-> **Cifratura** -> trasformazione di un testo in un altro, che può essere fatta per **blocchi** o su **stream di dati**.

- Blocco, funzione $f(T) = T'$

Decifratura -> operazione inversa. Il mittente cifra il messaggio e il destinatario la decifra.

- entrambi conoscono una chiave -> può essere nota la funzione ma senza chiave è inutilizzabile

-> **Cifratura** -> **Simmetrica** -> stessa chiave, che <u>deve</u> rimanere segreta.

- **Asimmetrica**
  - funzione one-way
    - $f(x) \rightarrow y$ (facile)
    - $f'(y) \rightarrow x$ quasi impossibile
  - una chiave per cifrare e decifrare. Ogni utente ha 2 chiavi una chiave pubblica e una privata

-> **Autenticazione**

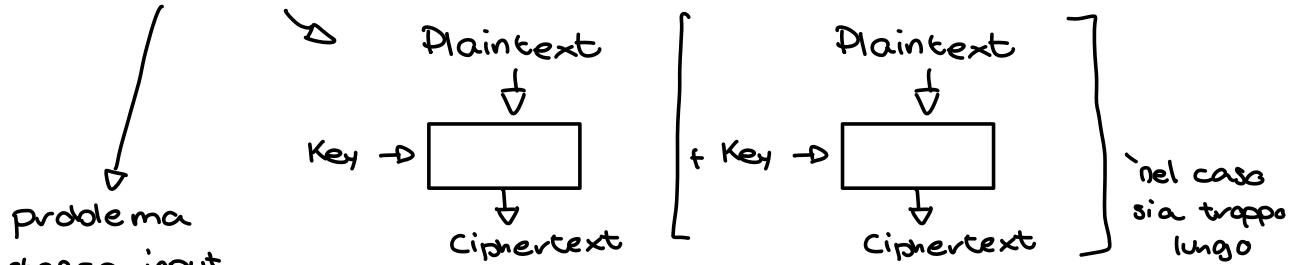
- Il mittente invia il messaggio cifrato con la chiave pubblica del destinatario però prima applica la sua chiave segreta, ottenendo una firma.

Il destinatario, riceve il (pacchetto) e decifra con la propria chiave segreta e ottiene testo e firma quindi decifra la firma con la chiave pubblica del mittente e se è corretto il mittente è verificato.

Signature -> solo una parte del messaggio

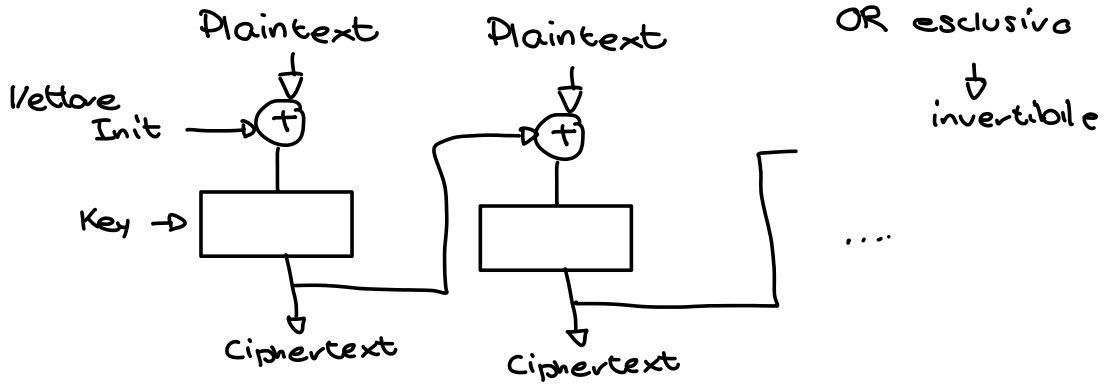
→ Attacco MIM (Man in the Middle)

→ Electronic Codebook (ECB)



problema stesso input da stesso output quindi una ripetizione di un blocco → aiuta l'attaccante

→ Cipher Block Chaining (CBC)



→ Cifratura di Stream

→ ipotiziamo come un blocco piccolo come un byte
→ codominio limitato

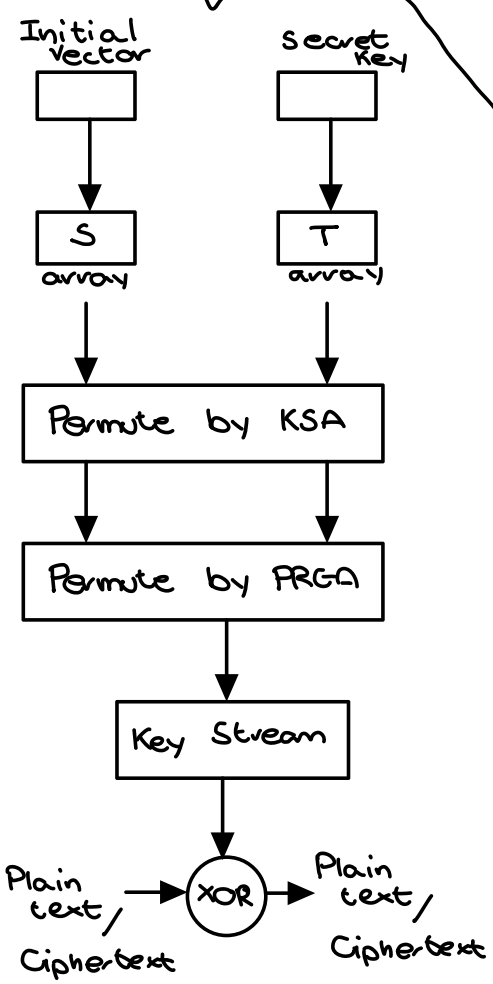
Key stream, stream di bit (pseudo) casuali

XOR

Rivest Chiper 4 (RC4)

→ Algoritmo di generazione di byte (pseudo) casuali.

fasi:
- KSA
- PRGA → modello simile a quello a blocchi
  → chiave segreta + operazioni



Per generare il Key stream viene utilizzato uno stato interno segreto il quale consiste
- S, 256 byte (cambiato ogni volta)
- due puntatori i, j da 8 bit

+ chiave segreta da 40 a 2048 bit

⚠ Chiave più lunga > difficoltà da parte dell'attaccante, > tempo richiesto dagli algoritmi

Key-scheduling algorithm (KSA) → aggiungo "rumore"

→ algoritmo per inizializzare la permutazione dell'array S

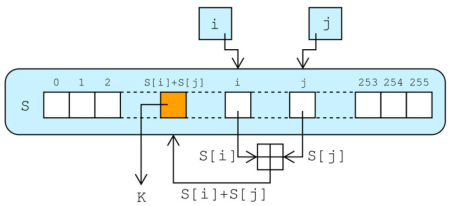
```
for i from 0 to 255
    S[i] := i
endfor
j := 0
for i from 0 to 255
    j := (j + S[i] + key[i mod keylength]) mod 256
    swap values of S[i] and S[j]
endfor
```

→ La tecnica utilizzata: viene utilizzato il valore del puntatore j precedente, insieme a quello della casella S corrente e insieme al risultato con la chiave segreta. e scambio S[i] e S[j]

→ Pseudo-random generation algorithm (PRGA)

↓ generazione di bit pseudo casuali

↳ eseguito per un numero indefinito di volte



↓ messi in XOR con il messaggio

↓ non è la key segreta → ma una nuova chiave

```
i := 0
j := 0
while GeneratingOutput:
    i := (i + 1) mod 256
    j := (j + S[i]) mod 256
    swap values of S[i] and S[j]
    K := S[(S[i] + S[j]) mod 256]
    output K
endwhile
```

K è uno dei valori di S, ovvero un numero che va da 0 a 255 (mod 256)

→ Funzioni Crittografiche

↓ ↳ Da un testo producono una firma di dimensione fissa

↳ digest

- Deterministiche (stesso output con stesso input)
- Veloci
- Computazionalmente intrattabili
- Bassissima probabilità di avere stesso output con input diversi (iniettiva)
- Piccole variazioni sull'input grandi variazioni sull'output

→ MD5 e SHA

Appunti da siti / fonti aggiuntive

- Gli algoritmi a chiave simmetrica, sono algoritmi per la crittografia che utilizzano le stesse chiave per criptare e decriptare

- Le chiavi rappresentano un segreto condiviso tra due o più parti → entrambe le parti sono a conoscenza della chiave segreta

- Tipi → cifrari a flusso
  - → cifrano le cifre (byte) o letteve di un messaggio una alla volta
  - cifrari a blocchi
    - prendono n bits e li crittografano come una singola unità

- La crittografia a chiave pubblica (asimmetrica)
  - → coppia di chiavi, una pubblica e una privata
  - basati su problemi matematici, funzioni unidirezionali