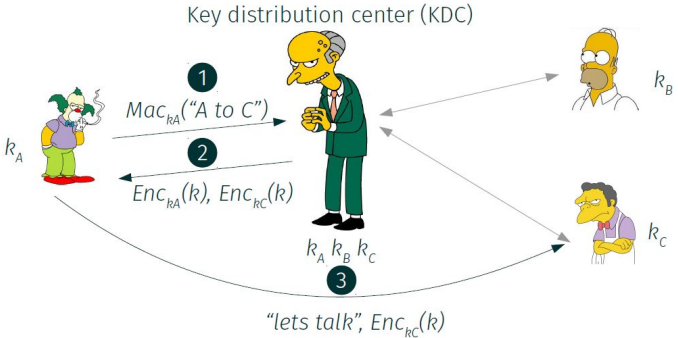
目前为止我们研究的密码学模型中，加密采用的密钥与解密采用的密钥是相同的。我们称这类密码体制为对称密钥密码体制。



同时，在共有N个用户的系统中，每个用户维护$N-1$个密钥，整个系统中有N^2量级的密钥。给密钥管理带来麻烦。

公钥密码学简介

目前为止我们研究的密码学模型中，加密采用的密钥与解密采用的密钥是相同的。我们称这类密码体制为对称密钥密码体制。



引入可信的密钥分发中心可以一定程度上解决，但只适用于内部系统。

公钥密码学

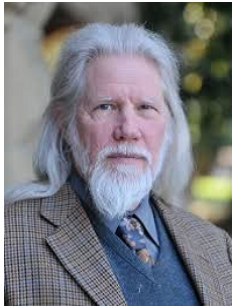
为此引入公钥密码体制，公钥密码体制中加密密钥和解密密钥不同，并且已知加密密钥无法计算得到解密密钥。
因此实体可以对外公布自己的加密密钥，其他实体就可以利用该加密密钥加密消息。由于其他实体没有解密密钥，因此无法解密密文，因此可以保证消息的私密性。

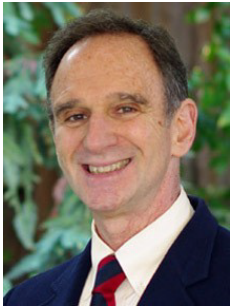
- 一个密钥是加密密钥，发送者用来加密消息，是公开的，称为公开密钥（public key）。
- 另一个密钥是解密密钥，接收者用来解密恢复明文，是秘密持有的，称为解密密钥（private key）。

公钥密码学的开端

- 1970年，James Ellis在一篇题为“非秘密加密的可能性”的论文中提出了公钥密码学的思想，但是这篇论文没有在公开文献中发表。
- 1973年，Clifford Cocks在“关于非秘密加密的注释”论文中描述了一个本质上与RSA密码体制相同的公钥密码体制，这篇论文也没有在公开文献中发表。
- 1976年，Diffie和Hellman在一篇题为“密码学的新方向”中公开提出公钥密码体制的思想，提出了密钥协商的协议。
- 1977年，Rivest,Shamir和Adleman发明了著名的RSA密码体制。
- 1985年，El Gammal 提出了一个加密方案，与Diffie-Hellman密钥协商具有类似的原理。

公钥密码学





644 IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-22, NO. 6, NOVEMBER 1976

New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

COMMUNICATIONS OF THE ACM *A Publication of the Association for Computing Machinery*
1133 AVENUE OF THE AMERICAS NEW YORK, NEW YORK 10036 212 265-6300
R. L. ASHENHURST, Editor-in-Chief
MYRTLE R. KELLINGTON, Executive Editor

Reply to:

Susan L. Graham
Computer Science Division - EECS
University of California, Berkeley
Berkeley, Ca. 94720

October 22, 1975

Mr. Ralph C. Merkle
2441 Haste St., #19
Berkeley, Ca. 94704

Dear Ralph:

Enclosed is a referee report by an experienced cryptography expert on your manuscript "Secure Communications over Insecure Channels." On the basis of this report I am unable to publish the manuscript in its present form in the Communications of the ACM.

Diffie和Hellman因为对现代密码学的基础性贡献获得2015年的图灵奖。

公钥加密方案

公钥密码体制可以抽象为一种陷门单向函数。

单向性：已知明文x和加密密钥k很容易计算出密文y，而且已知密文y和公钥k很难计算出明文x，这与单向函数的定义类似。

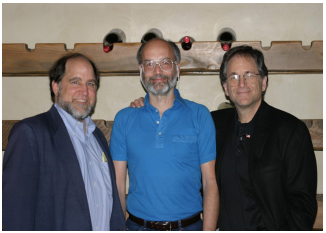
有陷门：而且已知密文y和解密密钥d，可以很容易算出明文x，解密密钥d是单向函数的陷门。

RSA加密方案

RSA加密方案

- 数论知识
  - 群的定义
  - 群的性质
  - 元素求逆
- RSA密码体制
  - 算法正确性
  - 算法安全性
  - 算法高效性

Shamir, Rivest, Adleman



Shamir, Rivest, Adleman in 2003
三人获2002年图灵奖。

代数结构

定义

一个非空集合S连同若干个定义在S上的运算$f_1, f_2, ..., f_m$组成的系统称为代数系统或代数结构，记作$< S, f_1, f_2, ..., f_m >$。

由定义可知，一个代数系统需满足以下3个条件：

1. 有一个非空集合S。
2. 有建立在S上的一些运算。
3. 这些运算在S上是封闭的。

代数结构

定义

一个非空集合S连同若干个定义在S上的运算$f_1, f_2, ..., f_m$组成的系统称为代数系统或代数结构，记作$< S, f_1, f_2, ..., f_m >$。

由定义可知，一个代数系统需满足以下3个条件：

1. 有一个非空集合S。
2. 有建立在S上的一些运算。
3. 这些运算在S上是封闭的。

例：模m的剩余类

在$\mathbb{Z}_m$上定义运算$+_m$和$\times_m$，对任意的$[a], [b] \in \mathbb{Z}_m$，

$$[a] +_m [b] = (a + b) \mod m = [a + b] \in \mathbb{Z}_m,$$
$$[a] \times_m [b] = (a \times b) \mod m = [a \times b] \in \mathbb{Z}_m,$$

则$< \mathbb{Z}_m, +_m, \times_m >$是一个代数结构。

代数结构的更多性质

代数系统的更多性质

交换律

设$< S, * >$是一个代数系统，若对任意的$x, y \in S$有$x * y = y * x$，则称二元运算$*$是可交换的，或说$*$满足交换律。

结合律

设$< S, * >$是一个代数系统，若对任意的$x, y, z \in S$有$(x * y) * z = x * (y * z)$，则称二元运算$*$是可结合的，或说$*$满足结合律。

分配律

设代数系统$< S, *, \odot >$，对任意$x, y, z \in S$，若$x * (y \odot z) = (x * y) \odot (x * z)$，则称$*$对$\odot$满足左分配律；若$(y \odot z) * x = (y * x) \odot (z * x)$，则称$*$对$\odot$满足右分配律；若两者都满足，则称$*$对$\odot$满足分配律。

代数结构的更多性质

单位元

设代数系统$< S, * >$，且存在$e_l, e_r, e \in S$，对任意$x \in S$，若$e_l * x = x$，则称e_l是S中关于$*$的一个左单位元；若$x * e_r = x$，则称e_r是S中关于$*$的一个右单位元。若e关于$*$既是左单位元又是右单位元，则称e为S中关于$*$的单位元。

例：在$< Z_m, +_m, \times_m >$中，运算$+_m$的单位元是$[0]$，运算$\times_m$的单位元是$[1]$。

逆元

设$< S, * >$是一个代数系统，e是S中关于$*$的单位元。对于$x \in S$，若存在$y_l \in S$使得$y_l * x = e$，则称y_l是x的左逆元；对于$x \in S$，若存在$y_r \in S$使得$x * y_r = e$，则称y_r是x的右逆元。对于$x \in S$，若存在$y \in S$既是x的左逆元又是x的右逆元，则称y为x的逆元，通常记为x^{-1}。

代数结构的更多性质

零元

设代数系统$< S, * >$，且存在$\theta_l, \theta_r, \theta \in S$，对任意的$x \in S$，若$\theta_l * x = \theta_l$，则称$\theta_l$是$S$中关于$*$的一个左零元；若$x * \theta_r = \theta_l$，则称$\theta_r$是$S$中关于$*$的一个右零元。若$\theta$关于$*$既是左零元又是右零元，则称$\theta$为$S$中关于$*$的零元。

例：在$< Z_m, \times_m >$中，$[0]$是零元，因为对任意的$[a] \in Z_m$，有$[a] \times_m [0] = [0] \times_m [a] = [0]$。

群的定义

群

群$< G, * >$是一个代数系统，其中二元运算$*$满足以下3条：

1. 对所有的$a, b, c \in G$： $a * (b * c) = (a * b) * c$
2. 存在一个元素是e，对任意元素$a \in G$，有： $a * e = e * a = a$
3. 对每一$a \in G$，存在一个元素a^{-1}，使： $a^{-1} * a = a * a^{-1} = e$

简单地说, 群是具有一个可结合运算, 存在单位元, 每个元素存在逆元的代数系统。

有限群与可交换群

有限群的阶数

如果G是有限集合，则称$< G, * >$是有限群；如果G是无限集合，则称$< G, * >$是无限群。有限群G的元素个数$|G|$称为群的阶数。

可交换群

群中的运算$*$一般称为乘法。如果$*$是一个可交换运算，那么群$< G, * >$就称为可交换群，或称阿贝尔群。在可交换群中，若运算符$*$改用$+$，则称为加法群，此时逆元a^{-1}写成$-a$。

群运算解的唯一性

定理

如果$< G, * >$是一个群，则对于任何$a, b \in G$，

- (a) 存在一个唯一的元素x，使得$a * x = b$。
- (b) 存在一个唯一的元素y，使得$y * a = b$。

群运算解的唯一性

定理

如果$< G, * >$是一个群，则对于任何$a, b \in G$，

- (a) 存在一个唯一的元素x，使得$a * x = b$。
- (b) 存在一个唯一的元素y，使得$y * a = b$。

证：(a) 至少有一个x满足$a * x = b$，即$x = a^{-1} * b$，因为

$$a * (a^{-1} * b) = (a * a^{-1}) * b = e * b = b$$

如果x是G中满足$a * x = b$的任意元素，则

$$x = e * x = (a^{-1} * a) * x = a^{-1} * (a * x) = a^{-1} * b$$

所以，$x = a^{-1} * b$是满足$a * x = b$的唯一元素。 □

群元素的幂运算

幂

定义群$< G, * >$的任意元素a的幂：对$n \in N$，则

$$a^0 = e$$

$$a^{n+1} = a^n * a$$

$$a^{-n} = (a^{-1})^n$$

对任意$m, k \in I$，a^m, a^k都是有意义的，另外群中结合律成立，不难证明以下指数定律成立。

$$a^m * a^k = a^{m+k} \quad (m, k \in I)$$

$$(a^m)^k = a^{mk} \quad (m, k \in I)$$

群元素的阶及其性质

元素的阶

定义　设$< G, * >$是一个群, 且$a \in G$, 如果存在正整数n使$a^n = e$, 则称元素的阶是有限的, 最小的正整数n称为元素a的阶。 如果不存在这样的正整数n, 则称元素a具有无限阶。

显然, 群的单位元e的阶是1。

群元素的阶及其性质

元素的阶

定义　设$< G, * >$是一个群, 且$a \in G$, 如果存在正整数n使$a^n = e$, 则称元素的阶是有限的, 最小的正整数n称为元素a的阶。 如果不存在这样的正整数n, 则称元素a具有无限阶。

显然, 群的单位元e的阶是1。

定理

群中的任一元素和它的逆元具有同样的阶。

证　设$a \in G$具有有限阶n，即$a^n = e$，因此：

$$(a^{-1})^n = a^{-1*n} = (a^n)^{-1} = e^{-1} = e$$

如果(a^{-1})的阶是m，则$m \leq n$。另一方面

$$a^m = [(a^{-1})^m]^{-1} = e^{-1} = e$$

因而$n \leq m$，故$m = n$。

群元素的阶及其性质

定理

在有限群$< G, * >$中，每一个元素具有一有限阶，且阶数至多是$|G|$。

证　设a是$< G, * >$中任一元素。在序列$a, a^2, a^3, ..., a^{|G|+1}$中至少有两元素是相等的。不妨设$a^r = a^s$，这里$1 \leq s < r \leq |G| + 1$。因为

$$e = a^0 = a^{s-s} = a^s * a^{-s} = a^r * a^{-s} = a^{r-s}$$

所以，a的阶数至多是$r - s \leq |G|$，证毕。

群元素的阶及其性质

定理

在有限群$< G, * >$中，每一个元素具有一有限阶，且阶数至多是$|G|$。

证　设a是$< G, * >$中任一元素。在序列$a, a^2, a^3, ..., a^{|G|+1}$中至少有两元素是相等的。不妨设$a^r = a^s$，这里$1 \leq s < r \leq |G| + 1$。因为

$$e = a^0 = a^{s-s} = a^s * a^{-s} = a^r * a^{-s} = a^{r-s}$$

所以，a的阶数至多是$r - s \leq |G|$，证毕。

循环群与生成元

设$< G, * >$是一个群，I是整数集合。如果存在一个元素$g \in G$，对于每一个元素$a \in G$都有一个相应的$i \in I$，能把a表示成g^i形式，则称$< G, * >$是一个循环群。或说循环群是由g生成的，g是$< G, * >$的**生成元**。

群的生成元及其性质

定理

设$< G, * >$是由$g \in G$生成的有限循环群，如果$|G| = n$，则$g^n = e$，

$$G = \{g, g^2, g^3, ..., g^n = e\}$$

且n是使$g^n = e$的最小正整数，即生成元的阶数与群的阶数相等。

群的生成元及其性质

定理

设$< G, * >$是由$g \in G$生成的有限循环群，如果$|G| = n$，则$g^n = e$，

$$G = \{g, g^2, g^3, ..., g^n = e\}$$

且n是使$g^n = e$的最小正整数，即生成元的阶数与群的阶数相等。

证：(1) 假定有正整数$m < n$使得$g^m = e$，则对G中任一元素g^k，
设$k = mq + r, \quad 0 \leq r < m$，于是

$$g^k = g^{mq+r} = (g^m)^q * g^r = g^r$$

这意味着G中每一个元素都可以写成g^r形式，但$r < m$，所以G中最多有m个不同元素，这与$|G| = n$矛盾，所以$g^m = e$而$m < n$是不可能的。

群的生成元及其性质

定理

设$< G, * >$是由$g \in G$生成的有限循环群，如果$|G| = n$，则$g^n = e$，

$$G = \{g, g^2, g^3, ..., g^n = e\}$$

且n是使$g^n = e$的最小正整数，即生成元的阶数与群的阶数相等。

证：(1) 假定有正整数$m < n$使得$g^m = e$，则对G中任一元素g^k，
设$k = mq + r, \quad 0 \leq r < m$，于是

$$g^k = g^{mq+r} = (g^m)^q * g^r = g^r$$

这意味着G中每一个元素都可以写成g^r形式，但$r < m$，所以G中最多有m个不同元素，这与$|G| = n$矛盾，所以$g^m = e$而$m < n$是不可能的。

(2) $\{g, g^2, g^3, ..., g^n\}$中的元素全不相同。若不然有$g^i = g^j$，不妨设$i < j$，于是$g^{j-i} = e$。但$j - i < n$。所以这是不可能的。

由于$< G, * >$是群，其中必有单位元，
由(2)得$G = \{g, g^2, g^3, ..., g^n\}$，因此，
由(1)得$g^n = e$，证毕。

元素求逆

欧几里得算法 Euclidean Algorithm(a,b)：求整数a,b的最大公约数

算法6.1：Euclidean Algorithm(a,b)

$r_0 \leftarrow a$

$r_1 \leftarrow b$

$m \leftarrow 1$

while $r_m \neq 0$ **do**

$q_m \leftarrow \lfloor \frac{r_{m-1}}{r_m} \rfloor$

$r_{m+1} \leftarrow r_{m-1} - q_m r_m$

$m \leftarrow m+1$

endwhile

$m \leftarrow m-1$

return $(q_1, ..., q_m; r_m)$

元素求逆

欧几里得算法 Euclidean Algorithm(a,b)：求整数a,b的最大公约数

算法6.1：Euclidean Algorithm(a,b)

$r_0 \leftarrow a$

$r_1 \leftarrow b$

$m \leftarrow 1$

while $r_m \neq 0$ **do**

$q_m \leftarrow \lfloor \frac{r_{m-1}}{r_m} \rfloor$

$r_{m+1} \leftarrow r_{m-1} - q_m r_m$

$m \leftarrow m+1$

endwhile

$m \leftarrow m-1$

return $(q_1, ..., q_m; r_m)$

我们有

$$r_m = \gcd(a,b)$$

原因：在该算法中

$$\begin{aligned} &\gcd(r_0, r_1) \\ =&\gcd(r_1, r_2) = ... \\ =&\gcd(r_{m-1}, r_m) = r_m. \end{aligned}$$

元素求逆

欧几里得算法 Euclidean Algorithm(a,b)：求整数a,b的最大公约数

算法6.1：Euclidean Algorithm(a,b)

$r_0 \leftarrow a$

$r_1 \leftarrow b$

$m \leftarrow 1$

while $r_m \neq 0$ **do**

$q_m \leftarrow \lfloor \frac{r_{m-1}}{r_m} \rfloor$

$r_{m+1} \leftarrow r_{m-1} - q_m r_m$

$m \leftarrow m+1$

endwhile

$m \leftarrow m-1$

return $(q_1, ..., q_m; r_m)$

注意，每一个r_j都可以写为r_{j-1}和r_{j-2}的线性组合：

$$r_j = r_{j-2} - q_{j-1} r_{j-1},$$

那么通过迭代，最后的r_m就可以写成r_0和r_1的组合，那么如果a,b互素，就有

$$ax + by = \gcd(a,b) = 1,$$

就找到了$a^{-1} \mod b$。

所以我们改进下算法：每一步不记录商q_j的值，而是记录r_0和r_1的系数s_j和t_j：

$$r_j = s_j r_0 + t_j r_1.$$

扩展的欧几里得算法

假定按下面构造定义了两个数列$s_0, s_1, ..., s_m$和$t_0, t_1, ..., t_m$

$$s_j = \begin{cases} 1, & j = 0 \\ 0, & j = 1 \\ s_{j-2} - q_{j-1}s_{j-1}, & j \geq 2 \end{cases} \qquad t_j = \begin{cases} 0, & j = 0 \\ 1, & j = 1 \\ t_{j-2} - q_{j-1}t_{j-1}, & j \geq 2 \end{cases}$$

定理5.1

对于$0 \leq j \leq m$，有$r_j = s_j r_0 + t_j r_1$，其中r_j按欧几里得算法定义，s_j, t_j按上述定义。

扩展的欧几里得算法

利用数学归纳法进行证明：
对于$j=0$和$j=1$，命题显然成立。
假设命题对于$j=i-1$和$j=i-2$成立，其中$i \geq 2$.

由归纳假定，则有：$r_{i-2}=s_{i-2}r_0+t_{i-2}r_1$和$r_{i-1}=s_{i-1}r_0+t_{i-1}r_1$
此时：

$$\begin{aligned} r_i &= r_{i-2}-q_{i-1}r_{i-1} \\ &= s_{i-2}r_0+t_{i-2}r_1-q_{i-1}(s_{i-1}r_0+t_{i-1}r_1) \\ &= (s_{i-2}-q_{i-1}s_{i-1})r_0+(t_{i-2}-q_{i-1}t_{i-1})r_1 \\ &= s_ir_0+t_ir_1. \end{aligned}$$

得证。 □

扩展的欧几里得算法

扩展的欧几里得算法 Extended Euclidean Algorithm(a, b)

算法6.2：Extended Euclidean Algorithm(a, b)

$a_0 \leftarrow a$
$b_0 \leftarrow b$
$t_0 \leftarrow 0$
$t \leftarrow 1$
$s_0 \leftarrow 1$
$s \leftarrow 0$
$q \leftarrow \lfloor \frac{a_0}{b_0} \rfloor$
$r \leftarrow a_0 - qb_0$

while $r > 0$ **do**
  $temp \leftarrow t_0 - qt$
  $t_0 \leftarrow t$
  $t \leftarrow temp$
  $temp \leftarrow s_0 - qs$
  $s_0 \leftarrow s$
  $s \leftarrow temp$
  $a_0 \leftarrow b_0$
  $b_0 \leftarrow r$
  $q \leftarrow \lfloor \frac{a_0}{b_0} \rfloor$
  $r \leftarrow a_0 - qb_0$
endwhile

$r \leftarrow b_0$
return (r, s, t)

有

$$r = \gcd(a, b),$$
$$sa + tb = r.$$

利用欧几里得算法进行模逆运算

推论5.2

假定$\gcd(r_0, r_1) = 1$，那么$r_1^{-1} \mod r_0 = t_m \mod r_0$.

证明：由定理5.1，有

$$1 = \gcd(r_0, r_1) = s_m r_0 + t_m r_1$$

两边模r_0约化等式，得

$$t_m r_1 \equiv 1 (\mathrm{mod}\, r_0).$$

□

例：计算$28^{-1} \bmod 75$

模逆运算

例：计算$28^{-1} \bmod 75$

i	r_j	q_j	s_j	t_j
0	75		1	0
1	28	2	0	1
2	19	1	1	-2
3	9	2	-1	3
4	1		3	-8

因此，我们发现 $3 * 75 - 8 * 28 = 1$

应用推论5.2，可得到

$28^{-1} \bmod 75 = -8 \bmod 75 = 67$

模n的乘法群与欧拉函数

模n的既约剩余类（模n的乘法群）：
$\mathbb{Z}_n^* = \{[1], [p_1], ..., [p_{t-1}]\}$，其中$\gcd(p_i, n) = 1, 1 \le i \le t-1$，$t$为$Z_n^*$的阶，记为$\phi(n)$。

欧拉函数$\phi(n)$

$\phi(n)$表示小于n且与n互素的正整数的个数。

例如：下表列出了30以内的整数的$\phi(n)$值，$\phi(1)$被规定为1.

Table – 某些数和它们的欧拉函数

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\phi(n)$	1	1	2	2	4	2	6	4	6	4	10	4	12	6	8
n	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
$\phi(n)$	8	16	6	18	8	12	10	22	8	20	12	18	12	28	8

显然，对于任意一个素数p，有

$$\phi(p) = p - 1.$$

欧拉函数

欧拉函数$\phi(n)$

$\phi(n)$表示小于n且与n互素的正整数的个数。

现在假定有两个不同的素数p和q，则对于$n = pq$，有

$$\phi(n) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1).$$

为了完全证明这一命题，考虑$\mathbb{Z}_n$集合为$\{0, 1, 2, \cdots, (pq-1)\}$，而不与$n$互素的元素包括集合$\{p, 2p, \cdots, (q-1)p\}$、集合$\{q, 2q, \cdots, (p-1)q\}$和0。因此

$$\begin{aligned}\phi(n) &= pq - [(q-1) + (p-1) + 1] \\ &= pq - (p+q) + 1 \\ &= (p-1) \times (q-1) \\ &= \phi(p) \times \phi(q).\end{aligned}$$

$\mathbb{Z}_p^*$群

若p为素数，则$\mathbb{Z}_p^* = \{1, 2, ..., p-1\}$，是一个有限循环群。例如

$$\mathbb{Z}_{13}^* = \{[1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12]\}.$$

可以验证2是一个模13的生成元（4不是生成元）：

$$\begin{array}{lll} 2^1 \bmod 13 = 2 & 2^5 \bmod 13 = 6 & 2^9 \bmod 13 = 5 \\ 2^2 \bmod 13 = 4 & 2^6 \bmod 13 = 12 & 2^{10} \bmod 13 = 10 \\ 2^3 \bmod 13 = 8 & 2^7 \bmod 13 = 11 & 2^{11} \bmod 13 = 7 \\ 2^4 \bmod 13 = 3 & 2^8 \bmod 13 = 9 & 2^{12} \bmod 13 = 1 \end{array}$$

12的因式分解为$12 = 2^2 * 3$，我们可以仅通过验证$2^6 \neq 1 (\mathrm{mod}\, 13)$以及$2^4 \neq 1 (\mathrm{mod}\, 13)$来验证2是一个模13的生成元。为什么验证这两者就可以?

拉格朗日定理

定理(Lagrange)

假定G是一个阶为$|G|$的乘法群，且$g \in G$，那么g的阶整除$|G|$。

推论1

如果$b \in \mathbb{Z}_n^*$，那么$b^{\phi(n)} \equiv 1 (\mathrm{mod}\, n)$

推论1重要，记住！

推论2(Fermat)

假定p是一个素数，且$b \in \mathbb{Z}_p$，那么$b^p \equiv b (\mathrm{mod}\, p)$

$\mathbb{Z}_p^*$的生成元判定

定理

如果$p>2$是一个素数，且$\alpha \in \mathbb{Z}_p^*$，那么$\alpha$是一个模$p$的生成元，当且仅当$\alpha^{(p-1)/q} \not\equiv 1(\mathrm{mod} p)$对于所有满足$q|(p-1)$的素数$q$都成立。

证明：如果α是一个模p既约剩余类的生成元，那么对于所有的$1 \leq i \leq p-2$，都有α^i恒不等于$1(\mathrm{mod}\, p)$，所以结果成立。

反过来，假定$\alpha \in Z_p^*$不是模p的生成元。则令d为α的阶，那么根据Lagrange定理，有$d|(p-1)$。因为α不是生成元，所以$d<p-1$。那么$(p-1)/d$是一个大于1的整数。

令q为$(p-1)/d$的素因子，那么d是$(p-1)/q$的一个因子。由于$\alpha^d \equiv 1(\mathrm{mod}\, p)$且$d|(p-1)/q$，于是有$\alpha^{(p-1)/q} \equiv 1(\mathrm{mod}\, p)$，矛盾。 □

RSA密码体制

RSA密码体制描述如下：

RSA密码体制

设$n = p * q$，其中p和q为素数。设$\mathcal{M} = \mathcal{C} = \mathbb{Z}_n$，密钥$K = \{(n, p, q, e, d) : ed \equiv 1 (\mathrm{mod} \phi(n))\}$，定义

- 加密：$\mathsf{Enc}_K(x) = x^e \mod n$，
- 解密：$\mathsf{Dec}_K(y) = y^d \mod n$

公钥：n, e；私钥：p, q, d.

RSA密码体制

RSA密码体制描述如下：

RSA密码体制

设$n = p * q$，其中p和q为素数。设$\mathcal{M} = \mathcal{C} = \mathbb{Z}_n$，密钥$K = \{(n, p, q, e, d) : ed \equiv 1 (\mathrm{mod} \phi(n))\}$，定义

- 加密：$\mathsf{Enc}_K(x) = x^e \mod n$，
- 解密：$\mathsf{Dec}_K(y) = y^d \mod n$

公钥：n, e；私钥：p, q, d.

问题：

- RSA是否正确?
- RSA是否安全?
- RSA是否能够在多项式时间加密和解密?

RSA密码体制的正确性证明

先证正确性：

由于$ed \equiv 1(\mathrm{mod}\,\phi(n))$，所以有$ed = t\phi(n)+1$，$t$为某个正整数。$\phi(n) = \phi(p)\phi(q) = (p-1)(q-1)$。

- 若$x \in \mathbb{Z}_n^*$，则

$$(x^e)^d \equiv x^{t\phi(n)+1}(\mathrm{mod}\,n) \equiv (x^{\phi(n)})^t x(\mathrm{mod}\,n) \overset{(推论1)}{\equiv} 1^t x(\mathrm{mod}\,n) \equiv x(\mathrm{mod}\,n)$$

- 若$x \in \mathbb{Z}_n \setminus \mathbb{Z}_n^*$，则$x$是$p$的倍数或$q$的倍数，不妨设$x$是$p$的倍数，$x = kp$，则$x$与$q$互素（否则与$x < pq$矛盾）

$$\begin{aligned} x^{\phi(q)} &\equiv 1 \pmod{q} (推论1) \\ x^{t\phi(p)\phi(q)} &\equiv 1 \pmod{q} \\ x^{t\phi(n)} &\equiv 1 \pmod{q} \\ \exists\, l, 使得 x^{t\phi(n)} &= lq+1 \end{aligned}$$

两边同乘$x = kp$有$x^{t\phi(n)+1} = lkpq + kp = lkn + x$，故$x^{ed} \equiv x \pmod{n}$。证毕。

RSA密码体制例子

下面描述一个RSA密码体制的小例子

假定Bob选取$p = 101, q = 113$，那么$n = 11413$，

$\phi(n) = 100 * 112 = 11200$

由于$11200 = 2^5 * 5^2 * 7$，所以可以选择一个整数e，当且仅当e不能被2，5或7整除。假定Bob选取$e = 3533$。那么

$$e^{-1} \mod 11200 = 6597.$$

Bob在一个目录中发布$n = 11413, e = 3533$

RSA密码体制例子

下面描述一个RSA密码体制的小例子

假定Bob选取$p = 101, q = 113$，那么$n = 11413$，

$\phi(n) = 100 * 112 = 11200$

由于$11200 = 2^5 * 5^2 * 7$，所以可以选择一个整数e，当且仅当e不能被2，5或7整除。假定Bob选取$e = 3533$。那么

$$e^{-1} \mod 11200 = 6597.$$

Bob在一个目录中发布$n = 11413, e = 3533$

假定Alice想加密明文9726并发送给Bob。她将计算

$$9726^{3533} \bmod 11413 = 5761$$

然后把密文5761通过信道发出。Bob在收到密文5761后，计算

$$5761^{6597} \bmod 11413 = 9726$$

RSA密码体制的安全性是基于相信加密函数$e_K(x) = x^e \mod n$是一个单向函数这一事实，所以，对于一个敌手来说试图解密密文将是计算上不可行的。

RSA密码体制的参数生成算法

RSA参数生成算法

1. 生成两个大素数，p和q，$p \neq q$
2. $n \leftarrow pq$，且$\phi(n) \leftarrow (p-1)(p-1)$
3. 选择一个随机数$e(1 < e < \phi(n))$，使得$\gcd(e, \phi(n)) = 1$
4. $d \leftarrow e^{-1} \bmod \phi(n)$
5. 公钥为(n, e)；私钥为(p, q, d)

RSA密码体制的安全性

对RSA密码体制的一个明显攻击就是密码分析者试图分解n

如果敌手可以做到这点，那么就可以很简单的计算出

$$\phi(n) = (p-1)(q-1)$$

然后敌手就可以和Bob一样地利用公钥e计算出解密密钥

$$d = e^{-1} \bmod \phi(n)$$

如果RSA密码体制要成为安全的，那么要求$n = pq$必须足够大，使得分解它是计算上不可行的。

RSA密码体制的效率

RSA加密的效率问题。

假定x和y分别是k位和l位二进制表示的正整数；
假定$k \geq l$，容易看到对x和y执行各种运算所需的时间的上界估计：

- 计算$x+y$的时间复杂度为$O(k)$
- 计算$x-y$的时间复杂度为$O(k)$
- 计算$x \cdot y$的时间复杂度为$O(kl)$
- 计算$\gcd(x,y)$的时间复杂度为$O(k^3)$
  - Euclidean算法的迭代次数为$O(k)$
    - 令$a > b$，则$r_i \geq 2r_{i+2}, 0 \leq i \leq m-2.$
  - 每次迭代，执行一次除法需要时间$O(k^2)$

RSA密码体制的效率

计算形如$x^c \mod n$的函数,n为k比特：

在RSA密码体制中，加密和解密显然都是这类模指数运算。

计算$x^c \mod n$可以通过$c-1$次模乘来实现，但是相对于k这是指数阶大的。

下面介绍“平方—乘”算法，该算法在计算上述模指数运算时可以运行在k的多项式时间。

“平方-乘”算法

例：令$n = 11413$，公开加密指数$b = 3533$。Alice利用平方—乘算法，通过计算$9726^{3533} \bmod 11413$来加密明文9726，过程如下：

i	b_i	z
11	1	$1^2 \times 9726 = 9726$
10	1	$9726^2 \times 9726 = 2659$
9	0	$2659^2 = 5634$
8	1	$5634^2 \times 9726 = 9167$
7	1	$9167^2 \times 9726 = 4958$
6	1	$4958^2 \times 9726 = 7783$
5	0	$7783^2 = 6298$
4	0	$6298^2 = 4629$
3	1	$4629^2 \times 9726 = 10185$
2	1	$10185^2 \times 9726 = 105$
1	0	$105^2 = 11025$
0	1	$11025^2 \times 9726 = 5761$

“平方-乘”算法

例：令$n = 11413$，公开加密指数$b = 3533$。Alice利用平方—乘算法，通过计算$9726^{3533} \bmod 11413$来加密明文9726，过程如下：

i	b_i	z
11	1	$1^2 \times 9726 = 9726$
10	1	$9726^2 \times 9726 = 2659$
9	0	$2659^2 = 5634$
8	1	$5634^2 \times 9726 = 9167$
7	1	$9167^2 \times 9726 = 4958$
6	1	$4958^2 \times 9726 = 7783$
5	0	$7783^2 = 6298$
4	0	$6298^2 = 4629$
3	1	$4629^2 \times 9726 = 10185$
2	1	$10185^2 \times 9726 = 105$
1	0	$105^2 = 11025$
0	1	$11025^2 \times 9726 = 5761$

因此，密文是5761

“平方-乘”算法

计算形如$x^c \mod n$的函数： 首先将指数c用二进制表示，即$c = \sum_{i=0}^{l-1} c_i 2^i, \quad c_i \in \{0,1\}$

Square-and-Multiply(x, c, n)

$z \leftarrow 1$
for $i \leftarrow l-1$ **down to** 0 **do**
  $z \leftarrow z^2 \mod n$
  if $c_i = 1$ **do**
    $z \leftarrow (z \times x) \mod n$
  endif
endfor
return z

“平方-乘”算法

计算形如$x^c \mod n$的函数： 首先将指数c用二进制表示，即$c = \sum_{i=0}^{l-1} c_i 2^i, \quad c_i \in \{0,1\}$

Square-and-Multiply(x, c, n)

$z \leftarrow 1$
for $i \leftarrow l-1$ **down to** 0 **do**
  $z \leftarrow z^2 \mod n$
  if $c_i = 1$ **do**
    $z \leftarrow (z \times x) \mod n$
  endif
endfor
return z

在该算法中，模乘的次数等于c的二进制中1的次数，因此模乘的执行次数至少为l，最多为$2l$

RSA密码体制小结

到目前为止，我们已经讨论了RSA的加密和解密运算。

关于RSA的参数生成算法，

第一步，构造素数p和q的方法将在下一节讨论；

第二步，计算n和$\phi(n)$是直接的，可以在时间$O(k^2)$内完成；

第三步，产生加密密钥e和第四步产生解密密钥d，利用欧几里得算法时间复杂度为$O(k^3)$

Thank you for your attention !

Questions ?