Cryptography.
Lecture 4.

Elisa Lorenzo García

22nd February 2016

Contents

We have an intuitive idea of what cryptography is: its fundamental objetive is to enable two people, usually refered to as Alice (A) and Bob (B), to communicate over an insecure channel in such a way that an opponent, Oscar (O) or Eve (E) (because eavesdropper), cannot understand what is being said. There are two different approaches, one that is design the proper tool to insure secrecy, and the other one, that is the attack of these tools to find out their weaknesses.

Cryptography has been used for long. Going back to Caesar, one finds the example of "the shaved slave". The main weakness (besides the fact that you have to wait for the hair of the slave to grow ...) is that the slave can speak. The conclusion, is that the secrecy cannot be based on the method, it has to be based on additional secret information, called keys.

The main references for today's lecture are chapters III, IV and VI in [1], chapters 1 and 2 in [3], chapter 6 in [2] and chapters $6, 7$ and 16 in [4].

1 Basic definitions and examples

The information that Alice wants to send to Bob will be the plaintext. She will cipher or encrypt the plaintext to obtain a ciphertext with the help of a key. Bob, who knows the key will then decipher the ciphertext. While, Oscar, who does not, can try to decrypt it, in order to get the plaintext.

Definition 1.1. *A secret key cryptosystem is a* 3*-tuple* $(\mathcal{P}, \mathcal{C}, \mathcal{K})$ *where the following conditions are satisfied:*

1. $\mathcal{P}$ *is a finite set if possible plaintexts;*
2. $\mathcal{C}$ *is a finite set of possible ciphertexts;*
3. $\mathcal{K}$ *is a finite set of possible keys called keyspace;*
4. *Fir each* $K \in \mathcal{K}$*, there is an encryption rule* $\mathcal{E}_K : \mathcal{P} \to \mathcal{C}$ *and a decryption rule* $\mathcal{D}_K : \mathcal{C} \to \mathcal{P}$ *such that* $\mathcal{D}_K \otimes \mathcal{E}_K = Id$*. (So,* $\mathcal{E}_K$ *is injective).*

First of all, Alice and Bob have to choose a random key $K \in \mathcal{K}$ (we will discuss later public-key cryptosystems). A message is a string $x = x_1...x_n$. Each x_1 is encrypted using $\mathcal{E}_K$ as $y_i = \mathcal{E}_K$. She sends the ciphertext $y = y_1...y_n$.

1.1 The shift cipher

Let represent the 26 usual letters by elements in $\mathbb{Z}/26\mathbb{Z}$ (for instance $A \to 0$, $B \to 1, ..., Z \to 26$). We consider then $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}/26\mathbb{Z}$. For $K \in \mathcal{K}$, we define

$$\mathcal{E}_K : x \to x + K.$$

The decryption rule is given by

$$\mathcal{D}_K : y \to y - K.$$

In spite of its extreme weakness, there are only 26 possibilities for the key, it was used by South officers during the American Civil War and even by the Russian army in 1915.

1.2 The substitution cipher

We take $\mathcal{P} = \mathcal{C} = \mathbb{Z}/26\mathbb{Z}$ and $\mathcal{K} = S_{26}$, so the encryption and decryption rules for $\sigma \in \mathcal{K}$ are

$$\mathcal{E}_K : x \to \sigma(x),\ \mathcal{D}_K : y \to \sigma^{-1}(y).$$

1.3 The Vigenère cipher

We define $\mathcal{P} = \mathcal{C} = \mathcal{K} = (\mathbb{Z}/26\mathbb{Z})^m$. For a key $(k_1, ..., k_m) \in \mathcal{K}$, we define

$$\mathcal{E}_K(x_1, ..., x_m) = (x_1 + k_1, ..., x_m + k_m), \text{ and}$$

$$\mathcal{D}_K(y_1, ..., y_m) = (y_1 - k_1, ..., y_m - k_m).$$

For $m = 1$, to recover the shift cipher.

1.4 The Hill cipher

We put $\mathcal{P} = \mathcal{C} = (\mathbb{Z}/26\mathbb{Z})^m$, and $\mathcal{K} = \mathrm{GL}_m(\mathbb{Z}/26\mathbb{Z})$, and for a key $K \in \mathbb{K}$, we define $\mathcal{E}_K(x) = xK$ and $\mathcal{D}_K(y) = yK^{-1}$.

The Hill cipher is a generalization of Vigenère cipher.

In conclusion, we see that this cryptographic constructions are based on two principles: substitutions of letters and affine transformation. In all this cases a exhaustive attack works or otherwise a statistics one, see [4]

Of course, as the set of keys is always finite, exhaustive research is always an option and thus any cryptosystem can be theoretically broken. However, the opponent has not an infinite power of computation or unlimited time, so if the keyspace is too big (which means more than 2^{60} keys), one must have to try clever methods. For instance, statistical methods. See 1.3.1 in [3].

Besides the encryption of the text we have to deal with issues on the protocol: how to be sure that Alice is the sender? How to be sure that the message has not been modified? Encryption is not enough for that and we will need to introduce new notions (e.g. signature).

2 Some cryptosystems

We list some of the main problems people deal with in cryptography.

1. Confidentiality: is the property that an information is not available to unauthorized people.

2. Integrity: is the way to prevent an unauthorized modification of the data.

3. Authentication: consists in checking the identity of the different elements involved in a communication.

4. Non-repudation: is a mechanism to prevent to deny a contract.

5. Signature: is a system to prove authentication of the sender, integrity of data and non-repudiation.

6. Certification: is the way, a trusted entity validates a certain information.

7. Key management: is the problem of distribution, integrity, ... of the keys.

8. Proof (Zero-knowledge proof): a proof of being in possession of a secret (without giving extra information).

Of course, we will not have time to discuss them, but you can get an idea of the complexity of the subject. We now list some of the most famous cryptosystems.

2.1 RSA

The possibility of the present scheme, called a public key cryptosystem, was first publicly suggested by Delffie and Hellman in their classic paper. However, they did not yet have a practical implementation. In the next years, several methods were proposed. The most successful one, based on the idea that factorization of integers into their prime factors is hard, was proposed by Rivest, Shamir, and Adleman in 1977, and is known as the RSA algorithm.

Bob chooses two distinct large primes p and q and multiplies them together to form $n = pq$. He also chooses an encryption exponent e such that $\gcd(e, (p-1)(q-1)) = 1$. He sends the pair (n, e) to Alice but keeps the values of p and q secret. In particular, Alice, who could possibly be an enemy of Bob, never needs to know p and q to send her message to Bob securely. Alice writes her message as a number m. If m is larger than n, she breaks the message into blocks, each of which is less than n. However, for simplicity, let us assume that $m < n$. Alice computes $c \equiv m^e \bmod n$ ans sends c to Bob. Since Bob knows p and q, he can compute $(p-1)(q-1)$ and therefore can find the decryption exponent d with $de \equiv 1 \bmod (p-1)(q-1)$.

The primes p, q are taken randomly and with at least 100 digits in order to get a good cryptosystem.

Remark 2.1. *The problem of factorizing n is equivalent to computing $\phi(n)$.*

Example 2.2. *First, we choose integer (usually larger ...) $k = 3$, and $l = 4$, and we work with an alphabet of $N = 26$ letters. To send the message "YES" to a user A with enciphering key $(n, e) = (46927, 39423)$, we find the numerical equivalent of "YES", namely: $24 \cdot 26^2 + 4 \cdot 26 + 18 = 16346$ (we split the message in blocks of size k), and then we compute 16346^{39423} mod 4927, which is $21166 = 1 \cdot 26^3 + 5 \cdot 26^2 + 8 \cdot 26 + 2$ ="BFIC" (blocks of length l). How do A read the message?*

The RSA is still used (with n larger than $2^{2^{11}}$) and it is a safe cryptosystem, but quantum computers could eventually break them .....

2.2 DLP

Fix a prime p. Let α and β be nonzero integers mod p and suppose $\beta = \alpha^x \bmod p$. The problem of finding x is called the discrete logarithm problem. If n is the smallest positive integer such that $\alpha^n = 1 \bmod p$, we may assume $0 \leq x < n$, and then we denote $x = L_\alpha(\beta)$.

A function $f(x)$ is called a one-way function if $f(x)$ is easy to compute, but, given y, it is computationally infeasible to find x with $f(x) = y$. Modular exponentiation is such an example, and multiplication of large primes can also be regarded as a probable one-way function.

Under certain conditions, there are different attacks for the DLP: the Pohlig-Hellman algorithm, the "Baby Step, Giant Step" and the Index Calculus among others.

3 Elliptic curves cryptosystems

An elliptic curve is a genus 1 curve given by an equation $E: y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6/\mathbb{F}_{||}$. If the characteristic of the field is different from $2, 3$, we can make $a_1 = a_2 = a_3 = 0$ (that is, we can find an $\mathbb{F}_q$-isomorphic curve with these parameters equal to zero). We denote by $0 = (0:1:0)$ the point at infinity. We can define a group law on the points of an elliptic curve where 0 plays the roll of the zero element.

The group law can be just defined by explicitly algebraic formulas, but it is nice to see how this group law looks for elliptic curves with real coefficients.

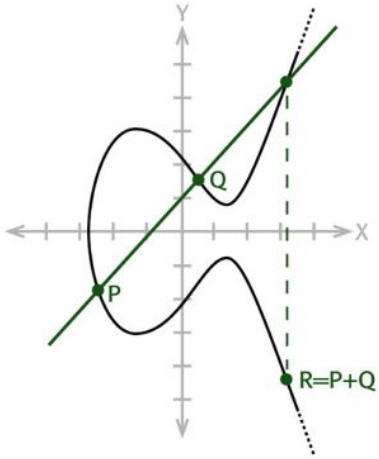

Figure 1 – The group law.

Algebraically, the group law for two points $P_1 = (x_1, y_1)$, and $P_2 = (x_2, y_2)$ is given by $P_3 = P_1 + P_2 = (x_3, y_3)$, where

$$x_3 = m^2 - x_1 - x_2, \;\; y_3 = m(x_1 - x_3) - y_1,$$

and

$$m = \begin{cases} (y_2 - y_1)/(x_1 - x_2) & \text{if } P_1 \neq P_2 \\ (3x_1^2 + b)/(2y_1) & \text{if } P_1 = P_2 \end{cases}.$$

If the slope m is infinite, then $P_3 = 0$.

In the lecture it has been discussed how to use Elliptic curves to factor integers, via Lenstra algorithm. For a reference, see [5] We construct a cryptosystem based on the DLP in the group of rational points of the curve (even if the group law is denoted additively, it can be also consider as a multiplicative one).

The crucial issue concerning the security of an elliptic curve cryptosystem is whether the discrete logarithm problem for the group $E(\mathbb{F}_q)$ of an elliptic curve is computationally infeasible. By the Polish-Hellman algorithm, if the order of the point P can be factored into small primes, then the DLP can be computes easily. Therefore, we usually choose a point P with large prime order. As the order of P divides the order of $E(\mathbb{F}_q)$, this forces us to find elliptic curves E with $E(\mathbb{F}_q)$ divisible by a large prime. This requirement raises the question of how to count the number of points of a given elliptic curve. the first polynomial-time algorithm for counting the number of $\mathbb{F}_q$-rational points of a given elliptic curve over $\mathbb{F}_q$ was designed by Schoof and improved later by Elkies-Atkin (we will see this algorithm in Lecture 7).

Consider an elliptic curve $E : y^2 = x^3 + Ax + B$ for some $A, B \in \mathbb{F}_q$ with $4A^3 + 27B^2 \neq 0$. Choose a point $P \in E(\mathbb{F}_q)$ such that the order n of P is a large prime, which is bigger than p. Choose a random integer d and compute the point $Q = [d]P$. All parameters except d are public, while the discrete logarithm d is kept secret. To sign a message $m \in \mathbb{Z}/n\mathbb{Z}$, we choose a random $k \in (\mathbb{Z}/n\mathbb{Z})^*$ and compute $[k]P = (x_1, y_1)$. Put $r = x_1$ and compute $s := k^{-1}(m + dr) \in \mathbb{Z}/n\mathbb{Z}$. The pair (r, s) is the signature of the message m. To verify the signature, we do the following:

i) Compute $w = s^{-1} \in \mathbb{Z}/n\mathbb{Z}$ (note that if $s = 0$, we choose another random k until we get $s \neq 0$).

ii) Compute $u_1 = mw \bmod n$ and $u_2 = rw \bmod n$.

iii) Compute $X = [u_1]P \oplus [u_2]Q$;

iv) if $X = 0$, then reject the signature, otherwise compute $v = x_2 \bmod$ n.

v) Accept the signature if an only if $v = r$.

The above scheme works properly. Indeed, if a signature (r, s) on a message m was generated, then $s = k^{-1}(m + dr) \bmod n$. Rearranging gives

$$k \equiv s^{-1}(m + dr) \equiv wm + wrd \equiv u_1 + u_2 d \bmod n.$$

Thus, $X = [u_1]P \oplus [u_2]Q = [u_1 + u_2 d]P = [k]P$, and so $v = r$ as required.

We will not discuss it today, but there are also higher genus curves cryptosystems. It was proved that genus greater or equal than 4 were not safe enough, genus 3 can be theoretically broken, but genus 2, for so many reason that I am not going to explain, are a really good alternative to elliptic curve cryptosystems.

4 Exercises

Exercise 4.1. *Suppose that the following* 40*-letter alphabet is used for all plaintexts and ciphertexts: A-Z with numerical equivalents* $0 - 25$*, blank*$= 26$*,* . $= 27$*,* ? $= 28$*,* \$ $= 29$*, the*

numerals $0-9$ *with numerical equivalents* $30-39$. *Suppose that plaintext message has values* $k=2, l=3, 40^2 < n < 40^3$.

a) Send the message "SEND \$7500" to a user whose enciphering key is $(n,e)=(2047,179)$.

b) Break the code by factoring n *and then computing the deciphering key* (n,d).

c) Explain why, even without factoring n*, a codebreaker could find the deciphering key rather quickly. In other words, why is (in addition to its small size)* 2047 *a particularly bad choice for* n*?*

Exercise 4.2. *Look for the Polish-Hellman method and compute for* $p=19$ *the discrete logarithm* $L_2(14)$.

Exercise 4.3. *Compute the order of the point* $(0,0)$ *in* $E: y^2+y=x^3+x/\mathbb{F}_7$.

Exercise 4.4. *Consider the elliptic curve* $E: y^2=x^3+x+4$ *defined over* $\mathbb{F}_{23}$. *We have* $\#E(\mathbb{F}_{23})=29$. *It is clear that* $P=(0,2)$ *is a point on the curve, and moreover, it generates all the* $\mathbb{F}_{23}$*-rational points. Sign a message* $m=10$ *and verify the signature.*

References

[1] N. Koblitz, *A course in Number Theory and Cryptography*, Springer, 1991.

[2] H. Niederreite, C. Xing, *Algebraic geometry in coding theory and cryptography*, Princeton University Press, 2007.

[3] C. Ritzenthaler, Lecture notes *Cryptology course*, 2006.

[4] W. Trappe, L.C. Washington, *Introduction to Cryptography with Coding Theory*, Pearson Education International, Prentice Hall, 2002.

[5] H. Lenstra, *Factoring Integers with Elliptic Curves*, Annals of mathematics, Vol 126, No(3) pp. 649-673