Solutions for week 6, Cryptography Course - TDA 352/DIT 250

In this weekly exercise sheet: you will go through for all the different topics that we have seen during the course. Almost every question can be present in the exam.

Easy

---

1. **Definition (Semantic Security Game)**:
   Let (E, D) be a encryption scheme on $(\mathcal{M}, \mathcal{K}, \mathcal{C})$. Let $\mathcal{A}$ be an Adversary and $\mathcal{C}$ be the Challenger. We define the **semantic security game** as the game with the following procedure:

   (a) $\mathcal{A}$ challenges $\mathcal{C}$ in the semantic security game

   (b) $\mathcal{A}$ chooses m_0, m_1 with $\text{len}(m_0) = \text{len}(m_1)$, from some distribution D over $\mathcal{M}$

   (c) $\mathcal{A}$ sends to $\mathcal{C}$ the messages m_0 and m_1

   (d) $\mathcal{C}$ selects at random a secret key $k \xleftarrow{R} \mathcal{K}$

   (e) $\mathcal{C}$ flips a random coin, i.e. $\mathcal{C}$ picks $b \in \{0, 1\}$ uniformly at random

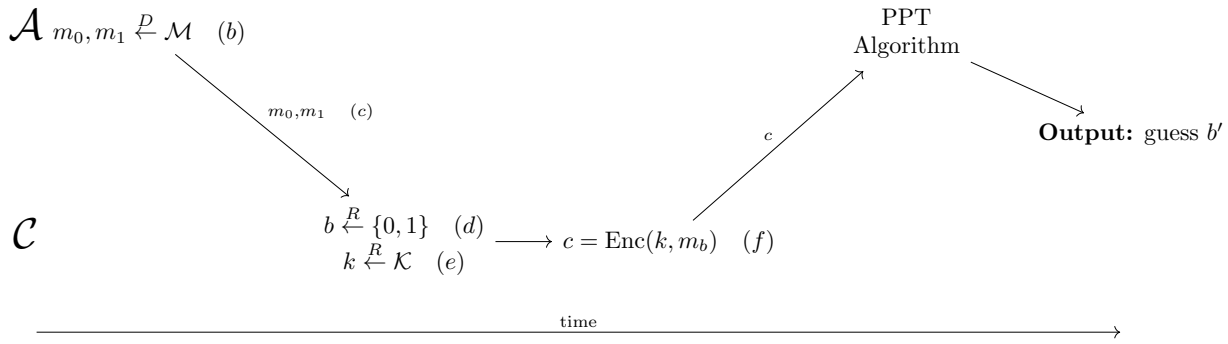
   (f) $\mathcal{C}$ encrypts m_b using k and obtains the ciphertext c, i.e. $c = \text{Enc}(k, m_b)$

   (g) $\mathcal{C}$ sends c to $\mathcal{A}$

   (h) $\mathcal{A}$ uses a probabilistic polynomial time (PPT) algorithm to obtain a guess b' for which bit $\mathcal{C}$ chose in step (e)

   (i) $\mathcal{A}$ output his guess b'

   The attacker $\mathcal{A}$ wins the semantic security game if $b = b'$.
   As a diagram:



   Definition (Semantic Security):
   Let (E, D) be a encryption scheme on $(\mathcal{M}, \mathcal{K}, \mathcal{C})$. The encryption scheme is **semantically secure** if for all the PPT adversaries $\mathcal{A}$, it holds that

$$\Pr(b' = b) \leq \frac{1}{2} + \epsilon \qquad \text{where } \epsilon \text{ is a negligible value}$$

   Definition: One Time Pad (OTP)
   The OTP has$\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0, 1\}^n$ where $\mathcal{M}$ is the message-space, $\mathcal{K}$ is the key-space and $\mathcal{C}$ is the ciphertext-space.
   Let $m \in \mathcal{M}$, $k \in \mathcal{K}$ and $c \in \mathcal{C}$. The OTP encryption algorithm is defined as

$$\text{Enc}(k, m) = m \oplus k$$

The OTP decryption algorithm is defined as

$$\text{Dec}(k, c) = c \oplus k$$

Proof: OTP is perfect secret.
To prove that the OTP is semantic secure, you should observe that $\mathcal{M} = \mathcal{K} = \mathcal{C}$. This implies $|\mathcal{M}| = |\mathcal{K}|$.
Therefore, we just need to prove that for every two messages m_0, m_1 and for every ciphertext c, it holds that $\Pr(\text{Enc}(k, m_0) = c) = \Pr(\text{Enc}(k, m_1) = c)$ for some random key $k \in \mathcal{K}$.
We can rewrite $\text{Enc}(k, m_0) = c$ as $k \oplus m_0 = c$, from which we have $k = c \oplus m_0$ (a similar reasoning hold for m_1).
Let us fix m_0 and c, then $\Pr(\text{Enc}(k, m_0) = c) = \Pr(k = c \oplus m_0) = (\text{ the probability of choosing randomly the keys } k \text{ such that } k = c \oplus m_0\ ) = \frac{1}{|\mathcal{K}|}$.
The same reasoning (and probability) holds for m_1. This concludes the proof of the fact that the OTP encryption scheme is perfect secret.

Using the tricks explained in the solution of exercise 6:

- If $b = 0$, we have $c = k \oplus m_0$
- If $b = 1$, we have $c = k \oplus m_1$

We don't have any strange behaviour. We don't have plaintext informations. We don't have repetitions. We don't have any clue on how to divide the problem.

Trick 5: *if you not able to find anything new, try combining what you have.* A lot of times, you should look at your information from a different point of view: you should *combine what you know* and see if this tells you something new.

We know that the OTP is perfectly secure and so $\Pr(k \oplus m_0 = c) = \Pr(k \oplus m_1 = c)$ for any pair of (distinct) messages. This equality means that there exists an unique key k_0 such that $m_0 \oplus c = k_0$ and an unique k_1 such that $m_1 \oplus c = k_1$. Since m_0 and m_1 are different, also k_0 and k_1 must be different.

Going back to the exercise, we can compute $c \oplus m_0$ (where c is the ciphertext returned by the challenger $\mathcal{C}$). According to the value of b, we can distinguish the following two cases:

- $b = 0$, in this case $c \oplus m_0 = k$
- $b = 1$, in this case $c \oplus m_0 = k \oplus m_1 \oplus m_0$

Since $m_0 \neq m_1$, the two messages should be different in at least 1 bit. Without loss of generality, let us assume that the bit in which m_1 differs from m_2 is the first bit.
Lets denote the first bit of the key k as $\hat{k} \in \{0, 1\}$. Observe that the first bit of $m_1 \oplus m_0$ will be 1 since it is the bit in which they differ.
We have

- If we compute $\hat{k}$, $b' = 0$
- If we compute $\hat{k} + 1$, $b' = 1$

Since k is uniformly and randomly generated, we have that every bit of k will be uniformly and randomly generate (try to think using exercise 7a and the *"coin flipping"*). This means that $\hat{k}$ is uniform and random in $\{0, 1\}$. So the probability $\Pr(\hat{k} = 0) = \Pr(\hat{k} = 1) = \frac{1}{2}$.
Since we are trying to predict the result of a random coin flip, we can conclude that our best guess for b' is flip a coin, i.e. $\frac{1}{2}$. So we (the adversary) have no non-negligible advantage.
Therefore the OTP cipher is semantically secure.

2. **Definition:** A function $\mathcal{F} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ is a *Pseudo Random Function* if the function $\mathcal{F}_k = \mathcal{F}(k, \cdot)$, for a uniformly random selected key $k \xleftarrow{R} \mathcal{K}$ is *indistinguishable* from a function chosen uniformly ar random from the set of all possible functions from $\mathcal{M}$ to $\mathcal{C}$.

Definition: A block cipher is a cipher (E, D) over $\mathcal{K}, \mathcal{M}, \mathcal{C}$ with $\mathcal{M} = \mathcal{C}$ for which

- $D(k, E(k, m)) = m$ for all $m \in \mathcal{M}$
- It has a *round* structure that is iterated for a fixed number r of time
- It has a key-scheduler that expands the secret key k into r-round keys

Let (E, D) be a block cipher.
ECB: in ECB, the data M is divided into blocks $\{m_i\}$ and each block is separately encrypted using E and the key k. If we denote with c_i the i-th encrypted block, we have that

$$c_i = E(k, m_i) \qquad m_i = D(k, c_i)$$

CBC: in CBC, the data M is divided in blocks $\{m_i\}$, a random string initialization vector IV is chosen and each block is encrypted with E and a key k. The IV is placed at the beginning of the ciphertext. If we denote with c_i the i-th encrypted block, we have that

$$c_0 = E(k, IV \oplus m_0) \quad c_i = E(k, c_{i-1} \oplus m_i) \qquad m_0 = D(k, m_0) \oplus IV \quad m_i = D(k, c_i) \oplus c_{i-1}$$

3. **Definition:**
Let λ be the security parameter. Let $(\text{Gen}, \text{Enc}, \text{Dec})$ be the algorithm for the (textbook) RSA public key encryption scheme defined as

- $\text{Gen}(\lambda)$:
  - Choose at random two distinct primes p, q with λ bits and compute $N = pq$ and $\phi(N) = (p-1)(q-1)$
  - Choose at random a number $e \xleftarrow{R} \mathbb{Z}_{\phi(N)}$ coprime with $\phi(n)$ and compute d such that $d \equiv e^{-1} (\mod \phi(N))$
  - The secret key is $\text{sk} = (N, d)$ and the public key is $\text{pk} = (N, e)$
- $\text{Enc}(\text{pk}, m)$: Compute $c \equiv m^e (\mod N)$
- $\text{Dec}(\text{sk}, c)$: Compute $m \equiv c^d (\mod N)$

The following two assumption are considered computationally secure:

- **Assumption Factorization**:
Given a composite integer N, the factoring problem is to find two positive integers p, q such that $pq = N$.
- **Assumption Discrete Logarithm for RSA in chosen plaintext attack**:
Given N, the encryption c of a message m. The discrete logarithm is to find d such that $c^d \equiv m (\mod N)$.

The discrete logarithm assumption is considered computationally hard since it is not known if exists an efficient algorithm to compute the discrete logarithm.
A different computational hard assumption is the integer factorization.
The RSA encryption scheme achieve security over these two assumptions.

4. **Definition:** An hash function $H : \{0,1\}^* \to \{0,1\}^n$ is a **secure hash function** (or collision resistant) if for all probabilistic polynomial-time adversaries $\mathcal{A}$ defined as

- $\mathcal{A}$ knows H
- $\mathcal{A}$ outputs x, x'
- $\mathcal{A}$ find a collision if and only if $x \neq x'$ and $H(x) = H(x')$

it holds that

$$\Pr(\mathcal{A} \text{ find a collision for } H) < \epsilon \qquad \epsilon \text{ negligible}$$

Hash functions are used for authentication and data integrity and these functions are used in different ways to achieve, for example:

- Data integrity. Adding the result of the hash function at the end of a message so that an attacker cannot change the content of the message.

- Authentication. Hash functions are used in public key infrastructure (PKI) with public key encryption scheme, to achieve the authentication of a user using certificates.
- Secure communication established. In TLS/SSL, hash functions are used to ensure that all the steps of the communication are not tampered.
- Password managing. Hash function are commonly used in password managing and checking.
- Pseudo Random Function. Since for secure hash functions is hard to find a collision, hash functions are sometimes used as PRF.

Medium

5.
   - **Unconditional security**. The cryptosystem cannot be broken even by an Adversary with unlimited computational power.
   - **Computational security**. The best known attack to the cryptosystem is unfeasible in practice.
   - **Provable security**. An Attacker that is able to break the cryptosystem, could be used also to solve some well-known difficult problem (thus to break some hardness assumption).

   The same definitions can be seen in this way:

   - **Unconditional security**. A mathematical proof can prove that it is unfeasible to break the cryptosystem. In this class we can find Secret Sharing schemes that are unconditionally secure if the adversary has less keys that the threshold of the scheme. In this case, the security is based only on the mathematical impossibility to reconstruct the right polynomial.
   - **Computational security**. The cryptosystem has a **known** attack but executing the attack is computationally expensive (economically, for example) and it may take a long time (several thousands of years). In this class we can find brute force attacks that tries **all** the possible keys and, generally, are computationally unfeasible.
   - **Provable security**. The cryptosystem is based on mathematical problems. The problems are assumed as *hard problem* since it is **not known** any efficient attack on them. In this class we can found cryptosystem that base their security on the factorization problem, the discrete logarithm problem or the hash-function collision finding.

6. **Proof**: ECB is not semantically secure.
   Let $m_0 = m\|m$ and $m_1 = m\|\hat{m}$ with $\text{len}(m) =$ len of the block. The challenger will reply with $c = c_1\|c_2$ as

   - if $b = 0$, then $c = c\|c = \text{Enc}(k, m)\|\text{Enc}(k, m)$
   - if $b = 1$, then $c = c\|\hat{c} = \text{Enc}(k, m)\|\text{Enc}(k, \hat{m})$

   If $c_1 = c_2$, then the adversary will output $b' = 0$. If $c_1 \neq c_2$, then the adversary will output $b' = 1$.

   $$\Pr(b' = b) = 1$$

   Proof: CBC is semantic secure if the block cipher is semantically secure.

   Let $m_0 = m\|m$ and $m_1 = m\|\hat{m}$ with $\text{len}(m) =$ len of the block. The challenger will reply with $c = c_1\|c_2$ as

   - if $b = 0$, then $c = c\|c = \text{Enc}(k, IV \oplus m)\|\text{Enc}(k, \text{Enc}(k, IV \oplus m) \oplus m)$
   - if $b = 1$, then $c = c\|\hat{c} = \text{Enc}(k, IV \oplus m)\|\text{Enc}(k, \text{Enc}(k, IV \oplus m) \oplus \hat{m})$

   Since the first block is the same, the adversary can define $r = \text{Enc}(k, IV \oplus m)$ and obtain

   - if $b = 0$, then $\text{Enc}(k, r \oplus m)$
   - if $b = 1$, then $\text{Enc}(k, r \oplus \hat{m})$

Since the adversary knows r, m and $\hat{m}$, the adversary has to break the semantic security of the block cipher (Enc, Dec).
But the blockcipher is semantically secure and so CBC is semantically secure if the block cipher is semantically secure.

7. The DH key exchange protocol is defined with the following protocol between A and B:

   - A generate a description of a cyclic group $G = \langle g \rangle$ of order q and B accepts the public parameters.
   - A chooses at random a value $a \in \{1, \cdots, q-1\}$ and compute $A = g^a$ in G and sends it to B
   - B chooses at random a value $b \in \{1, \cdots, q-1\}$ and compute $B = g^b$ in G and sends it to A
   - A computes the common shared secret key $B^a = \text{sk} = g^{ab}$
   - B computes the common shared secret key $A^b = \text{sk} = g^{ba}$

   The main difference is that DH is a key-exchange protocol and ElGamal is a encryption scheme. This means that they have different application.
   On the other hand, they work on a cyclic group with a generator and their security is connected to the Discrete Logarithm problem.

8. **Proposition:**[1] Given a year with N days, the generalized birthday problem asks for the minimal number n such that, in a set of n randomly chosen people, the probability of a birthday coincidence is at least 50% (assuming that birthdays are independent random variables with same distribution $\Pr(X = \text{ day }) = \frac{1}{N}$.
   In other words, n is the minimal integer such that

   $$1 - \prod_{i=1}^{n-1} \left(\frac{N-i}{N} \right) \geq \frac{1}{2}$$

   and

   $$n \simeq 1.177\sqrt{N}$$

   The birthday problem solutions are really small with respect to the N (this is the paradox part!). The main goal of the paradox is trying to find in a set of N people, any possible pair of them with the same birthday. The number of possible pair is pretty *big* (this is what tricks people) and so if we randomly take n pairs, the probability that they have the same birthday grows *slower* than the dimension N.
   The birthday paradox relates to hash function since a hash function is considered to be probabilistically equivalent to the uniform distribution on a fixed dimension set, i.e., the digests as the output strings of the hash function.
   Since we have the uniform distribution and a fixed number of possible digests N, we can use the birthday paradox to find n as the minimum number of different messages such that the probability of obtaining a collision is greater than $\frac{1}{2}$

9. The verification is a straightforward calculation:

   $$r = g^s X^{-e} = g^{y+xe}(g^x)^{-e} = g^{y+xe-xe} = g^y = Y.$$

   Thus a correctly produced signature will verify, since $h(m||r) = h(m||Y) = e$.

Hard

---

10. Fermat's little theorem states that $K^{p-1} = 1$, so $K \cdot K^{p-2} = 1$. Thus $K^{-1} = K^{p-2}$.

    For RSA decryption, we note that, according to Euler's theorem, $e^{\Phi(\Phi(N))} = 1 \in \mathbb{Z}_{\Phi(N)}$. This could be used to find $d = e^{-1}$, but it becomes necessary to compute $\Phi(\Phi(N))$, which is infeasible for practical N.

[1] Link to a interactive explanation of the birthday problem.

11. • We pick $a_1, a_2 \xleftarrow{R} F$ uniformly at random. Say they become $a_1 = 3$ and $a_2 = 1$ and then we define

$$h(x) = s + a_1 x + a_2 x^2 = 6 + 3x + x^2$$

We have that the shares of $s = 6$ are

$$\begin{aligned} s_1 = h(1) = 6 + 3 + 1 \pmod{11} &= 10 \\ s_2 = h(2) = 6 + 3 * 2 + 4 \pmod{11} &= 5 \\ s_3 = h(3) = 6 + 3 * 3 + 9 \pmod{11} &= 2 \\ s_4 = h(4) = 6 + 3 * 4 + 16 \pmod{11} &= 1 \\ s_5 = h(5) = 6 + 3 * 5 + 25 \pmod{11} &= 2 \end{aligned}$$

• Given the shares s_3, s_4, s_5 it is possible to reconstruct the secret s by computing the Lagrange interpolation.

$$\begin{aligned} \delta_3(x) = \prod_{j=4,5} \frac{x-j}{3-j} = \frac{(x-4)(x-5)}{(3-4)(3-5)} &= \frac{x^2 - 9x + 20}{2} \pmod{11} \\ = (x^2 + 2x + 9)6 \pmod{11} &= 6x^2 + x + 10 \pmod{11} \end{aligned}$$

$$\begin{aligned} \delta_4(x) = \prod_{j=3,5} \frac{x-j}{4-j} = \frac{(x-3)(x-5)}{(4-3)(4-5)} &= \frac{x^2 - 8x + 15}{-1} \pmod{11} \\ = (x^2 + 3x + 4)10 \pmod{11} &= 10x^2 + 8x + 7 \pmod{11} \end{aligned}$$

$$\begin{aligned} \delta_5(x) = \prod_{j=3,4} \frac{x-j}{5-j} = \frac{(x-3)(x-4)}{(5-3)(5-4)} &= \frac{x^2 - 7x + 12}{2} \pmod{11} \\ = (x^2 + 4x + 1)6 \pmod{11} &= 6x^2 + 2x + 6 \pmod{11} \end{aligned}$$

We let

$$h(x) = s_3\delta_3(x) + s_4\delta_4(x) + s_5\delta_5(x) \pmod{11}$$

and since we have $s_3 = 2, s_4 = 1$ and $s_5 = 2$, it holds:

$$s = h(0) = s_3 \cdot 10 + s_4 \cdot 7 + s_5 \cdot 6 \pmod{11} = 2 \cdot 10 + 1 \cdot 7 + 2 \cdot 6 \pmod{11} = 6 \pmod{11}$$

12. • The Mignotte's scheme we are considering has $m_1 = 5, m_2 = 6, m_3 = 7, m_4 = 11, m_5 = 13$. Fixing a threshold $t \in \{1, \cdots, 4\}$, it has to hold

$$m_{5-t+1} \cdots m_t < m_1 \cdots m_{t+1} \qquad \gcd(m_i, m_j) = 1 \text{ for all } i, j \in \{1, \cdots, 5\} \text{ with } i \neq j$$

We can easily see that $\gcd(m_i, m_j) = 1$ for all the possible choice of $i, j \in \{1, \cdots, 5\}$ with $i \neq j$. We now consider the different values for t:

– $t = 1$, it can be a possible threshold since

$$13 = m_5 < m_1 m_2 = 5 \cdot 6 = 30$$

– $t = 2$, it can be a possible threshold since

$$143 = 11 \cdot 13 = m_4 m_5 < m_1 m_2 m_3 = 5 \cdot 6 \cdot 7 = 210$$

– $t = 3$, it can be a possible threshold since

$$1001 = 7 \cdot 11 \cdot 13 = m_3 m_4 m_5 < m_1 m_2 m_3 m_4 = 5 \cdot 6 \cdot 7 \cdot 11 = 2310$$

– $t = 4$, it can be a possible threshold since

$$6006 = 6 \cdot 7 \cdot 11 \cdot 13 m_2 m_3 m_4 m_5 < m_1 m_2 m_3 m_4 m_5 = 5 \cdot 6 \cdot 7 \cdot 11 \cdot 13 = 30030$$

• If $t = 2$, from the check

$$143 = 11 \cdot 13 = m_4 m_5 < m_1 m_2 m_3 = 5 \cdot 6 \cdot 7 = 210$$

we have that $s \in \{143, \cdots, 210\}$

- We have that

$$\begin{cases} s \equiv 0 \pmod 5 \\ s \equiv 1 \pmod 6 \\ s \equiv 5 \pmod 7 \end{cases}$$

From the Chinese Remainder Theorem, we start studying the last two equations:

$$\begin{cases} s \equiv 1 \pmod 6 \\ s \equiv 5 \pmod 7 \end{cases}$$

From the extended Euclidean algorithm we obtain that $1 = (-1) \cdot 6 + (1) \cdot 7$ and for this reason, we obtain that the partial solution $\hat{s}$ is

$$\hat{s} = 5 \cdot (-1) \cdot 6 + 1 \cdot (1) \cdot 7 \pmod{6 \cdot 7} = -30 + 7 \pmod{42} \equiv 19 \pmod{42}$$

We now consider

$$\begin{cases} s \equiv 0 \pmod 5 \\ s \equiv 19 \pmod{42} \end{cases}$$

From the extended Euclidean algorithm, we have that $1 = (-2) \cdot 42 + (17) \cdot 5$ and for this reason, we obtain that the solution s is

$$s = 19 \cdot (17) \cdot 5 + 0 \cdot (-2) \cdot 42 \pmod{5 \cdot 42} = 1615 \pmod{210} \equiv 145 \pmod{210}$$

and $s \in \{143, \cdots, 210\}$. We found that the secret $s = 145$.

13. (a) Mallory's assumption is that Alice's message is $10x$ for some integer x. Then we have $c = (10m)^e = 10^e m^e$, where the computations are in $\mathbb{Z}_N^*$. Mallory can compute 10^e and invert it using the extended Euclidean algorithm to get 10^{-e}. Finally, he constructs the bid $c \cdot (10)^{-e} \cdot 11^e$, which equals $(11m)^e$, i.e. the encryption of $11m$.

An alternative solution is to do a brute force attack, based on the reasonable assumption that a bid is likely to be "small", say at most one million. If, further, the bid is also a multiple of ten, this gives only 10^5 possibilities and Mallory can encrypt all possible bids until he gets c. Now he knows Alice's bid, so he can add 10% and encrypt this number. This is certainly feasible, but still involves a lot of computation, so this solution is inferior to the one above.

(b) Two main ingredients in padding are randomization (to avoid that the same message encrypted twice gives the same encryption) and redundancy (so that randomly constructed ciphertexts are unlikely to be encryptions of a valid message).

Think

14. Everything!

15. **Proposed answer**: from a stream cipher and a block cipher, we can build everything we have seen during the course and the security of the new *"walls"* can be reduced to the security of these *"bricks"*, in this particular order:

- From a stream cipher, we have access to a PRG and its security is determined by the security of the stream cipher
- From a PRG and a block cipher, we can construct all the mode of operations on block cipher that will be secure with respect the randomness of the PRG or the security of the block cipher
- From a block cipher, we can build secure hash functions.
- From the PRG, we can generate random uniform numbers and the numbers will be *truly* random.
- From the PRG, we can generate random uniform numbers, and so build the public key exchange protocols. In this case, the security for the PK encryption scheme will be based on complexity assumptions.
- From the different secure construction, we can build all the cryptographic protocol needed for TLS/SSL and so communicate in a secure way.

I don't want to give you a formal proof of *why* the security is always connected with the *bricks*, but if you study *how* the different cryptographic primitives works, you will always find that the security is always related to ***"real randomness"*** or ***"this has to look random"***.

16. For this question there is **no** correct answer. The question wants you to reason on the social/moral/human rights that cryptography are able to provide:

    - *Privacy* as the human right to have some secret information
    - *Secure communication* as the right to communicate with someone without being spied
    - *Information storage* as the right to store information that no one can modify
    - *Authentication* as the capacity to correctly and digitally identify a person in a group
    - *Knowledge* as the fact that cryptography is a *bridge* between mathematics, informatics and other science and cryptography is a *tool* for human-right laws, social sciences and many other academic disciplines or real-life scenario

    Any of these reason can be used to help or harm people. It's your choice how to use cryptography.

17. It is possible. This cryptographic primitive is contained in a class of **Cryptographic Obfuscation**. The topic is an open problem in cryptography and it is, nowadays, a theoretical construction described in different papers, such as this.
    If you want, you can read this paper and you can notice that the question given is the abstract of that paper!