

- se il CS è a segretezza perfetta,  $\mathbf{Pr}(x|y) = \mathbf{Pr}(x)$
- dal teorema di Bayes,  
 $\mathbf{Pr}(x|y) = \mathbf{Pr}(x) \iff \mathbf{Pr}(y|x) = \mathbf{Pr}(y)$
- $\forall y, \mathbf{Pr}(y|x) = \mathbf{Pr}(y) > 0$
- questo dice che, fissato  $x$ ,  $\forall y \in \mathcal{C}$  c'è una chiave  $k \in \mathcal{K}$  tale che  $e_k(x) = y$
- il numero di chiavi è maggiore o uguale al numero dei CT

$$|\mathcal{K}| \geq |\mathcal{C}| \geq |\mathcal{P}|$$

- Teorema di Shannon
- nel caso in cui  $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{P}|$ , un crittosistema è a segretezza perfetta se e solo se
  - ①  $\forall x \in \mathcal{P} \forall y \in \mathcal{C}, \exists! k \in \mathcal{K}$  tale che  $e_k(x) = y$
  - ② le chiavi sono scelte con probabilità uniforme (pari a  $\frac{1}{|\mathcal{K}|}$ )

## one-time pad

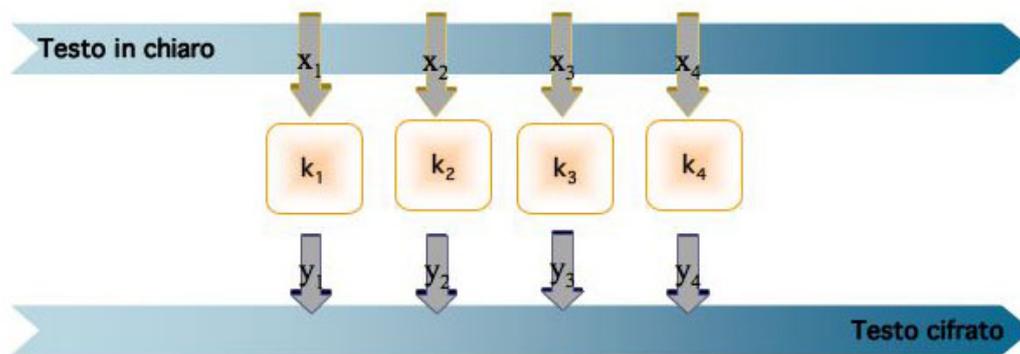
- L'one-time pad o cifrario di Vernam (1917) è il crittosistema tale che
- $\mathcal{P} = \mathcal{C} = \mathcal{K} = (\mathbb{Z}_2)^m$
- se  $k = (k_1, k_2, \dots, k_m)$  si ha
  - $e_k(x_1, x_2, \dots, x_m) = (x_1 + k_1, x_2 + k_2, \dots, x_m + k_m)$
  - $d_k(y_1, y_2, \dots, y_m) = (y_1 - k_1, y_2 - k_2, \dots, y_m - k_m) = (y_1 + k_1, y_2 + k_2, \dots, y_m + k_m)$   
(l'operazione è lo XOR bit a bit)
- è un crittosistema a segretezza perfetta

## one-time pad

- Vantaggi:
  - Cifratura e decifratura molto semplici da calcolare.
  - A segretezza perfetta.
- Svantaggi:
  - La chiave è lunga quanto il testo in chiaro.
  - La chiave può essere usata una sola volta.
  - Si ha il problema di distribuire e conservare in modo sicuro le chiavi.

## one-time pad

nell' **one-time pad**, il testo in chiaro  $x = x_1x_2\dots$  viene cifrato bit per bit con la chiave  $k = k_1, k_2, \dots$  per ottenere il testo cifrato:  
 $x_i \oplus k_i = y_i$



## cifrario a flusso: definizione

### Definizione

Un **cifrario a flusso sincrono** è una sestupla  $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{L}, \mathcal{E}, \mathcal{D})$ , e una funzione  $g$  tali che

- 1  $\mathcal{P}$  è un insieme finito di testi in chiaro (plaintext)
- 2  $\mathcal{C}$  è un insieme finito di testi cifrati (ciphertext)
- 3  $\mathcal{K}$  è un insieme finito di chiavi. ( $\mathcal{K}$  è detto spazio delle chiavi)
- 4  $\mathcal{L}$  è un insieme finito detto alfabeto del flusso di chiave (keystream alphabet)
- 5  $g$  è il generatore di keystream: una funzione che a ogni chiave  $k \in \mathcal{K}$  associa una stringa infinita  $z_1z_2\dots$ , detta keystream, con  $z_i \in \mathcal{L} \ \forall i \geq 1$
- 6 per ogni  $z \in \mathcal{L}$  c'è una funzione di cifratura  $e_z \in \mathcal{E}$ ,  $e_z : \mathcal{P} \rightarrow \mathcal{C}$  e una funzione di decifratura  $d_z \in \mathcal{D}$ ,  $d_z : \mathcal{C} \rightarrow \mathcal{P}$  tali che per ogni  $x \in \mathcal{P}$  si ha  $d_z(e_z(x)) = x$

- esempio: il cifrario di Vigenère può essere pensato come un cifrario a flusso:
- $\mathcal{P} = \mathcal{C} = \mathcal{L} = \mathbb{Z}_{26}, \mathcal{K} = \mathbb{Z}_{26}^m$
- $e_z(x) = x + z$  e  $d_z(y) = y - z \pmod{26}$
- data  $(k_1, k_2, \dots, k_m) \in \mathcal{K}$ , per generare il flusso di chiave si usa la regola

$$z_i = \begin{cases} k_i & \text{se } i \leq m \\ z_{i-m} & \text{se } i > m \end{cases}$$

- si ottiene  $k_1 k_2 \dots k_m k_1 k_2 \dots k_m k_1 k_2 \dots$  come keystream

## caso binario

- molto spesso, avremo  $\mathcal{P} = \mathcal{C} = \mathcal{L} = \mathbb{Z}_2$
- con  $e_z(x) = x + z \pmod{2}$ ,  $d_z(y) = y + z \pmod{2}$
- $e_z(x) = x \oplus z$ ,  $d_z(y) = y \oplus z$
- un cifrario a flusso “imita” l’one-time pad
- la sicurezza sta nella funzione  $g$
- $g$  è un generatore di stringhe pseudocasuali (PRG)
- esempi di cifrari a flusso recenti
  - RC4 (nel SSL e WEP)
  - A5 (GSM)
  - progetto eSTREAM

## PRG

- la funzione  $g$  è un generatore di stringhe pseudocasuali (PRG)
- nella definizione “ideale”,  $g$  produce una stringa binaria infinita
- in pratica  $g : \mathbb{Z}_2^s \rightarrow \mathbb{Z}_2^n$  con  $n \gg s$
- la stringa binaria di lunghezza  $s$  in input si dice chiave o **seme**
- l'output deve “somigliare” a una stringa binaria casuale
- questa somiglianza è misurata con test statistici
- inoltre l'output non deve essere prevedibile
- non possiamo avere segretezza perfetta: la chiave è troppo corta
- serve un'altra definizione di sicurezza

## ricorrenze lineari

- generare il flusso di chiave usando una ricorrenza lineare di grado  $m$
- $\mathcal{P} = \mathcal{C} = \mathcal{L} = \mathbb{Z}_2, \mathcal{K} = \mathbb{Z}_2^{2m}$
- si parte da una  $m$ -pla binaria  $(k_1, k_2, \dots, k_m)$ , e si pone  $z_1 = k_1, \dots, z_m = k_m$
- per  $i > m$ ,  
$$z_{i+m} = \sum_{j=0}^{m-1} c_j z_{i+j} = c_0 z_i + c_1 z_{i+1} + \dots + c_{m-1} z_{i+m-1},$$
 $c_j \in \mathbb{Z}_2$ , tutto modulo 2
- la chiave è la  $2m$ -pla  $(k_1, \dots, k_m, c_0, \dots, c_{m-1})$
- se si parte dalla stringa 000... si ha in output 00000000..., quindi bisogna partire da una stringa non nulla
- la stringa di partenza viene chiamata **seme**

- esempio: sia  $m = 4$ , consideriamo la ricorrenza

$$z_{i+4} = z_{i+1} + z_i$$

- se partiamo dalla stringa 0001, la ricorrenza dà
- 0001001101011110001...
- partendo da un'altra stringa non nulla si ottiene una permutazione ciclica dell'output: per esempio 1011 dà 1011110001001101011
- l'output è periodico di periodo  $15 = 2^4 - 1$
- questo è il massimo periodo che può avere una stringa prodotta da una ricorrenza lineare di grado 4: ci sono 15 stringhe binarie non nulle di lunghezza 4