one-time pad

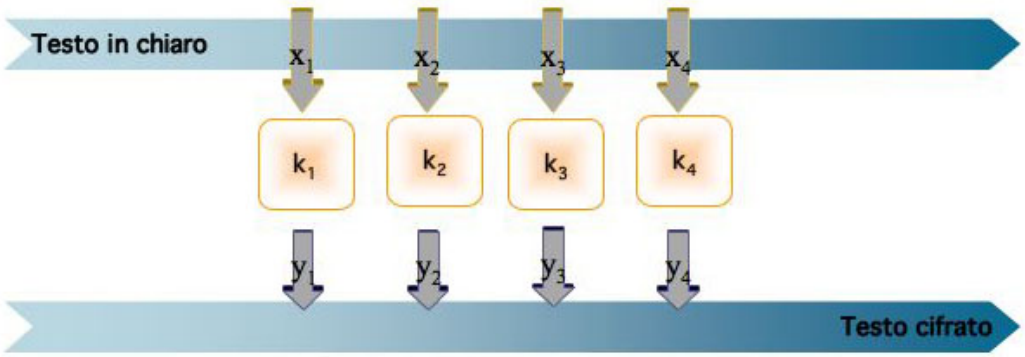
- L'one-time pad o cifrario di Vernam (1917) è il crittosistema tale che
- $\mathcal{P} = \mathcal{C} = \mathcal{K} = (\mathbb{Z}_2)^m$
- se $k = (k_1, k_2, \ldots, k_m)$ si ha
  - $e_k(x_1, x_2, \ldots, x_m) = (x_1 + k_1, x_2 + k_2, \ldots, x_m + k_m)$
  - $d_k(y_1, y_2, \ldots, y_m) = (y_1 - k_1, y_2 - k_2, \ldots, y_m - k_m) = (y_1 + k_1, y_2 + k_2, \ldots, y_m + k_m)$
    (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 $x = x_1 x_2 \ldots$ viene cifrato bit per bit con la chiave $k = k_1, k_2, \ldots$ 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 $z_1 z_2 \ldots$, 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 $e_z \in \mathcal{E}$, $e_z : \mathcal{P} \to \mathcal{C}$ e una funzione di decifratura $d_z \in \mathcal{D}$, $d_z : \mathcal{C} \to \mathcal{P}$ 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 $(k_1, k_2 \ldots, k_m) \in \mathcal{K}$, 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 $k_1 k_2 \ldots k_m k_1 k_2 \ldots k_m k_1 k_2 \ldots$ 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 $g : \mathbb{Z}_2^s \to \mathbb{Z}_2^n$ con $n >> s$
- 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

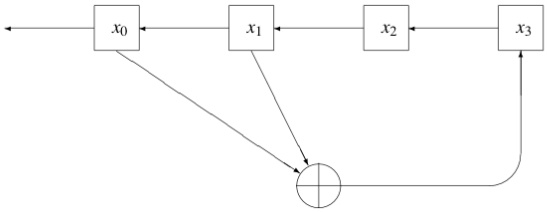
- generare il flusso di chiave usando una ricorrenza lineare di grado m
- $\mathcal{P} = \mathcal{C} = \mathcal{L} = \mathbb{Z}_2$, $\mathcal{K} = \mathbb{Z}_2^{2m}$
- si parte da una m-pla binaria $(k_1, k_2 \ldots, k_m)$, e si pone $z_1 = k_1, \ldots, z_m = k_m$
- per $i > m$, $z_{i+m} = \sum_{j=0}^{m-1}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

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

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

- se partiamo dalla stringa 0001, la ricorrenza dà
- 000100110101111000 1 . . .
- 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

registri a scorrimento lineare



- questa situazione è realizzata da un registro a scorrimento lineare a reatroazione (linear feedback shift register)
- alcune posizioni (nell'esempio quelle di posto 0 e 1) sono collegate a un addizionatore (che esegue uno XOR)
- a ogni "istante"
  - tutta la stringa viene spostata a sinistra di una posizione dando un bit in output
  - il bit che proviene dall'addizionatore viene inserito a destra (feedback)

- in generale, l'output di un registro a scorrimento a m bit ha periodo massimo $2^m - 1$ (perchè ci sono $2^m - 1$ stringhe binarie non nulle di lunghezza m)
- un registro a scorrimento di grado m con periodo $2^m - 1$ si dice primitivo
- si può dimostrare che per ogni $m \geq 1$ esiste una ricorrenza lineare il cui periodo è $2^m - 1$
- da una chiave lunga 100 posso ottenere una stringa binaria lunga $2^{50} - 1$ da usare come keystream
- la stringa ottenuta somiglia a una stringa casuale (questo non vuol dire molto)

crittoanalisi di RSL

- ma i cifrari basati sui RSL non vanno bene dal punto di vista crittografico
- basta conoscere $2m$ coppie (testo in chiaro, corrispondente testo cifrato) per ricostruire completamente la ricorrenza
- noto il plaintext $x_1, x_2, \ldots, x_n$ e il corrispondente ciphertext $y_1, y_2, \ldots, y_n$, si ottiene $z_1, z_2, \ldots z_n$ (basta sommare mod 2)
- ora basta scoprire i coefficienti $c_0, c_1, \ldots, c_{m-1}$
- se $n \geq 2m$, dalla ricorrenza si ottiene un sistema lineare di m equazioni nelle incognite $c_0, c_1, \ldots, c_{m-1}$
- si può dimostrare che questo sistema è sempre risolubile

esempio

- sia $m = 4$, $x = 10111011$ e $y = 01101101$
- allora $z = 11010110$
- i primi 5 bit danno $0 = c_3 + c_1 + c_0$
- i bit da 2 a 6 danno $1 = c_2 + c_0$
- $1 = c_3 + c_1$
- $0 = c_3 + c_2 + c_0$
- la soluzione è $c_0 = 1, c_1 = c_2 = 0, c_3 = 1$, e la ricorrenza è

$$z_{i+4} = z_{i+3} + z_i$$

cifrari a flusso non sincroni

- funziona tutto come in un cifrario a flusso
- ma ogni elemento del flusso di chiave z_i
- dipende anche dai precedenti plaintext $x_1 \ldots x_{i-1}$ e/o dai precedenti ciphertext $y_1 \ldots y_{i-1}$
- Esempio: cifrario autoclave