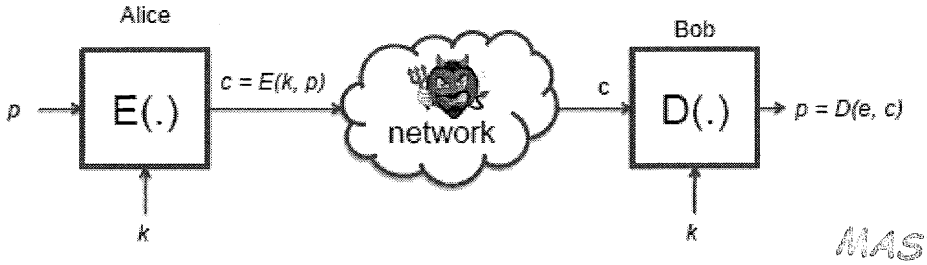
Symmetric Encryption

lunedì 3 luglio 2017 10:35

Blocks of a symmetric cryptosystem

A symmetric cryptosystem can be seen as composed by the following blocks:



MASTER COPY
Tel. 050 8312126
Cell. 388 9837745

Where:

- **p**: plaintext (the information to transmit)
- **k**: key, shared secret between Bob and Alice
- **c**: cyphertext (what travels on the channel)
- **E**: encryption function, starting from the plaintext and the key, produces the cyphertext, as $c = E(p, k)$
- **D**: decryption function, starting from the cyphertext and the same key as before, produces back the original plaintext as $p = D(c, k)$

The encryption and decryption function are implemented in software by libraries or directly in hardware if strict performances are required. The basic idea is that even if someone is able to obtain the encrypted message, will be "difficult" to obtain the initial plaintext.

A good cryptosystem is secure even if everything about the system, except the key, is public knowledge. This means that even if the enemy has knowledge about the system, without the key, it is difficult to obtain the initial plaintext.
This kind of system brings the advantages that only the key must be shared and secret, and the latter is a small secret to maintain, instead of keeping secrecy of the entire system and its functioning.
History tells that hiding the entire system (Security through Obscurity) is not so good (GSM, RC4, Enigma...)

Cypher Definition

A cipher defined over a set of keys, plaintexts and cyphertexts (K, P, C) is a pair of efficient algoritms where:

$$E: P\ x\ K \rightarrow C$$
$$D: C\ x\ K \rightarrow P$$

The encryption algorithm is a function from the domain obtained by the cartesian product of key domain and plaintext domain with values into cyphertext domain. This can be seen as: given a couple plaintext-key (p, k), with $p \in P, k \in K$ the algorithm produces a value into cypherext domain $c \in C$.
Given the definition, a cypher is a pair of efficient algos, such that (consistency equation):

$$\forall\ p \in P\ , k \in K: D\left(k, E(p, k)\right) = p$$

That means that this pair of algos, for each plaintext and for each key, must be able to retrieve the original plaintext message encrypted with k, starting from the cyphertext and k itself.
Also E can be randomized (if the key is random for instance), but defined E, D must always be deterministic, in order to obtain back the plaintext.

Security of a cypher

A symmetric cypher is secure, given a pair of plaintext and cyphertext (p, c) iff:

- Given c, it is "difficult" to determine p without key and vice versa (break the algorithm)
- Given both c and p it is difficult to obtain k (resistance against plaintext attacks)

With "difficult" we would mean impossible, but the latter is not practicable cause means also impossible to implement. So difficult means so hard that the adversary is not able the obtain p or k, because it will take too much time.

MASTER COPY
Tel. 050 8312126
Cell. 388 9837745

Perfect Cypher (Shannon)

A cypher (E,D) defined over $(\mathcal{P}, \mathcal{K}, \mathcal{C})$ has perfect secrecy iff:

$$\forall\, p \in P\,, c \in C\colon \Pr(P = p | C = c) = \Pr(P = p)$$

Where: P is a random vairbale in $\mathcal{P}$ and C is a random vaibale in $\mathcal{C}$

This means that for each plaintext and cyphertext, the a posteriori probability of having chosen a certain message p to transmit, given that a certain cyphertext corresponding to it has been generated, is equal to the probability of transmitting the initial message p. This can be seen also as: the fact that the cyphertext is exactly c, does not hold any information about the plaintext which it hides.

Shannon's Theorem

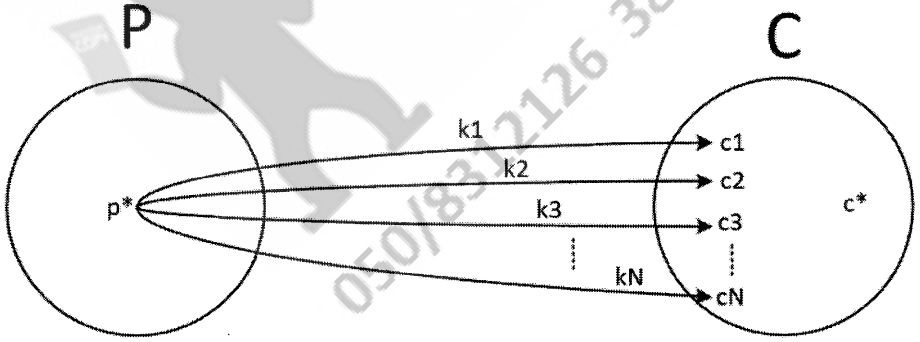
Unfortunately, a perfect cypher is not practicable, because a cypher is perfect if the number of possible keys is larger than the number of possible messages: $|\mathcal{K}| \geq |\mathcal{P}|$

Proof by contradiction

1) #keys < #plaintexts $\rightarrow |\mathcal{P}| \geq |\mathcal{K}|$
2) # cyphertexts > #plaintexts this must be true otherwise the encryption could not be invertible, cause two different cyphertexts may be function of the same plaintext, hence $|\mathcal{C}| \geq |\mathcal{P}|$

Hence $|\mathcal{C}| \geq |\mathcal{K}|$.

Taken a plaintext p^* with a not null probabilty of being transmitted, that is $\Pr(P = p^*) \neq 0$, encypt the message with all the possible keys k1, k2, k3, ..., obtaining c1, c2, c3, ...



Since the number of cyphertexts is larger than the number of keys, there will be a certain cyphertext c^* which cannot be obtained starting from p^* with none of the keys. So:

$$\exists c^* \in C \mid \Pr(P = p^* | C = c^*) = 0$$

In this sense, there must be a couple (p^*, c^*) which does not obey to Shannon's perfect cypher definition if the number of messeages is larger than the number of keys.

This is exactly what happens in practice because we usually have long messages which may contains lots of different characters and so the number of total possible messages is larger than the number of possible keys.

Unconditional Security

Other requirements for perfect secrecy are:

- An adversary is assumed to have an infinite computing resources

- Observation of the cyphertext does not provide any information about plaintext
- Key is truly randomly generated
- The length of the key is at least equal to the one of the message.

The observation of the cyphertext must provide a null information about the plaintext and this brings to another definition of perfect cypher:

"Given two messages m1,m2 of same size, encrypted with the same key k truly randomly generated, the probability that the cyphertext observed on the channel is generated from message m1, is equal to the probability of being generated from message m2".

$$\forall m_1, m_2 \in P : |m_1| = |m_2|, k \in K : k \leftarrow random()$$
$$\Pr\left(c = E(m_1, k)\right) = \Pr(c = E(m_2, k))$$

If only two messages compose the set P, then those two probabilities are both 0.5, which means that I cannot obtain any kind of information about the plaintext only looking at the cyphertext on the channel.

One Time Pad - OTP

Given a message t bits long, $m \in (0,1)^t$ and a key perfectly randomly generated and of the same length,
$k \in (0,1)^t \leftarrow random()$, the cyphertext will be $c \in (0,1)^t$ and

- E(): $c_i = m_i \oplus k_i \ \forall i \in [1,t]$
- D(): $m_i = c_i \oplus k_i \ \forall i \in [1,t]$

We immediately see that the encryption/decryption function is very fast to execute, cause it is a simple xor operation, but at the same that the key must long as the message and the longer the message, the longer the key. It is very hard to generate perfect random keys of long bit streams, hence this crypto algorithm was often used in military communications, where messages were short and strongly formatted.

MASTER COPY
Tel. 050 8312126
Cell. 388 9837745

OTP as perfect cypher

OTP is a perfect cypher if:

1) All the messages have the same length
   a. $\forall m, m' \in P : |m| = |m'|$
2) Each messages has a not null probability of being transmitted
   a. $\forall m \in P : \Pr(P = m) \neq 0$
   b. This is a strong assumption, cause if we are transmitting text, not all combination of bits can be translated into ascii code and so cannot correspond to a letter.
3) The key must be perfectly randomly generated
   a. $k \in K \leftarrow random(\square)$

How it works: supposing that message is composed by letters, encryption can be implemented by simply:

$$c[i] = m[i] + k[i] \ mod \ 26$$

Translate each letter into a number, for the message and for the key, sum up those two numbers and then performs mod 26 in order to ensure that the result will be a letter again.

m =	S	U	P	P	O	R	T	J	A	M	E	S	B	O	N	D
k =	W	C	L	N	B	T	D	E	F	J	A	Z	G	U	I	R
c =	O	W	A	C	P	K	W	N	F	V	E	R	H	I	V	U

Example:

m[0] = S = 18
k[0] = W = 22
c[0] = (18+22) mod 26 = 40 mod 26 = 14 = O

Only one key is able to extract the exact initial message from the plaintext.
Thus the adversary, even if he tries all the possible keys, he will have a set of decrypted plaintexts: some of them will not have any kind of sense. Among the remaining results, the adversary will never be able to understand which of them is the correct one.
Supposing to try all the possible keys, assume that two of them, k, k', will give back an undestrandable result, but which is the right to choose between those two? The adversary will never know.

k

m =	S	U	P	P	O	R	T	J	A	M	E	S	B	O	N	D
k =	W	C	L	N	B	T	D	E	F	J	A	Z	G	U	I	R
c =	O	W	A	C	P	K	W	N	F	V	E	R	H	I	V	U

k'

c =	O	W	A	C	P	K	W	N	F	V	E	R	H	I	V	U
k' =	M	W	L	J	V	T	S	E	F	J	A	Z	G	U	I	R
m =	C	A	P	T	U	R	E	J	A	M	E	S	B	O	N	D

OTP is a perfect cypher
Considering the assumptions:

- All the messages have the same size t bits.
- All the messages have not null probability of being transmitted
- Keys randomly generated and of the same size of the message (t bits)

According to Shannon, we want to prove that in OTP

$$Pr(P = p|C = c) = \Pr(P = p)$$

Applying Bayes:

$$\Pr(P = p|C = c) = \frac{\Pr(P = p, C = c)}{\Pr(C = c)}$$

Lemma: $\forall p \in P, c \in C \quad \exists! k : c = p \oplus k \quad \rightarrow \quad k = p \oplus c$

$$\Pr(C = c) = \frac{\#\,(p,k) : c = p \oplus k}{\#\,all\ possible\ (p,k)}$$

Numerator: Since only one pair p and k will involve into a certain cyphertext c, that is only by m_1 and k_1 it is possible to obtain c_1, we want to obtain all the couples p,k such that their xor is equal to a certain cyphertext c, for all the possible cyphertexts. Hence:

If I take m_1, only one key will produce c_1
If I take m_2, only one key will produce c_2
If I take m_3, only one key will produce c_3
...

So, for all possible cyphertexts $c_1, c_2, c_3, \ldots \#(p,k) : c = p \oplus k = |P|$

$$\Pr(C = c) = \frac{|P|}{|P|x|K|} = \frac{1}{|k|} = \frac{1}{2^t}$$

Introduce a random variable K which maps all the possible keys. Since one only key can map p into c, the probability of selecting p ($P = p$) AND transmitting c ($C = c$), can be written as:

$$\Pr(P = p, C = c) = \Pr(P = p, C = c, K = k)$$

The new condition added does not alter the probability, because $C = c$ only if p is selected ($P = p$) and k is selected ($K = k$).
Since only p and k can generate c, the condition $C = c$ is now redudant, casue for sure, if $P = p$ and

MASTER COPY
Tel. 050 8312126
Cell. 388 9837745

$K = k$ will be $C = c$, hence:

$$\Pr(P = p, C = c, K = k) = \Pr(P = p, K = k)$$

Since the selection of a message is independent by the selection of a key, we have:

$$\Pr(P = p, K = k) = \Pr(P = p) \cdot \Pr(\mathrm{K} = \mathrm{k}) = \Pr(\mathrm{P} = \mathrm{p}) \cdot \frac{1}{2^{\mathrm{t}}}$$

So finally:

$$\Pr(P = p | C = c) = \frac{\Pr(P = p) \cdot \frac{1}{2^t}}{\frac{1}{2^t}} = \Pr(P = p)$$

MASTER COPY
Tel. 050 8312126
cell. 388 9837745

Under those assumptions, OTP is a perfect cypher according to Shannon.

OTP Pros-Cons

PRO:

- Perfect, according to Shannon
- Efficient, due to poor computational requirements introduced by the xor operation

CONS:

- Long random keys generation
- One time pad: a key must be used just once
  - Suppose to have $c_1 = m_1 \oplus k$ and $c_2 = m_2 \oplus k$. An adversary may obtain c_1 and c_2 and then perform the xor: $c_1 \oplus c_2 = m_1 \oplus k \oplus m_2 \oplus k = m_1 \oplus m2$. Even if the result is not a plaintext but a xor between two of them, it just gives an additional information about single plaintext. In fact, if $m_1[i] \oplus m_2[i] = 0$, this mens that $m_1[i] = m_2[i]$. Registering even more cyphertext it is possible to obtain all the equals letter in the text, which is lot of information.
- Known-plaintext attack breaks OTP:
  - We have three kinds of attack:
    1) Cyphertext-only attack: adversary only knows cyphertext
    2) Known-plaintext attack: adversary knows pairs of plaintext and corresponding cyphertext
    3) Chosen-plaintext attack: adversary has control on plaintext
  - Under a known-plaintext attack, if adversary know p and c it can easily obtain $k = p \oplus c$
- Malleability: an adversary is able to work on cyphertext in order to change the plaintext:
  - Alice transmit $c = p \oplus k$
  - Adversary obtain catches the cypher and perturbates it: $c' = c \oplus \gamma$
  - Bob:
    - Receives c' instead of c
    - Decrypts c' instead of c
    - Obtains p' instaed of p
    - $p' = c' \oplus k = (c \oplus \gamma) \oplus k = \Big(\big((p \oplus k) \oplus \gamma\big) \oplus k\Big) = p \oplus \gamma$