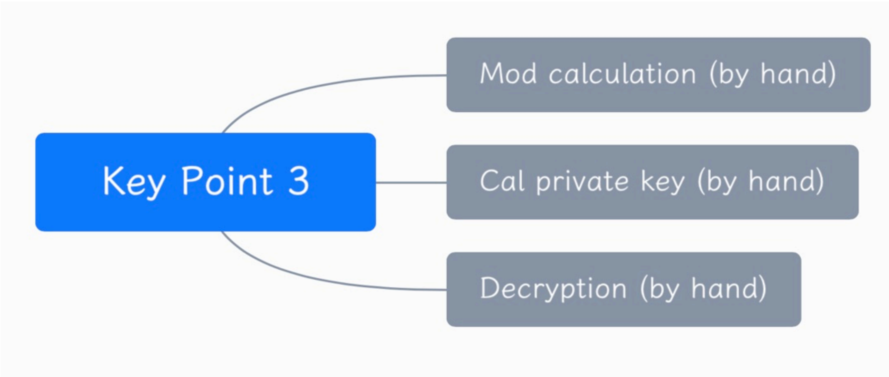
CS211FZ Note 3 | Algorithms & Data Structures | DSA2

Key Point 3 (Cryptography)

- Mod calculation (by hand)
- Cal private key (by hand)
- Decryption (by hand)



L4 Cryptography

- Symmetric Encryption对称加密
  - 加密和解密使用**同一个密钥**
- 非对称加密 RSA
  - Trapdoor
  - public-key cryptography
  - 加密和解密使用**不同的密钥**

对称加密&&非对称加密

- **对称加密**：加密和解密都是用同一个密钥的算法，称作对称加密。

  （1）甲方选择某一种加密规则，对信息进行加密；

  （2）乙方使用同一种规则，对信息进行解密。

- **非对称加密**：加密和解密需要不同的密钥。

  （1）乙方生成两把密钥（公钥和私钥）。公钥是公开的，任何人都可以获得，私钥则是保密的。

  （2）甲方获取乙方的公钥，然后用它对信息加密。

  （3）乙方得到加密后的信息，用私钥解密。

加密解密过程：d(c(x)) = x

涉及参数：x明文　c(x)密文　d私钥　c公钥

在RSA私钥和公钥生成的过程中，共出现过p,q,N,φ(N),e,d，其中(N,e组成公钥)，其他的都不是公开的，一旦d泄露，就等于私

那么能不能根据N,e推导出d呢：

ed ≡ 1(mod φ(N))。只有知道e和φ(N)，才能算出d

φ(N)=(p-1)(q-1)。只有知道p和q,才能算出φ(N)

n=pq,只有将n分解才能算出p和q

4.1 对称加密 symmetric cryptography

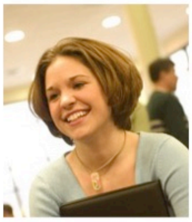
Simple Example

- Bob and Alice want to communicate but don't want anybody else to be able to read the message

- Alice tells Bob her secret key that she uses to encode the message
  - Secret key is → advance each letter one place in the alphabet

- Now when Alice sends Bob a message, she can be sure that it is secure because the secret key is needed to decode it and nobody else will have it

Cpc't b hpctijuf!



Bob is listening in!

Problem

- Bob is listening in and can obtain the encrypted message (ciphertext)

- He also knows Alice's secret key since she has made it available to him

- The major problem here is that everybody Alice communicates with knows her secret key and therefore all these people can decrypt her messages

- So much for security – Alice will lose all her friends!

4.2 非对称加密 asymmetric encryption（DSA）

A one-way system

- This sounds like magic

- How can you possibly have a system where you can encode a message easily but not decode that same message?
- Surely to decode it you just do the opposite of the thing you did to encode it (e.g., subtract one letter of the alphabet rather than add one)?
- We need a one-way system – an irreversible process

Public Key Cryptography

- Public key cryptography, also known as asymmetric cryptography is a form of cryptography where a user has a pair of cryptographic keys
- The private key is kept secret while the public key is widely distributed
- The keys are related mathematically but the private key cannot be derived from the public key
- A message encrypted by the public key must be decrypted by the corresponding private key

公钥加密技术 Public Key Cryptography：

公钥加密，也被称为非对称加密，是一种用户拥有一对加密密钥的加密形式。

1、私钥保密，而公钥广泛分发

2、密钥在数学上是相关的，但私钥不能从公钥派生

3、用公钥加密的消息必须用对应的私钥解密

前驱数学知识：

1. 互质关系

如果两个正整数，除了1之外没有其他公因子，我们称这两个数是互质关系。比如15和32，说明不是质数也可以构成互质关系。

2. 欧拉函数

如果n=1 则 $\phi(1)=1$ *应为1与任何(包括自己)都构成互质关系*

如果n是质数 则 $\phi(n)=n-1$ *应为质数与每个小于他的数都构成互质关系*

如果n是质数的某一个次方，即n=p^k(p为质数，k>=1),则

Key Generator

- Alice picks a generator number g
- She also picks a modulus p
- Now Alice picks a random private key x that lies somewhere between 0 and p
- Using these three numbers Alice computes g^x modulo p
- Alice then publishes (p, g, g^x mod p) as her public key
- Others know that the generator has been raised to some power x and then moduloed to give the result
- However, they are unable to figure out what the power is

Encryption Algorithm

- Bob has Alice's public key (p, g, g^x mod p)
- Bob converts his message m into a number between 0 and p
- He then picks a random number y again between 0 and p
- Bob now calculates g^y mod p and $m.(g^x)^y$ mod p
- Bob sends the cipher (g^y mod p, $m.g^{xy}$ mod p) to Alice
- Bob can't figure out Alice's private key, but he has encrypted his message by raising Alice's public key to the power of a random number
- Only Alice's secret key can reverse this process.

Example

- Bob chooses a password as 131513
- He wants to communicate this shared password to Alice without eavesdroppers being able to obtain it
- Bob now picks a random number, say 1000
- He has Alice's public key (150001, 7, 66436)
- He, therefore, calculates 7^{1000} mod 150001 which is 90,429
- He also calculates 131513. 66436^{1000} mod 150001 which is 57,422
- Now he sends these two numbers (90429, 57422) to Alice

Decryption Algorithm

- Alice receives the cipher (密码) from Bob, (g^y mod p, $m.g^{xy}$ mod p). we'll call these numbers c_1 and c_2.
- She uses her secret key to extract the message m.
- This is easy – she just computes c_2 / c_1^x
- And how does this work? $\frac{c_2}{c_1^x} = \frac{m \cdot g^{xy}}{g^{xy}} = m.$

Example

- Alice receives the cipher from Bob, (90429, 57422)
- She uses her secret key to compute $1 / c_1^x$
- It's best to avoid division when using modulo arithmetic
- $1 / c_1^x$ is the same as c_1^{p-1-x} which is 90429^{149887} mod 150001 which is 80802
- Now she multiplies 80802 by c_2 mod 150001 to yield the original message 131513
- This shared piece of information can now be used to encrypt documents sent between them

过程

- 1.爱丽丝与鲍勃事先互不认识，也没有可靠安全的沟通渠道，但爱丽丝现在却要通过不安全的互联网向鲍勃发送消息。
- 2.爱丽丝撰写好原文，原文在未加密的状态下称之为**明文x**
- 3.鲍勃使用密码学安全伪随机数生成器产生一对密钥，其中一个作为*公钥c*,另一个作为*私钥d*
- 4.鲍勃可以用任何方法发送*公钥c*给爱丽丝，即使伊夫在中间窃听到c也没关系。
- 5.爱丽丝用*公钥c*把**明文x**进行加密，得到**密文c(x)**
- 6.爱丽丝可以用任何方法传输**密文c(x)**给鲍勃，即使伊夫在中间窃听到**密文c(x)**也没问题
- 7.鲍勃收到密文，用自己的*私钥d*对密文进行**解密d(c(x))**，得到爱丽丝撰写的**明文x**
- 8.由于伊夫没有得到鲍勃的*私钥d*,所以无法得知**明文x**
- 9.如果爱丽丝丢失了她自己撰写的*原文x*，在没有得到鲍勃的*私钥d*的情况下，她的处境将等同伊夫，既无法通过鲍勃的*公钥c*和**密文c(x)**重新得到**原文x**

Ex.

加密和解密

加密

加密要用到公钥(N,e)。

假设Bob要向Alice发送加密信息m，他就要用Alice的公钥(N,e)对m进行加密。但m必须是整数（字符串可以取ascii值或unicode值），且m必须小于n。

所谓加密就是计算下式的c。

所谓加密就是计算下式的c

m^e ≡c(mod n)

假设m=65,Alice的公钥(3233,17),所以等式如下:

65^17 ≡ 2790 (mod3233)

所以c等于2790，Bob就把2790发给Alice。

Alice收到Bob发来的2790后，就用自己的私钥(3233,2755)进行解密.

c^d ≡ m(mod n)

也就是c的d次方除以n的余数就是m,

2790^2753 ≡ 65 (mod3233)

因此得到原文65.

Analogy

- Alice has told everyone what her lock is but only she has the key
- Bob attaches Alice’s lock to his message but he also adds his own lock on top of Alice’s lock
- Alice gets the message, uses her key on the original lock and both locks come off
- The double lock is necessary or else the message could be decrypted by an eavesdropper
- Bob must add an extra level of encryption that only he knows
- The genius of the system is that Alice is able to decrypt the cipher without needing to know what Bob has done

4.3 非对称加密计算（手算）

- Mod calculation (by hand)
- Cal private key (by hand)
- Decryption (by hand)

1. **by hand：**

Big Numbers

- Why is this allowed?
  - 27 = (2 x 11) + 5 → 27 mod 11 = 5
  - 27 x 3 = (2 x 11) x 3 + 5 x 3

    Irrelevant → (27 x 3) mod 11 = (5 x 3) mod 11 = 4
  - Only the remainder has any significance because when the multiples of 11 are multiplied they are still multiples of 11 and don't contribute
  - We only need to multiply the modulus remainder each time

- Calculate 7^{113} mod 150,001
- The trick is to break down 113 into simpler units
  - 7 mod 150001 = 7
  - 7^2 mod 150001 = (7 mod 150001)^2 = 7^2 = 49
  - 7^4 mod 150001 = (7 mod 150001)^2 = 7^2 = 2,401
  - 7^8 mod 150001 = 64,763
  - 7^{16} mod 150001 = 68,208
  - 7^{32} mod 150001 = 50,249
  - 7^{64} mod 150001 = 145,169

- 113 = (1 x 64) + (1 x 32) + (1 x 16) + (1 x 1)

- 113 = (1 x 64) + (1 x 32) + (1 x 16) + (1 x 1)
- 7^{113} mod 150,001 = (145,169 x 50,249 x 68,208 x 7) mod 150,001
- 7^{113} mod 150,001 = 66,436
- Of course, you can break 7^{113} up into any smaller powers that you like
- For example $7^{113} = 7^{50} \times 7^{50} \times 7^{13}$
- Find out the largest powers that your calculator can handle and use those

2. *计算机实现：*

How do I do this on my calculator?

- Given 7^4 mod 150001 = 2,401 find 7^8 mod 150001
- 7^8 mod 150001 = $7^4 \times 7^4$ = 2,401 x 2,401
- 2,401 x 2,401 = 5,764,801
- 5,764,801 / 150,001 = 38.43175046.......
- 38.43175046 – 38 = 0.43175046 ....
- 0.43175046 x 150,001 = 64,763
- 7^8 mod 150001 = 64,763

3. *代码实现：*

- How do we write a method for calculating modulus powers?

```
public static int modPow(int number, int power, int modulus){
     int result =1;
     for(int i=0;i<power;i++){
            result=result*number;
            result=result%modulus;
     }
     return result;
}
```

- This loop approach is extremely inefficient because it is O(n)
- We have to calculate every preceding modPow before arriving at the one we want
- This essentially defeats the purpose of the trapdoor because computing with the key is supposed to be far quicker than trying to hack the key

- We can write a recursive O(logn) method:

```
public static int modpow(int number, int power, int modulus){
        if(power==1)
            return number%modulus;
        else if (power % 2 ==0) {
            long halfpower=modpow(number, power/2, modulus);
            return (halfpower*halfpower) % modulus;
        }else{
            long halfpower=modpow(number, power/2, modulus);
            return (halfpower*halfpower*number) % modulus;
        }
}
```

- This is called exponentiation by squaring

$$\mathrm{Power}(x, n) = \begin{cases} 1, & \text{if } n = 0 \\ x \times \mathrm{Power}(x, n-1), & \text{if } n \text{ is odd} \\ \mathrm{Power}(x, n/2)^2, & \text{if } n \text{ is even} \end{cases}$$

历年卷习题 Cryptography

2018–Summer–Q2c

(c) Alice's ElGamal public key (p, g, $g^x modp$) is (23, 11, 9). Her private key is 2. Bob sends her the ciphertext (20, 7). What is the message? [5 marks]

CS211 Page 1 of 2 Summer 2018

2019–Summer–Q2b

(b) Alice's ElGamal public key (p, g, $g^x modp$) is (23, 11, 2). Obtain her private key by brute force. [8 marks]

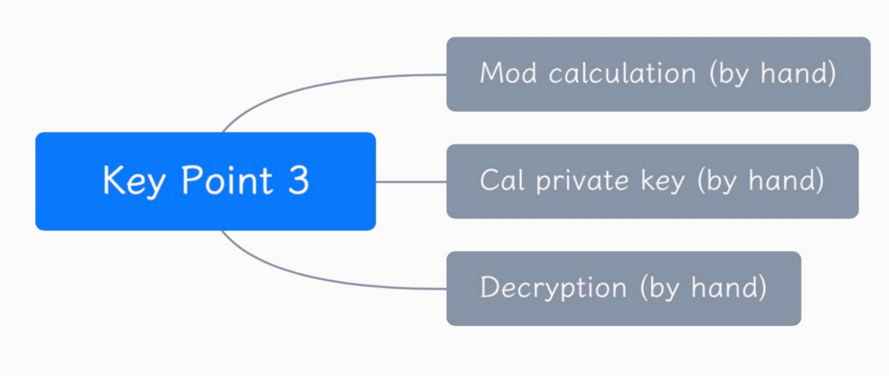
2019–Autumn–Q2b

(b) Show how to calculate 5^{23} mod 91 by hand, using a pen and paper algorithm.

参考：《RSA加密算法（公钥+私钥加密）》

http://t.csdn.cn/EpuBf

CS211 Note3 Review



CS211 Note-3

by Lance Cai

2022/07/05