古典密码&RSA基础

ISLAB_2019WINTER

INTRO

古典密码学(通识)

- 单表替换加密
- 多表替换加密
- others

现代密码学

- 对称加密
- 非对称加密

basic skills
各种进制转换、编码方式及相互转换

古典密码学

单表代换加密

明、密文一一对应。一种映射关系。

较短的密码可以观察之后直接爆破，较长的密码可以进行频率分析。

e.g.凯撒密码：把字母表看成循环队列，向前或后移动k位。在凯撒密码基础上还有移位密码，以ASCII码表作为循环队列进行移位操作。

多表代换加密

加密后字母不再保持原来的频率。

e.g.维吉尼亚密码：把26种凯撒密码的加密方式排列成一个矩阵。把秘钥重复至与明文相同的长度，再对照查表即可。

维吉尼亚——举个栗子

明文

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

密钥

明文：come greatwall
密钥：crypto
密文：efkt zferrltzn

others&tip

- others

e.g.栅栏密码、培根密码、键盘密码…

tip:

pycipher库：可以直接破解一些简单的密码，如凯撒、维吉尼亚。用法：
https://pycipher.readthedocs.io/en/master/

安装：sudo pip install pycipher

更多古典密码、编码方式有兴趣可以了解一下 https://www.tuicool.com/articles/2E3INnm

现代密码

对称加密

只有一个密钥，加密和解密使用相同的密钥。协商密钥易泄露，密钥数量大难以管理。

非对称加密

有一对公钥（public key）和私钥（private key）。A生成一对密钥，把公钥向其他人公开，自己保留私钥。得到公钥的B把信息加密后发给A, A再用自己的私钥进行解密。

RSA-数论基础

- 如果两个正整数，除了1以外，没有其他公因子，我们就称这两个数是互质关系
- 欧拉函数：小于等于n的正整数中与n构成互质关系的个数。对于素数p, φ(p)=p-1，对于对两个互质的数p,q， φ（pq）=φ(p)φ(q) = (p-1)(q-1)。
- 模反元素：ab ≡ 1 (mod r), 则a,b互为模r的模反元素。
- 贝祖等式：若设a,b是整数，则存在整数x,y，使得ax+by=gcd（a,b） gcd:最大公因数
- 如果两个正整数a和n互质，那么一定可以找到整数b，使得 ab-1 被n整除，或者说ab被n除的余数是1。
- 扩展欧几里得(egcd):已知a, b求解一组x，y，使它们满足ax+by = gcd(a, b) =d

RSA-数论基础

中国剩余定理：

求解同余方程组的最小非负整数解，其中模数两两互质。

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ x \equiv a_3 \pmod{m_3} \\ \cdots \\ x \equiv a_k \pmod{m_k} \end{cases}$$

举个栗子：存在一个数x，除以3余2，除以5余三，除以7余二，求这个数。

$$\gcd(3,5,7) = M = 3 \times 5 \times 7 = 105$$

$$\begin{cases} x \equiv 2 \pmod{3} \\ x \equiv 3 \pmod{5} \\ x \equiv 2 \pmod{7} \end{cases} \quad \begin{cases} m_1 = \frac{105}{3} = 35 \\ m_2 = \frac{105}{5} = 21 \\ m_3 = \frac{105}{7} = 15 \end{cases} \quad \begin{cases} 35t_1 \equiv 1 \pmod{3} \\ 21t_2 \equiv 1 \pmod{5} \\ 15t_3 \equiv 1 \pmod{7} \end{cases} \quad \begin{cases} t_1 = 2 \\ t_2 = 1 \\ t_3 = 1 \end{cases}$$

一个解 $x = t_1 m_1 \times 2 + t_2 m_2 \times 3 + t_3 m_3 \times 2 = 233$

最小非负解 $x = 233 - 105 \times 2 = 23$

RSA-数论基础

对于一个一般的模数互质的一次同余方程，求解过程即为

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ x \equiv a_3 \pmod{m_3} \\ \dots \\ x \equiv a_k \pmod{m_k} \end{cases}$$

令$M = \prod_{i=1}^{k} m_i$，即M是所有m_i的最小公倍数

t_i为同余方程$\frac{M}{m_i}t_i \equiv 1 \pmod{m_i}$的最小非负整数解

则有一个解为$x = \sum_{i=1}^{k} a_i \frac{M}{m_i} t_i$

通解为$x + i * M (i \in Z)$

特别的，最小非负整数解为$(x\%M + M)\%M$

证明过程为：

对于m_i，$M_i = \frac{M}{m_i} = m_1 \times m_2 \cdots \times m_{i-1} \times m_{i+1} \times \cdots \times m_k$

m_i、M_i互质，故存在t_i，使$t_i M_i \equiv 1 \pmod{m_i}$，则$a_i t_i M_i \equiv a_i \pmod{m_i}$

因为M_i的因子有m_j $(i \neq j)$，故$t_i M_i \equiv 0 \pmod{m_j}$，则$a_i t_i M_i \equiv 0 \pmod{m_j}$

$x = \sum_{i=1}^{k} a_i t_i M_i$，则可满足$x \equiv a_i \pmod{m_i}$，x是一个解.

RSA正常加密、解密过程

- 随机选择两个不相等的大质数p,q
- n = p*q, r = φ(n) = (p − 1)(q − 1)
- 随机选择一个比r小的整数e，且e,r互质
- 得到d，d是e对r的模反元素，即ed ≡ 1 (mod r)
- (n, e)是公钥，(n, d)是私钥，销毁p,q
- m：明文，c:密文。当B要给A发消息，就要用公钥把给A的消息m进行加密成密文c，即 $m^e \equiv c \pmod{n}$ ，然后A就可以用自己的私钥把c解密成明文m，即 $c^d \equiv m \pmod{n}$

在有公钥的情况下，如果想知道私钥, 就需要知道p,q，但是大数因数分解是很难的，所以RSA比较安全。

正确性的证明有兴趣的可以自己找找看看

tip

gmpy2库：是对GMP的封装。

提供求模反元素的gmpy2.invert，则d = gmpy2.invert(e, r)。gmpy2.iroot(x,n)也是低指数时常用的。

更多gmpy2的用法可参考 https://gmpy2.readthedocs.io/en/latest/

当然，求模反元素也可以自己写，用扩展欧几里得

```
def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        g, y, x = egcd(b % a, a)
        return (g, x - (b // a) * y, y)
def modinv(a, m):
    g, x, y = egcd(a, m)
    if g != 1:
        raise Exception('modular inverse does not exist')
    else:
        return x % m
d=modinv(e,(p-1)*(q-1))
```

TASK

https://ctf.bugku.com/challenges 加密-简单加密

tip:base64编码：三个8bit→四个6bit(二进制前两位是零)

如果要编码的二进制数据不是3的倍数，最后会剩下1个或2个字节怎么办？在原数据后面添加1个或2个零值字节，使其字节数是3的倍数。然后，在编码后的字符串后面添加1个或2个等号“=”。