Crypto 6

Note: this material is not intended to replace the live lecture for students.

Contents

Questions by a student

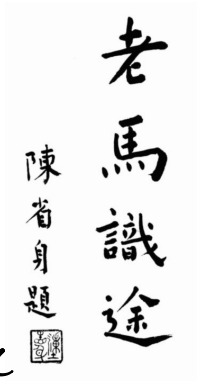
1) vector spaces over $\mathbb{Z}_2$

$S: \mathbb{Z}_2^6 \to \mathbb{Z}_2^4$

$S(x_1, x_2, x_3, x_4, x_5, x_6) = (s_1, s_2, s_3, s_4)$

It must not be close to linear ← formula

$l(x_1, x_2, x_3, x_4, x_5, x_6) = c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4 + c_5x_5 + c_6x_6$



3) $\forall x,y \quad f(x+y) = f(x) + f(y)$

$f(0) = 0 \Rightarrow f(0+0) = f(0) + f(0)$

$f(0) = f(0) + f(0)$

$0 = f(0)$ c.v.d.

$S(x) = L(x) + cte$ (constant)

4) DES has key 56 bits

→ $DES_{k_1} \circ DES_{k_2} = E_{k_1 k_2}$

57 bits ← Meet in the Middle

$DES_{k_1} \circ DES_{k_2} \circ DES_{k_3} = 3DES$

$E_{k_1 k_2} \circ DES_{k_3}$

↑ 57 bits ↑ 56 bits

56+56=112

112 > 56

112 ≈ $2^{112} + 2^{56}$

6.1 Groups and subgroups

What is a Group ?

Roughly speaking a group is a set with two operations $*$ and inversion.

Formally: By a group $(G, *)$ we mean a set G and a operation $*$ between elements of G such that:

i) any two elements a, b of G can be used to get another element $a * b \in G$;

ii) associativity : $(a * b) * c = a * (b * c)$;

iii) there is a neutral element e i.e. for all $a \in G$

$$a * e = e * a = a$$

iv) any element a has an inverse i.e. there is $b \in G$ such that $a * b = e$.

The cardinal $|G|$ is called **order of the group** G.

A **subgroup** H of G is a non empty subset $H \subset G$ closed with respect to $*$ containing the inverse of any of its elements.

Usually subgroups are defined by either generators or equations. To find generators from the equations is hard (Exercise 6.1.7).

Exercise 6.1.1

Show:
i) The neutral element e is unique.

ii) Given a the inverse is unique.

NOTE 6.1.2

Commutativity is not compulsory. The symmetric group S_n is not commutative if $n > 2$.

Exercise 6.1.3

Show that S_3 is not commutative.

6.1.4 Cyclic Group

A group $(G, *)$ is cyclic if there is $a \in G$ such that any other element b of G is of the form

$$b = \underbrace{a * a * a * a * a * \cdots * a}_{n-times}$$

In this case the element a is called **generator** of G. It is usual to write $b = a^n$.

6.1.5 Cyclic Subgroup generated by a

For a in G the cyclic subgroup or cyclic group generated by a and denoted by $\langle a \rangle$ is the set

$$\langle a \rangle = \{\cdots, a^{-3}, a^{-1}, 1, a, a^2, a^3, \cdots\}$$

The **order** of a is $|\langle a \rangle|$. Namely, the order of a is the smallest r such that $a^r = e$.

NOTE 6.1.6

Computing the order is a hard problem (see Exercise 6.4.9). Besides "brute force" the factorization of the cardinal $|G|$ is useful to compute the order of elements (Cf. Lagrange's theorem).

The group $\mathbb{Z}_p^* = \mathrm{GF}(p) \setminus \{0\}$ with p a prime number is cyclic. A generator g is also called **primitive element**.

6.1.1 Discrete Logarithm Problem (DLP)

Here the statement of DLP:

Given a cyclic group G with a generator g e.g. $G = \langle g \rangle$ and $y \in G$ find n such that:

$$y = g^n$$

If the operation of G is addition then the equation for n is : $y = n \cdot g$.

Exercise 6.1.7

Let p be a prime number and let g a primitive element of $\mathbb{Z}_p^*$. Given $A \in \mathbb{Z}_p^*$ consider the subgroup $H \subset \mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}$ defined as:

$$H = \{(x, y) \in \mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1} |\ g^x \cdot A^y = 1 \pmod{p}\}$$

Notice that H is defined by an equation and check that H is indeed a subgroup of $\mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}$.

Let $(a, 1)$ be a generator of H. Show that $g^{-a} = A$ where $-a$ is taken $\pmod{(p-1)}$.

NOTE 6.1.8

The DLP is hard as it is the computation of the order (Note 6.1.6). The above Exercise 6.1.7 is an special case of Hidden Subgroup Problem. Such problems share a "commutative" (abelian) nature, namely there is an abelian group involved. Thus these "abelian" problems can be solved efficiently by a Quantum Computer by using the Fourier Transform.

Fast computation of the order implies fast factorization of numbers of the form $N = p \cdot q$.

6.1.2 Computing powers a^n

Exercise 6.1.9

Compute

$$8^e \pmod p$$

where $p = 2^{20} - 2^3 - 1$ and $e = \frac{p+1}{4}$.

Exercise 6.1.10

Compute

$$8^e \pmod p$$

where $p = 2^{256} - 2^{32} - 977$ and $e = \frac{p+1}{4}$.

http://mathworld.wolfram.com/LandauSymbols.html
Let $n = (d_t d_{t-1} d_{t-2} \cdots d_0)_2$, $d_t = 1$, the binary representation (base-2).

6.1.11 Efficient computation of powers: square-and-multiply algorithm

```
T=e
For i=t downto i=0
    T = T * T
    if d_i = 1
       T = T * a
return(T)
```

Here is the **python** to compute $a^n \pmod q$:

6.1.12 Square-and-multiply in $\mathbb{Z}_q$

```
def squaremultiply(a,n,q):
    bina='{0:b}'.format(n)
    T=1
    amq=a%q
    for d in bina:
        T=(T*T)%q
        if d=='1': T=(T*amq)%q
    return(T)
```

6.1.3 Order of a group and of an element

Here we take $g \in \mathbb{Z}_n^*$ and consider the cyclic group $\langle g \rangle$ generated by g :

$$\langle g \rangle = \{g, g^2, g^3, \cdots, 1\}$$

Observe that there is a power w such that $g^w = 1$. The first such w is called the order (or period) of g. Notice that $w = |\langle g \rangle|$ i.e. the order of the element is the order of the cyclic subgroup generated by itself.

The cardinal $|\mathbb{Z}_n^*|$ is the Euler's totient function $\phi(n)$ that we will compute in 6.4.1.

6.1.13 Euler's ϕ function

Euler's $\phi(n)$ (also called totient) function counts the positive integers up to a given integer n that are relatively prime to n. That is to say, $\phi(n)$ is the cardinal of $\mathbb{Z}_n^*$ the invertible elements w.r.t. the product.

Lagrange's theorem

The order w of $g \in \mathbb{Z}_n^*$ is a divisor of $\phi(n)$.
The order of a subgroup divides the order of the group.

For example, let $g = 3$ in $\mathbb{Z}_{44}$. Then

$$\langle 3 \rangle = \{3, 9, 27, 37, 23, 25, 31, 5, 15, 1\}$$

So 3 has order 10 in $\mathbb{Z}_{44}^*$ and $10 | \phi(44) = \phi(4) \times \phi(11) = 2 \times 10$.

Exercise 6.1.14

Let $a \in \mathbb{Z}_n^*$. Show that $a^{\phi(n)} = 1 \pmod{n}$.

Exercise 6.1.15

Let $x \in \mathbb{Z}_{44}^* \setminus \langle 3 \rangle$. Show that there is $b \in \langle 3 \rangle$ such that $x = 43 \times b$

6.2 Solving the first degree equation $a \cdot x = 1 \pmod n$

$29 \cdot x = 1 \pmod{45}$

29	1
45	0
29	1
16	-1
13	2
16	-1
13	2
3	-3
1	14
3	-3
1	14
0	-45

So $x = 14$.

You can also use the following **python**:

6.2.1 Inverse in Python

```
''' inverse(r,N) = x ,
        such that r x = 1 (mod N).

'''
def inverse(r,N):
    x=1
    while (r*x)%N != 1 and  x<N:
              x+=1
    if  (r*x)%N == 1:
        return x
    else: return 'non invertible'
```

45. $x = 3 \pmod{45}$

$0 \cdot x = 3 \pmod{45}$

$0 = 3 \pmod{45}$

$\#\mathbb{Z}_p^* = p - 1 = \phi(p)$

For ex.: $\phi(45) = \#\mathbb{Z}_{45}^*$

$\phi(5) =$ trivial $\phi(9) =$ Chinese remainder Theorem

$\phi(5) \cdot \phi(9) = (5-1) \cdot \phi(9) = 4 \cdot \phi(9) = 4 \cdot 6 = 24$

$\mathbb{Z}_9 = \{0,1,2,3,4,5,6,7,8\}$

remove the one not coprime with 9

so it remains 6 elem. in the set

Exercise 6.2.2

Set $N = 16^{30} - 1$. Compute x such that $2x = 1 \pmod{N}$.

Exercise 6.2.3

Is the above **python** script 6.2.1 efficient ? That is to say, how many iterations are (in average) necessary to compute `inverse(r,N)` ?

Exercise 6.2.4

Let F_n be the sequence of Fibonacci $F_0 = 0, F_1 = 1$ and $F_n = F_{n-1} + F_{n-2}$ for $n > 1$. Compute x such that

$$F_{100} \cdot x \equiv 1 \pmod{F_{101}}$$

RING $(A, +, -, \times)$ $1, 0$

Field $(A, +, -, \times, /)$ $1, 0$

A^* unit or invertible

---

1) more about $ax = 1 \pmod{n}$

$d = g.c.d.(a, n) = g.c.d.(a, r)$ can exchange them, $n = aq + r$, $a < n$, $r < a$

but ϕ also: $d = ax + ny$ x, y

$a = 29$ and $n = 45 \Rightarrow 1 = g.c.d.(29, 45)$

$S \to \{29x + 45y : x, y \in \mathbb{R}\}$

$d = S = g.c.d.(29, 45)$ $d \in S$

$S \mid d$ $S \mid 29$ $S = 29x_0 + 45y_0$

6.2.1 Greatest common divisor (gcd)

Exercise 6.2.5

Here is a C code to compute $\gcd(a, b)$ from https://it.wikipedia.org/wiki/Architettura_ARM:

An algorithm that provides a good example of conditional execution is the subtraction-based Euclidean algorithm for computing the greatest common divisor. In the C programming language, the algorithm can be written as:

```c
int gcd(int a, int b) {
  while (a != b)  // We enter the loop when a < b or a > b, but not when a == b
    if (a > b)   // When a > b we do this
      a -= b;
    else         // When a < b we do that (no "if (a < b)" needed since a != b is checked in while condition)
      b -= a;
  return a;
}
```

How many iterations (loops) are going to be done to compute $\gcd(2^{200}, 2)$?

6.2.6 gcd in Python

```python
''' gcd(a,b) = [d,x,y] ,
        where $d$ is the g.c.d(a,b).
        The integers $x,y$ are such that $d=ax+by$.

'''
def gcd(a,b):
    import numpy as np
    M = np.array([[a,1,0],[b,0,1]])
    while M[0,0]*M[1,0] != 0:
              if M[0,0]<M[1,0]:
                  M[1]=M[1] - (M[1,0] // M[0,0])*M[0]
              else:
                  M[0]=M[0] - (M[0,0] // M[1,0])*M[1]
    if M[0,0] == 0:
             return M[1]
    else:
             return M[0]
```

Exercise 6.2.7

Is the above **python** script 6.2.6 efficient ? That is to say, how many iterations are (in average) necessary to compute `gcd(a,b)` ?

6.3 Quadratic residues: solving $x^2 = r \pmod n$

Teorema 6.3.1 ▶ Euler's Criterion

Let p be a prime number and r an integer non divisible by p. Then r is a quadratic residue modulo p if and only if

$$r^{\frac{p-1}{2}} \equiv 1 \pmod p$$

Group of quadratic residues

1) The set G of quadratic residues is a subgroup of $\mathbb{Z}_n^*$.

2) In case $n = p$ is a prime number $p = 2^s q + 1$ where q is odd then G has order $2^{s-1}q$.

3) If r is a quadratic residue modulo a prime $p \equiv 3 \pmod 4$ then a square root of r is $r^{\frac{p+1}{4}}$.

4) If r is a quadratic residue modulo a prime $p \equiv 1 \pmod 4$ then there is an efficient algorithm to find a square root e.g. Algorithm of Tonelli-Shanks.

Exercise 6.3.2

Check 1),2) and 3) above.

6.3.1 Two algorithms: Tonelli-Shanks and Cipolla

Let $p > 2$ a prime number and $p-1 = 2^s \times q$ where q is odd. The key observation of Tonelli-Shanks algorithm to solve $x^2 = n \pmod p$ is:

Fact TS if $t^{2^{s-1}} = 1 \pmod p$ then there is b^2 such that $t \times b^2 = 1 \pmod p$.

Now if n is a quadratic residue then $n^{\frac{(p-1)}{2}} = 1 \pmod p$. So if $r = n^{\frac{(q+1)}{2}}$ then

$$r^2 = n \times t \pmod p$$

where $t = n^q$ satisfies $t^{2^{s-1}} = 1 \pmod p$. Hence setting $x = r \times b$ we get

$$x^2 = r^2 \times b^2 = n \times t \times b^2 = n \pmod p .$$

Thus we got x such that $x^2 = n \pmod p$.

Exercise 6.3.3

Find x such that $x^2 = 2082 \pmod{6121}$.

Hint: setting $p = 6121$ first check that 2082 is indeed a quadratic residue $\pmod p$ i.e. $2082^{\frac{p-1}{2}} = 1 \pmod p$. So notice that $\frac{p-1}{2} = 4 \times 765$ and set $r = n^{\frac{765+1}{2}}$. Check that

$$r^2 = 2082 \times 2082^{765} = -2082 \pmod{6121}.$$

Then look for t such that $b^2 = -1 \pmod{6121}$ and use it.

Exercise 6.3.4

Let $z \in \mathbb{Z}_p$ such that $z^{2^{s-1}} = -1 \pmod p$ and let $t^{2^{s-1}} = 1 \pmod p$. Show that there is $b \in \mathbb{Z}_p$ such that

$$(t \times b^2)^{2^{s-2}} = 1 \pmod p .$$

Notice that by repetition of the above argument it follows that there is b such that

$$t \times b^2 = 1 \pmod p$$

as claimed in **Fact TS**.

6.3.5 Cipolla's algorithm

Let r be a quadratic residue modulo p. To find a square root $u \in \mathbb{Z}_p$ find $s \in \mathbb{Z}_p$ such that $s^2 - r$ is not a quadratic residue. Set $\mathrm{A} = \begin{bmatrix} s & s^2 - r \\ 1 & s \end{bmatrix}$. Compute u as

$$\mathrm{A}^{\frac{p+1}{2}} = \begin{bmatrix} u & 0 \\ 0 & u \end{bmatrix}$$

Exercise 6.3.6

Find x such that $x^2 = 5 \pmod{41}$
Hint: use that $z = 28 \pmod{41}$ satisfies $z^4 = -1 \pmod{41}$.

Exercise 6.3.7

Find u such that $u^2 = 4799 \pmod{1002257}$. Hint: **python**.

NOTE 6.3.8

To explain why Cipolla's method works it is necessary to use Frobenius's automorphism hence to develop more Galois Theory that is beyond this course.

6.4 The Chiñese Remainder Theorem (CRT)

The CRT allows us to compute in $\mathbb{Z}_{nm}$ regarding its elements as pairs. Here is an example:

$\mathbb{Z}_{24} \approx \mathbb{Z}_3 \times \mathbb{Z}_8$

0	(0, 0)
1	(1, 1)
2	(2, 2)
3	(0, 3)
4	(1, 4)
5	(2, 5)
6	(0, 6)
7	(1, 7)
8	(2, 0)
9	(0, 1)
10	(1, 2)
11	(2, 3)
12	(0, 4)
13	(1, 5)
14	(2, 6)
15	(0, 7)
16	(1, 0)
17	(2, 1)
18	(0, 2)
19	(1, 3)
20	(2, 4)
21	(0, 5)
22	(1, 6)
23	(2, 7)

So to compute $13 \times 18 \pmod{24}$ we compute $(1,5) \times (0,2) = (0,10) = (0,2)$ hence

$$13 \times 18 = 18 \pmod{24}$$

Exercise 6.4.1

By using the above table about $\mathbb{Z}_{24} \approx \mathbb{Z}_3 \times \mathbb{Z}_8$ compute a similar table for $\mathbb{Z}_{24}^* \approx \mathbb{Z}_3^* \times \mathbb{Z}_8^*$. Is $\mathbb{Z}_{24}^*$ a cyclic group ?

Exercise 6.4.2

Show that $\mathbb{Z}_{22}^*$ is a cyclic group.

NOTE 6.4.3

The question of when $\mathbb{Z}_n^*$ is a cyclic group was answered by Gauss. Here it is: $\mathbb{Z}_n^*$ is a cyclic if and only if n= $2, 4, p^k$ or $2p^k$ for p an odd prime.

Exercise 6.4.4

Find $x < 3 \times 5 \times 7$ such that

$$\begin{cases} x = 2 \pmod 3 \\ x = 3 \pmod 5 \\ x = 2 \pmod 7 \end{cases}$$

Exercise 6.4.5

Compute x such that $17 \times x = 1 \pmod{24}$.

Exercise 6.4.6

Compute ALL x such that $13 \times x = 18 \pmod{24}$.

Exercise 6.4.7

Compute ALL x such that $x^2 = 1 \pmod{24}$.

Exercise 6.4.8

Find $x < 809933$ such that

$$\begin{cases} x^2 = 62953 \pmod{809933} \\ x^2 = 504539 \pmod{854429} \end{cases}$$

Exercise 6.4.9

Let $N = p \cdot q$ be a product of two prime numbers. Assume you know N but not its prime factors. Assume you know $a \in \mathbb{Z}_N^*$ and its order $2d$. Then you can efficiently find the primes p, q. Hint: $(x-1)(x+1) = x^2 - 1$ and Extended Euclidean Algorithm.

6.4.1 CRT and Euler's ϕ function

Let $N = p^r \times q^s \times \cdots$ the factorization of N into powers of different prime numbers $p, q, \ldots$. By computing with pairs we get

$$\phi(N) = \phi(p^r) \times \phi(q^s) \times$$

$|\mathbb{Z}_{p^r}^*|$

$$\phi(p^r) = (p-1) \times p^{r-1}$$

For example $\phi(7^2) = 6 \times 7 = 42$ because:

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42
43	44	45	46	47	48	49

namely, $\phi(7^2) = 7^2 - 7 = 7 \times (7-1) = 7 \times 6 = 42$.

NOTE 6.4.10

Observe that to compute $\phi(n)$ using the above formulae it is necessary to factorize the number n.

Exercise 6.4.11

Let $N = p \cdot q$ be a product of two prime numbers. Assume you know N but not its prime factors p, q. Assume you know $\phi(N)$. Then you can efficiently find the primes p, q. Hint: find the roots of the equation $x^2 + (\phi(N) - N - 1)x + N = 0$.

6.5 Prime and pseudoprime numbers

Teorema 6.5.1 ▶ Fermat's Little Theorem

Let p be a prime number and r any integer. Then

$$r^p \equiv r \pmod{p}$$

6.5.1 Tests, Criteria and Certificates

Rabin-Miller: starting clue

Let n an odd number. Set $n-1 = 2^s \cdot d$ where d odd. If n is a prime number and $a \neq 0 \pmod{n}$ then either

$$a^d \equiv 1 \pmod{n}$$

or

$$a^{2^r \cdot d} \equiv -1 \pmod{n}$$

for some $0 \leq r \leq s-1$.

6.5.2 Rabin-Miller probabilistic test

```
Input #1: n > 3, an odd integer to be tested for primality
Input #2: k, the number of rounds of testing to perform
Output: "composite" if n is found to be composite, "probably prime" otherwise

write n as 2^r·d + 1 with d odd (by factoring out powers of 2 from n - 1)
WitnessLoop: repeat k times:
   pick a random integer a in the range [2, n - 2]
   x ← a^d mod n
   if x = 1 or x = n - 1 then
      continue WitnessLoop
   repeat r - 1 times:
      x ← x^2 mod n
      if x = n - 1 then
         continue WitnessLoop
   return "composite"
return "probably prime"
```

NOTE 6.5.3

The running time is $O(k \cdot \log^3(n))$. The mistake probability declaring prime a composite number is at most 4^{-k} [Rabin77, Theorem 2, page 134].

https://inventwithpython.com/cracking/chapter22.html

6.5.2 Generation of large prime numbers

Problem: How to construct a random n-bit prime number p ?

Idea of the solution: Pick a n-bit random number X from

$$[2^{n-1}, 2^n - 1]$$

and check if it is a prime number.

Iteration until the test gives a positive answer.

Running time ?

Teorema 6.5.4 ▶ Prime Number Theorem

The total number of prime $< N$ is (roughly)

$$\frac{N}{\log(N)}$$

So the number of primes in the interval $[2^{n-1}, 2^n - 1]$ is roughly

$$\frac{2^{n-1}}{n}$$

hence the probability that a random number in that range is a prime is $\frac{1}{n}$.

The failing probability after t trials is $(1 - \frac{1}{n})^t$. By putting $t = \alpha \cdot n$ we can estimate ($n >> 0$):

$$(1 - \frac{1}{n})^t = (1 - \frac{1}{n})^{\alpha \cdot n} = \left((1 - \frac{1}{n})^n\right)^{\alpha} \approx (e^{-1})^{\alpha} = e^{-\alpha}$$

So for $n = 1000$ and $t = 3000$ we have $e^{-\alpha} \approx 0.05$.

https://www.cs.purdue.edu/homes/hmaji/teaching/Fall%202018/lectures/11.pdf

6.6 Bibliography

Books I used to prepare this note:

[Paar10] Paar, Christof, Pelzl, Jan, *Understanding Cryptography, A Textbook for Students and Practitioners*, Springer-Verlag, 2010.

Here a list of papers:

[Rabin77] Michael O Rabin,; *Probabilistic algorithm for testing primality*, Journal of Number Theory, Volume 12, Issue 1, 1980, Pages 128-138. https://www.sciencedirect.com/science/article/pii/0022314X80900840#!

and some interesting links:

http://gauss.math.luc.edu/greicius/Math201/Fall2012/Exercises/ChineseRemainderThm.pdf

https://www.whitman.edu/mathematics/higher_math_online/section03.07.html

https://pynative.com/python-range-function/

https://inventwithpython.com/cracking/chapter22.html

https://cseweb.ucsd.edu/~mihir/papers/gb.pdf

https://en.wikipedia.org/wiki/Euler%E2%80%93Jacobi_pseudoprime

https://en.wikipedia.org/wiki/Baillie%E2%80%93PSW_primality_test

https://en.wikipedia.org/wiki/Strong_pseudoprime

https://en.wikipedia.org/wiki/Solovay%E2%80%93Strassen_primality_test

http://www.dtc.umn.edu/~odlyzko/doc/discrete.logs.hff.pdf

https://en.wikipedia.org/wiki/Fermat_primality_test