Crittografia

Indice

Capitolo 1

Introduzione

La crittografia fa parte, insieme alla crittoanalisi, della crittologia.
La **crittografia** è la scienza che studia tecniche e metodologie per **cifrare un testo in chiaro**, al fine di produrre un **testo cifrato comprensibile solo ad un ricevente legittimo**, il quale possiede l'informazione sufficiente (detta chiave) per decifrarlo, recuperando il testo in chiaro.
La **crittoanalisi** studia come decrittare un testo cifrato per ottenere il testo in chiaro: il verbo decrittare indica l'azione compiuta da un'entità che non possiede la chiave per recuperare, in modo legittimo, il testo in chiaro. In letteratura, suddetta entità viene indicata con il termine ascoltatore, oppure avversario o anche nemico. Lo scopo della crittografia è di produrre un messaggio cifrato m in modo che nessun avversario sia in grado di decrittarne il contenuto.
La **stegonografia** (dal greco *scrivere nascosto*), indica l'insieme delle tecniche che permettono a due (o più) entità in modo tale da occultare, agli occhi di un ascoltatore, non tanto il contenuto (come nel caso della crittografia), ma l'estistenza stessa di una comunicazione. In altre parole, è l'arte di nascondere un messaggio all'interno di un altro *messaggio contenitore*. Alcuni esempi sono:

- disposizione dei caratteri
- inchiostro invisibile
- contrassegno dei caratteri
- nascondere messaggi all'interno di bit di file multimediali
- ...

Proprietà di sicurezza di un messaggio

1. **Confidenzialità**
2. **Autenticazione**
3. **Non ripudio**

4. **Integrità**
5. **Anonimia** (canali in cui le entità comunicanti devono poter nascondere la loro identità)

Capitolo 2

Crittografia classica

2.1 Informazioni generali

Queste informazioni sono valide per qualsiasi schema crittografico.

Operazioni di trasformazione

In generale, possono essere fatte due operazioni di *trasformazione*:

- **sostituzione:** ciascune elemento del testo viene mappato un altro elemento
- **trasposizione:** gli elementi del testo in chiaro vengono scambiati di posto

$\Rightarrow$ è fondamentale che le operazioni siano invertibili per recuperare le informazioni

Spazio delle chiavi

Ogni schema di cifratura per essere robusto deve avere uno spazio delle chiavi abbastanza grande, altrimenti è vulnerabile ad un attacco di forza bruta; è una condizione necessaria ma non sufficiente.

Sicurezza

- ***unconditionally secure:*** è impossibile decifrare il testo criptato, indipendentemente dal tempo e risorse computazionali
- ***computationally secure:*** il tempo richiesto per decifrare il messaggio è superiore alla vita utile delle informazioni

Sicurezza perfetta

In un cifrario perfetto, osservando il messaggio cifrato l'avversario non ha alcuna possibilità di ottenere informazioni sul messaggio in chiaro; testo in chiaro e cifrato sono indipendenti tra loro; in termini probabilistici si può definire:

$$Prob(M|C) = Prob(M)$$

ovvero che la distribuzione delle probabilità non è influenzata dal fatto di conoscere il cifrato.

Principio di Kerckhoffs

La sicurezza di un crittosistema deve dipendere solo dalla sicurezza della chiave e non dalla segretezza dell'algoritmo usato.

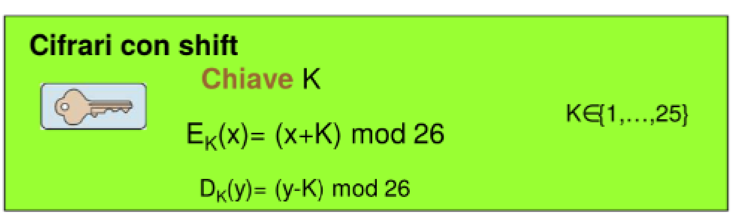
Tipi di attacco

Si può fare una distizione di diversi tipi di attacco in base alla conoscenza dell'avversario:

- ***Known Ciphertext Attack***
  - conosce solo il cifrato
- ***Known Plaintext Attack***
  - conosce anche il testo in chiaro; l'obiettivo è trovare la chiave
- ***Chosen Plaintext Attack***
  - può interagire con il sistema a far cifrare un messaggio a sua scelta
- ***Chosen Ciphertext Attack***
  - può interagire con il sistema e far decifrare un messaggio a sua scelta
- ***Chosen Text Attack***
  - è a conoscenza di coppie chiaro-cifrato

2.2 Cifrari con shift

Viene fatto uno shift delle lettere dell'alfabeto.



Sono possibili solamente 25 chiavi, è vulnerabile ad un attacco di forza bruta.

2.3 Cifrari monoalfabetici

Viene fatta una sostituzione con un alfabeto cifrante. Sono possibili 26! chiavi (tutte le possibili combinazioni per l'alfabeto cifrante), ovvero 2^{88}.

2.3.1 Crittoanalisi

Potrei pensare di essere al sicuro perché abbiamo un ampio spazio delle chiavi (in realtà ad oggi si considera sicuro 2^{128}), ma questo schema si può rompere con una semplice tecnica di crittoanalisi: **analisi delle frequenze**.
L'assunzione è che un simbolo molto frequente nell'alfabeto di origine lo sarà anche in quello cifrato, shiftato di qualche lettera.
Si possono fare dei ragionamenti di crittoanalisi anche sulle caratteristiche della lingua utilizzata per risalire al testo in chiaro (ad esempio, in inglese dopo la t spesso c'è l'$h\ldots$).

2.4 Cifrario Affine

Sono un caso particolare dei cifrari a sostituzione monoalfabetica. La sostituzione è data da una funzione detta *affine*:

$$c_i = E(p_i) = (k_1 p_i + k_2) mod 26$$

$\rightarrow$ la chiave è quindi data da **due costanti**.
La *decrittazione* avviene invece secondo la formula:

$$p_i = D(c_i) = (c_i - k_2) \cdot k_1^{-1}$$

con k_1^{-1} inteso come l'inverso modulo 26 di k_1, ovvero quel numero x che soddisfa l'equazione:

$$(k_1 \cdot x) mod 26 = 1$$

Affinché questo sia possibile è necessario che k_1 e 26 siano primi tra loro.
È un altro modo per scrivere la mappatura da un alfabeto in chiaro ad uno cifrante; dato che deve essere rispettata la condizione di invertibilità, si ottiene un sottoinsieme di tutti i possibili 26! alfabeti.

2.4.1 Crittoanalisi

Si riconduce ad un cifrario di sostituzione monoalfabetica, è vulnerabile ad una semplice analisi delle frequenze.

2.5 Cifrario di Playfair

Si ragiona su coppie di caratteri invece che su singoli caratteri: l'idea è che in questo modo si ha uno spazio di $26 \cdot 26$, è ancora possibile fare analisi delle frequenze ma è necessario un testo più ampio.

Si costruisce un rettangolo $5x5$, mettendo per convenzione nella stessa cella le lettere i e j; per cifrare viene tracciato un rettangolo tra la coppia di lettere, e la cifratura corrisponde ai vertici opposti.
Le lettere ripetute vanno separate da una lettera di riempimento (es. $mamma \rightarrow mamxma$)
Nel caso di lettere sulla stessa riga o colonna:

- stessa riga → si sostiuisce con lettere a destra
- stessa colonna → si sostituisce con lettere sottostanti

2.5.1 Crittoanalisi

È più sicuro rispetto alla cifratura monoalfabetica ($26 \cdot 26 = 676$ *digrammi*); è tuttavia sempre possibile fare un'analisi delle frequenze dei digrammi.

2.6 Cifrario di Hill

Permette di sostituire m lettere in chiaro con m lettere cifrate, usando m equazioni lineari.

Es. per m=2, (x_1,x_2) (y_1,y_2):

$$Y_1 = 11x_1 + 3x_2 \mod 26$$
$$Y_2 = 8x_1 + 7x_2 \mod 26$$

$$(y_1,y_2) = (x_1,x_2)\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

La chiave del cifrario è la matrice, i cui coefficienti vengono scelti in modo arbitrario.

Esempio

$$K=\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}. \qquad K^{-1}=\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$$

Il testo in chiaro è: *ciao*

$$(2,8)\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}. \qquad Y=(22+24,16+56)=(20,20)$$

$$(0,14)\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}. \qquad Y=(0+42,\ 0+98)=(16,20)$$

Il testo cifrato è: *UUQU*

Le fasi di cifratura e decifratura, in generale, sono:

$$y=E_k(x)=xK$$

$$x=D_k(y)=yK^{-1}$$

2.6.1 Crittoanalisi

È vulnerabile ad un attacco di tipo *known plaintext*: conoscendo la coppia chiaro-cifrato, tramite un calcolo matematico, è possibile calcolare la chiave.

L'avversario conosce m=2

$$\begin{pmatrix} 5 & 17 \\ 4 & 3 \end{pmatrix} K= \begin{pmatrix} 2 & 3 \\ 1 & 1 \end{pmatrix}$$

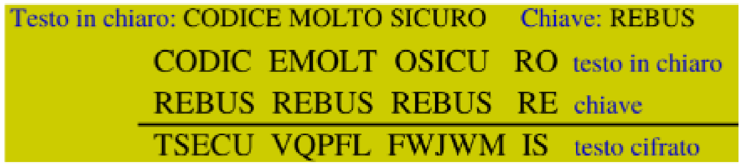
$$MK=C$$

$$M^{-1}MK=M^{-1}C \qquad \begin{pmatrix} 5 & 17 \\ 4 & 3 \end{pmatrix}^{-1}=\begin{pmatrix} 23 & 17 \\ 4 & 21 \end{pmatrix}$$

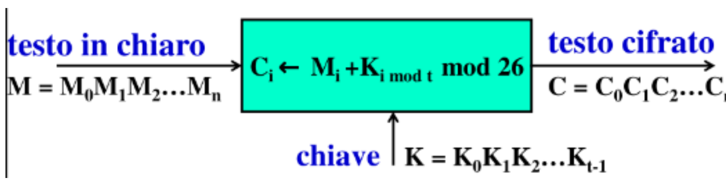
$$K=M^{-1}C$$

$$\begin{pmatrix} 23 & 17 \\ 4 & 21 \end{pmatrix}\begin{pmatrix} 2 & 3 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

2.7 Cifrario di Vigenère



Viene applicato un cifrario di shift per ogni singolo caratteri con la parola chiave; si ottiene che si hanno uguali caratteri cifrati che corrispondono a lettere in chiaro diverse e viceversa.
Matematicamente si può rappresentare in questo modo:



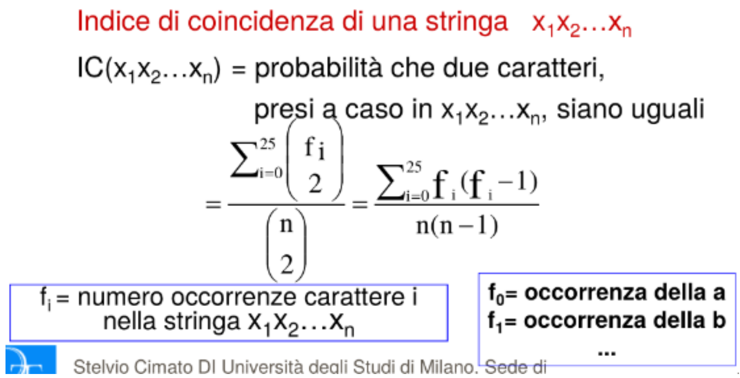
Sono possibili 26^t chiavi, con t lunghezza della chiave.

2.7.1 Crittoanalisi

È vulnerabile ad un attacco di tipo *known ciphertext*. Sfrutta dei metodi statistici per:

- determinare la lunghezza della chiave
  - indice di coincidenza
- determinare i caratteri della chiave
  - indice mutuo di coincidenza

Indice di coincidenza



L'indice di coincidenza è utile perché permette di capire qual è la lingua utilizzata; si fa un calcolo con n che tende ad infinito, e ciascuna lingua ha un suo valore per l'indice di coincidenza.
Si ok ma...come funziona=?

- si prova calcolare l'IC ponendo la lunghezza della chiave $t = 1$

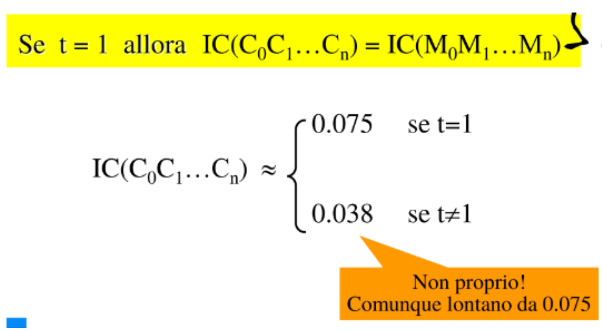


Figura 2.1: 0.038 = caratteri casuali, 0.075 = italiano

  se ottengo un valore noto vuol dire che ho trovato la lunghezza della chiave, altrimenti proseguo ponendo $t = 2$

- calcolo l'IC ponendo $t = 2$

Se t = 2 allora $IC(C_0C_2\ldots) = IC(M_0M_2\ldots)$ $\}$ L

$IC(C_1C_3\ldots) = IC(M_1M_3\ldots)$

$$\begin{array}{l} IC(C_0C_2\ldots) \approx \\ IC(C_1C_3\ldots) \approx \end{array} \begin{cases} 0.075 & \text{se } t=2 \\ 0.038 & \text{se } t\neq 2 \end{cases}$$

Comunque lontano da 0.075

- ...
- si continua iterativamente fino ad ottenere un valore noto

$$t = 5\ ? \quad \begin{cases} IC(C_0C_5\ldots) = 0.0710 \\ IC(C_1C_6\ldots) = 0.0721 \\ IC(C_2C_7\ldots) = 0.0805 \\ IC(C_3C_8\ldots) = 0.0684 \\ IC(C_4C_9\ldots) = 0.0759 \end{cases}$$

Tutti vicini a 0.075

$$\boxed{t = 5}$$

Indice mutuo di coincidenza

Si usa per determinare quali sono i caratteri che compongono la chiave.

Indice mutuo di coincidenza di $x_1x_2\ldots x_n$ e $y_1y_2\ldots y_{n'}$

$IMC(x_1x_2\ldots x_n; y_1y_2\ldots y_{n'})$ = probabilità che un carattere in $x_1x_2\ldots x_n$, ed uno in $y_1y_2\ldots y_{n'}$, presi a caso, siano uguali

$$= \frac{\sum_{i=0}^{25} f_i \cdot f'_i}{n \cdot n'}$$

f_i = numero occorrenze carattere i in $x_1x_2\ldots x_n$

f'_i = numero occorrenze carattere i in $y_1y_2\ldots y_{n'}$

E questo come funziona?

Supponiamo quindi di conoscere la dimesione t della chiave,

- La chiave è K= $(k_0, k_1, .. k_{t-1})$, con ogni k_i che indica il numero da addizionare per ottenere il testo cifrato

Si divide il ciphertext in t stringhe $y_0, y_1, \ldots y_{t-1}$

Si prendono due caratteri casuali di y_i e y_j:

- La probabilità che essi siano A è $p_{-k_i} p_{-k_j}$
- La probabilità che essi siano B è $p_{1-k_i} p_{1-k_j}$

- provo a porre $K_0 - K_1 = 0$

testo cifrato $C_0 C_1 \ldots C_n$

Se K_0-K_1= 0 allora $IMC(C_0 C_t \ldots; C_1 C_{t+1} \ldots) \approx 0.075$

$$IMC(C_0 C_t \ldots; C_1 C_{t+1} \ldots) \begin{cases} \approx 0.075 & \text{se } K_0\text{-}K_1 = 0 \\ \approx < 0.047 & \text{se } K_0\text{-}K_1 \neq 0 \end{cases}$$

- provo a porre $K_0 - K_1 = 1$

testo cifrato $C_0 C_1 \ldots C_n$

$Y_i \leftarrow C_i$ -1 mod 26

Se K_0-K_1= 1 allora $IMC(Y_0 Y_t \ldots; C_1 C_{t+1} \ldots) \approx 0.075$

$$IMC(Y_0 Y_t \ldots; C_1 C_{t+1} \ldots) \begin{cases} \approx 0.075 & \text{se } K_0\text{-}K_1 = 1 \\ \approx < 0.047 & \text{se } K_0\text{-}K_1 \neq 1 \end{cases}$$

- provo a porre $K_0 - K_1 = 1$

testo cifrato $C_0 C_1 \ldots C_n$

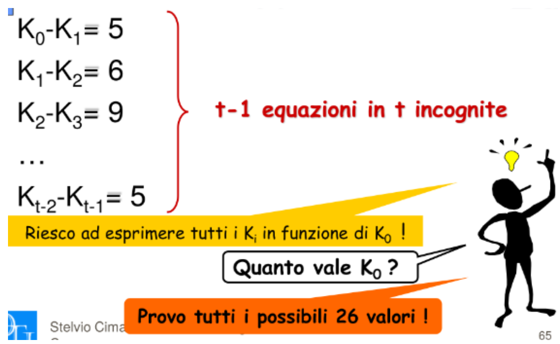
$Y_i \leftarrow C_i$ -2 mod 26

Se K_0-K_1= 2 allora $IMC(Y_0 Y_t \ldots; C_1 C_{t+1} \ldots) \approx 0.075$

$$IMC(Y_0 Y_t \ldots; C_1 C_{t+1} \ldots) \begin{cases} \approx 0.075 & \text{se } K_0\text{-}K_1 = 2 \\ \approx < 0.047 & \text{se } K_0\text{-}K_1 \neq 2 \end{cases}$$

- …si continua iterativamente fissando un sottotesto e modificando l'altro sottraendogli da 0 a 25; quando si trova un valore di IMC vicino a 0.075 (lingua italiana) significa che abbiamo trovato il corretto valore di shift $k_i - k_j$

A questo punto si possono scrivere $t-1$ equazioni in t incognite; si può riscrivere tutto in funzione di k_0, provando tutti i possibili 26 valori e vedendo quale soddisfa il sistema.



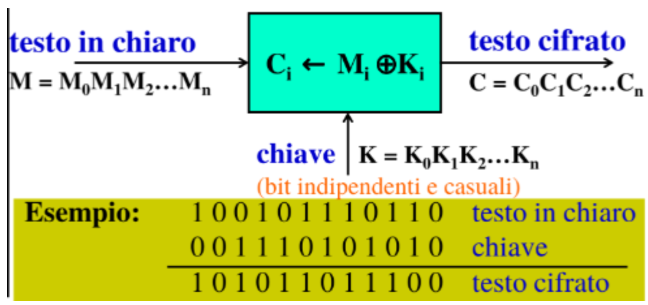
In sintesi:

- Determinare la lunghezza della chiave t
  - uso dell'indice di coincidenza

- Determinare il valore della chiave $K_0K_1K_2\ldots K_{t-1}$
  - calcolo delle differenze K_0-K_1, K_1-K_2 , …, K_{t-2}-K_{t-1}
    - uso dell'indice mutuo di coincidenza
  - calcolo di K_0: prova le 26 possibilità

2.8 One-Time-Pad

È definibile come il cifrario perfetto: viene presa una chiave lunga quanto il messaggio, i cui bit sono scelti in modo casuale.
Il messaggio risulta indecifrabile, perché il crittoanalista non può fare alcuna assunzione; il cifrario è *unconditionally secure*, perché a prescindere da tempo e risorse computazionali, non è possibile disitnguere quale sia il messaggio originale (il massimo che si può ottenere è tutte le possibili frasi di senso compiuto, ma senza poter stabilire qual è quella originale).



Il problema è che è necessario avere la chiave lunga quanto il messaggio che ci si vuole scambiare.

Capitolo 3

Cifrari a blocchi

I cifrari a blocchi operano su blocchi di n bit di input per produrre blocchi di altrettanti n bit in output; la trasformazione deve essere riversibile e dipende dalla chiave utilizzata.

Mappaggi

Si può pensare di fare un mappaggio tra tutti i possibili blocchi in input e il corrispondente cifrato, che deve essere unico; facendo questo mappaggio il numero di trasformazioni possibili è 2^n

Testo chiaro	cifrato
00	11
01	10
10	00
11	01

Il problema è che per blocchi di piccole dimensioni equivale a fare una cifratura a sostituzione (vulnerabile ad analisi statistica), mentre per blocchi grandi diventa un problema la gestione della chiave: **per blocchi di n bit la chiave è di dimensione** $n * 2^n$ (la chiave equivale alla tabella delle sostituzioni).

3.1 Cifratura di Feistel

L'idea è quella di approssimare un sistema ideale di cifratura, facendo uso di cifrature in sequenza per ottenerne una più complessa rispetto ad una singola trasformazione.
La chiave scelta permette di calcolare per ogni blocco di input uno di output; è una trasformazione robusta dato che è difficile da invertire se non si conosce la chiave.

3.1.1 Principi di Shannon

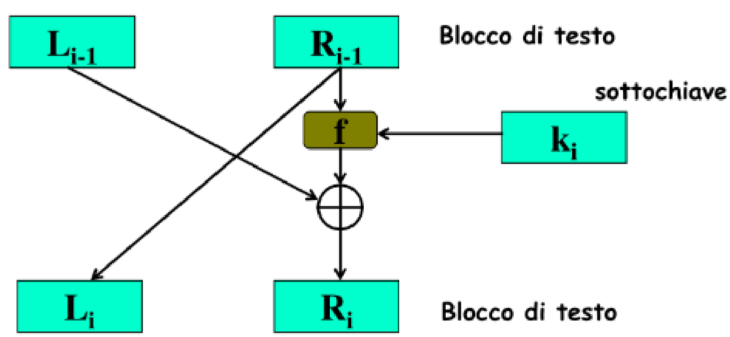
Feistel alterna **permutazioni** e **sostituzioni**; applicano i principi di Shannon per contrastare l'analisi statistica:

- **Diffusione:** ogni bit del cifrato dipende da più bit in chiaro; viene estesa la correlazione statistica, non c'è correlazione 1 : 1
- **Confusione:** si vuole aumentare la difficoltà nel capire la correlazione tra testo in chiaro/cifrato e la chiave; è reso possibile dal fatto che dalla chiave vengono generate più sottochiave, per un cui un cambiamento di un singolo bit della chiave genera cambiamenti a cascata su tutte le sottochiavi e il cifrato

3.1.2 Struttura di Feistel

- Blocchi e chiave di grande dimensioni aumentano la sicurezza ma diminuiscono le prestazioni
- Tutte le fasi hanno la stessa struttura
- A partire dalla chiave vengono prodotte tante sottochiavi quanti sono i *round*
- La funzione di round deve essere non lineare (altrimenti sarebbe facilmente invertibile)

La struttura di Feistel è la seguente:

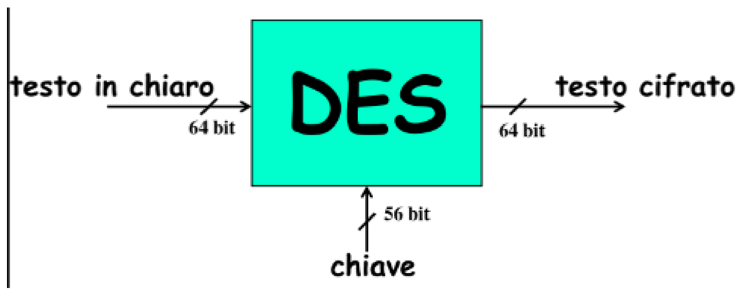


Il blocco di input viene diviso in due (*left* e *right*), e l'output viene ottenuto secondo la procedura mostrata in figura.
Questa operazione viene ripetuta per tutti i blocchi, per implementare i principi di Shannon.
È da notare come viene sfruttata la proprietà dello XOR per cui è possibile decifrare a prescindere della funzione f utilizzata.

- **Cifratura:** basta implementare un solo round, che verrà ripetuto su tutti i blocchi

- **Decifratura:** usa lo stesso algoritmo ma con le sottochiavi in ordine inverso

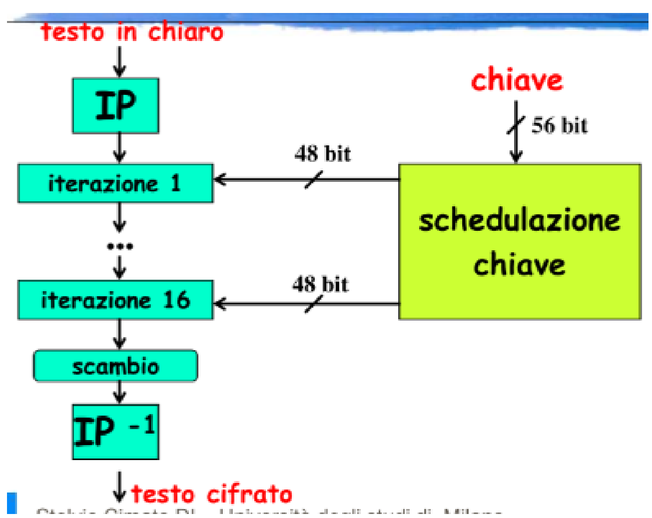
3.2 DES



Usa la struttra di Feistel, con:

- blocchi di 64 bit
- chiave di 64 bit, di cui 56 usati effettivamente dall'algoritmo
  - l'ultimo bit di ciascun byte viene usato come bit di parità (è lo XOR dei 7 precedenti)
  - spazio delle chiavi di 2^{56}

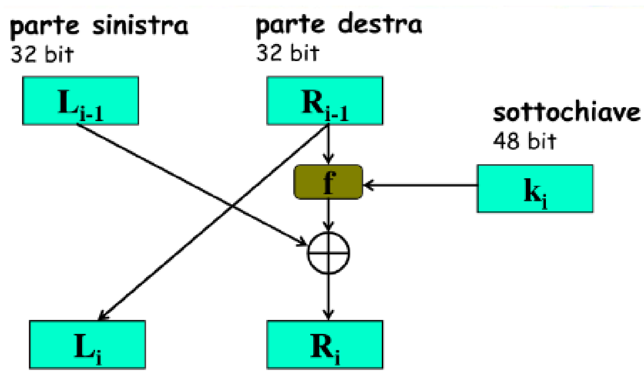
3.2.1 Struttura di DES



IP e IP^{-1}

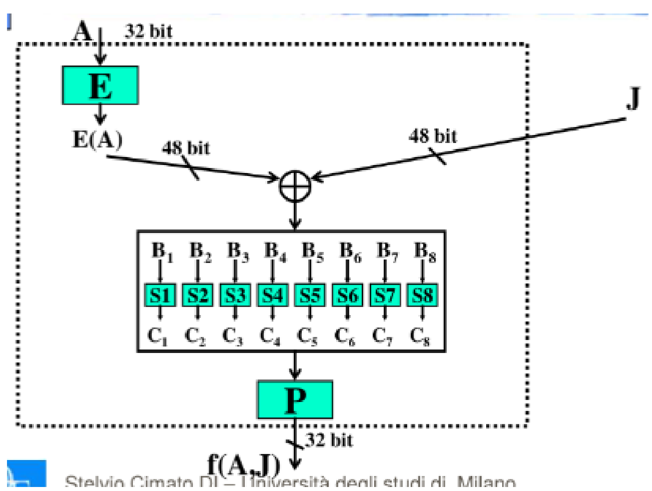
Una permutazione iniziale ed inversa alla fine; usano tabelle fissate per fare permutazioni dei bit.

Singola iterazione



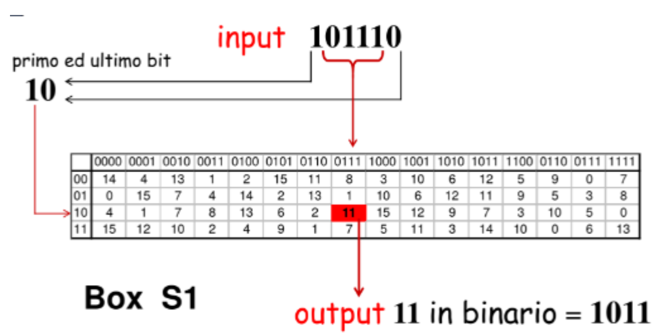
La funzione f

I 32 bit di input vengono espansi a 48, e poi vengono messi in XOR con la chiave.
I 48 bit risultanti vengono gestiti da 8 ***S-Box*** (6 bit ciascuna), ciascuna delle quali produce 4 bit per otterne 32; questi 32 bit vengono dati ad una funzione di permutazione e si ottiene il risultato finale.



S-Box

I 6 bit dati in input alle s-box fungono da indici di riga e di colonna per una tabella contenente tutti i possibili 16 valori dei 4 bit di output

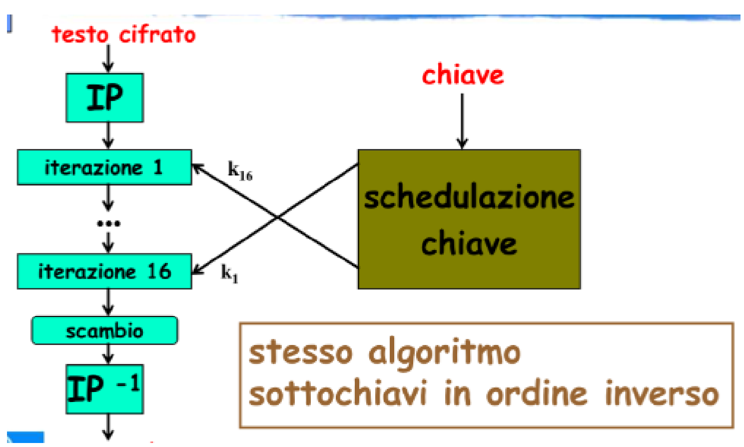


Le s-box hanno due proprietà importanti:

- cambiando un solo bit di input variano almeno due bit nell'output
- il numero di input per il quale il bit di output è 0 o 1 è circa lo stesso

3.2.2 Decifratura

Funziona allo stesso identico modo, ma usando l'ordine inverso delle sottochiavi.



3.2.3 Caratteristiche di DES

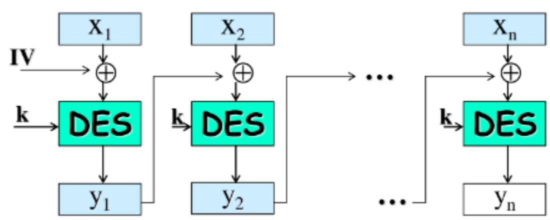
Cosidetto **effetto valanga:** piccoli cambiamenti nel testo in chiaro provocano un grande cambiamento nel cifrato; lo stesso vale per la chiave e l'algoritmo di schedulazione.
Rende difficile per un attaccante fare delle analisi tra i bit input e di output.

3.2.4 Modalità operative

Fino ad ora abbiamo visto come viene cifrato un singolo blocco; ora vediamo le modalità con cui cifrare più blocchi. Sono valide per tutti i cifrari simmetrici.

- **Electronic Codebook Chaining (ECB):** ciascun blocco di 64 bit viene cifrato in maniera indipendente; ha il vantaggio di evitare la propagazione degli errori ma ha il problema di essere deterministico (a stesso blocco e chiave, corrisponde semmpre lo stesso output); non è sicuro se usato con messaggi lunghi

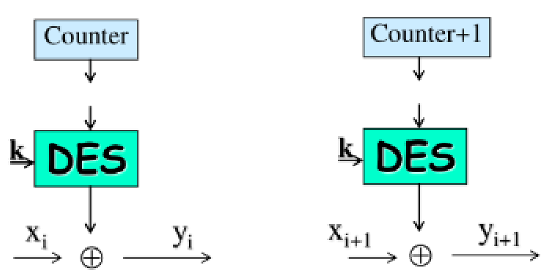
- **Cipher Block Chaining (CBC):** l'input si ottiene facendo lo XOR con il precedente blocco cifrato; per il primo blocco si usa un *inizialization vector* da scegliere in modo casuale; c'è propagazione degli errori ma la cifratura non è più deterministica



- **Cipher Feedback (CFB):** viene usato un registro a scorrimento che permette di poter usare la lunghezza del blocco a piacere; usa sempre un *iv* per rompere il determinismo, ha propagazione degli errori; l'idea è quella di avvicinarsi ad un cifrario a flusso dato che si può scegliere la lunghezza del blocco a piacere

- **Output Feedback (OFB):** ha il vantaggio di non propagare gli errori di trasmissione dei bit (usato ad esempio per trasmissione con i satelliti); ha lo svantaggio di essere più vulnerabile ad una modifica del flusso

- **Counter (CTR):** si usa un contatore delle dimensioni del blocco in chiaro; per ogni blocco successivo il contatore viene incrementato



Ha il vantaggio di essere sicuro ed efficiente

3.2.5 Crittoanalisi

Avendo una chiave di 56 bit DES è vulnerabile ad un attacco di brute force.

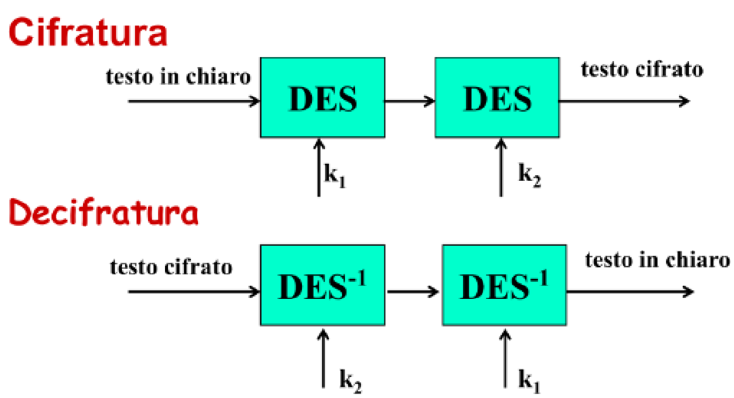
- La modalità ECB, essendo deterministica, non è resistente al CPA
- Le altre modalità sono tutte resistenti a CPA, dato che l'*iv* introduce un fattore di casualità
- Rimangono comunque degli schemi vulnerabili al CCA

Alcune tecniche di crittoanalisi (applicabili a tutti i cifrari a blocchi) sono:

- crittoanalisi differenziale → CPA, anche se necessita di 2^{47} testi in chiaro (poco applicabile nella realtà)
- crittoanalisi lineare → KPA, poco applicabile perché necessita di 2^{43} testi noti

3.3 DES doppio

Consiste nel cifrare due volte, applicando DES con due chiavi distinte.



È dimostrato che l'insieme delle permutazioni delle chiavi DES non è chiuso per composizione, ovvero che non sempre esiste una chiave $k_3 \equiv k_1 \cdot k_2$ (nel senso che dà in output lo stesso cifrato della doppia cifratura).

3.3.1 Attacco *meet in the middle*

È un attacco di tipo *known plaintext*.
Data una coppia nota (m, c) vengono eseguite tutte le cifrature per tutti i possibili 2^{56} valori di k_1, e memorizzate in una tabella.
Vengono eseguite tutte le possibili decifrature per i 2^{56} possibili valori di k_2, e si cerca una corrispondenza nella tabella.

Anche se si ha uno spazio delle chiavi delle chiavi di 2^{112}, la complessità computazionale rimane di 2^{56}; lo stesso vale in termini di spazio per salvare le righe della tabella.

3.4 DES-X

È una tipologia di DES che usa una tecnica detta *key withening* per migliorare la sicurezza rispetto al DES classico.
In questo caso, il cifrario non viene rafforzato contro attacchi analitici (analisi differenziale e lineare), ma solo contro gli attacchi a forza bruta. Vengono applicati due XOR prima e dopo l'esecuzione dell'algoritmo DES con due chiavi aggiuntive (una per XOR): $DES - X(M) = k_2 \oplus DES(M \oplus k_1)$

3.5 AES

Non è un cifrario di Feistel, ma opera in parallelo sull'intero blocco di input.

- blocchi e chiave da 128 bit
- 10 round
- vengono schedulate 44 sottochiavi a 32 bit

L'unità di base di AES è il byte; viene rappresentato sotto forma di una matrice $4x4$, che prende il nome di *stato*; tutte le operazioni prendono e restituiscono un byte, definendo un campo finito su valori fino a 2^8.
Ogni round è una composizione parallela di 4 operazioni, che prendono in input la matrice di stato:

- **SubBytes**
- **ShiftRows**
- **MixColumns**
- **AddRound Key**

Fanno eccezione una *AddRound Key* prima dei round, e l'ultimo dove manca la *MixColumns*.

SubBytes

Viene fatta la sostituzione dei byte con un altri, mediante un s-box tabellare, costruita in modo da resistere alla crittoanalisi.

ShiftRows

Viene fatto uno shift delle righe:

- la seconda shiftata di 1
- la terza shiftata di 2
- la quarta shiftata di 3

In questo modo con questa semplice permutazione i 4 byte di una colonna vengono riposizionati su colonne differenti.

MixColumns

Una colonna viene moltiplicata per un polinomio fissato, viene fatta una combinazione lineare.

AddRoundKey

Viene fatto lo XOR tra una colonna e la sottochiave di round.

3.6 Considerazioni sulla sicurezza

Ciò che viene fatto con i cifrari a blocchi è mettere il risultato in XOR con il successivo *plaintext*; è un modo per approssimare il cifrario perfetto dove si ha una chiave casuale lunga quanto il messaggio (sicurezza perfetta).
Usare i cifrari a blocchi in questo modo consente di ottenere un flusso di bit che si avvicina ad essere casuale; si sta approssimando OTP con 64 bit (DES) e 128 bit (AES).
Come operazione viene usato lo XOR perché non permette di trarre alcuna informazione osservando il cifrato.

Capitolo 4

Crittografia Asimmetrica

Una funzione botola è una funzione facile da computare in una direzione, ma difficile da computare in direzione opposta. RSA è un esempio di implementazione di funzione di questo tipo.

4.1 RSA

È un algoritmo di crittografia asimmetrico utilizzabile per criptare, decriptare, effettuare autenticazione tramite firma digitale e scambiare chiavi da usare in seguito per sistemi a cifratura simmetrica.
È basato sull'elevata complessità computazionale della fattorizzazione in numeri primi, che ne garantisce la sicurezza.

Generazione delle chiavi

La generazione delle chiavi per ciascun utente segue i seguenti passi:

1. scelti due primi p e q casuali da almeno 2048 bit
2. viene calcolato $n = pq$ e $\phi(n) = (p-1)(q-1)$
3. viene scelto un esponente pubblico e che sia coprimo con $\phi(n)$ (e minore di n)
4. viene calcolato l'esponente privato d tale che il prodotto con l'esponente pubblico sia congruo a 1 modulo $\phi(n)$
   $\rightarrow ed \equiv 1 \ mod \ \phi(n)$

- chiave pubblica $k_u = (e, n)$
- chiave privata $k_p = (d, p, q)$

Per la generazione dei parametri:

- per trovare p e q vengono generati dei numeri dispari casuali, e poi viene fatto un test di primalità
- per trovare l'esponente privato si utilizza l'algoritmo di Euclide esteso

Cifratura e decifratura

- **Cifratura:** $c = m^e \ mod \ n$
- **Decifratura:** $m = c^d \ mod \ n$

Calcolo dell'esponente

L'operazione di calcola della potenza modulare $x^y \ mod \ z$ può essere implementata in diversi modi:

- *naive:* si moltiplica x per sé stesso per y volte facendo il modulo ad ogni passo, per evitare cifre grandi; poco efficiente siccome la complessità è asintotica a y
- *left to right:* siano $y_0 y_1 \dots y_n$ i bit che formano y
  - definiamo $y = y_0 + 2(y_1 + 2(y_2 \cdots + 2(y_{n-1} + 2y_n)))$
  - da cui $\Rightarrow x^y = x^{y_0} + 2(x^{y_1} + 2(x^{y_2} \cdots + 2(x^{y_{n-1}} + 2 \cdot x^{y_n})))$
- *right to left:* siano $y_0 y_1 \dots y_n$ i bit che formano y
  - definiamo $y = y_0 \cdot 2^0 + y_1 \cdot 2^1 + \cdots + y_n \cdot 2^n$
  - da cui $\Rightarrow x^y = x^{y_0 \cdot 2^0} + x^{y_1 \cdot 2^1} + \cdots + x^{y_n \cdot 2^n}$

Con gli ultimi due metodi la complessità passa da essere esponenziale a lineare rispetto al numero dei bit di y.

4.2 Crittosistema di El-Gamal

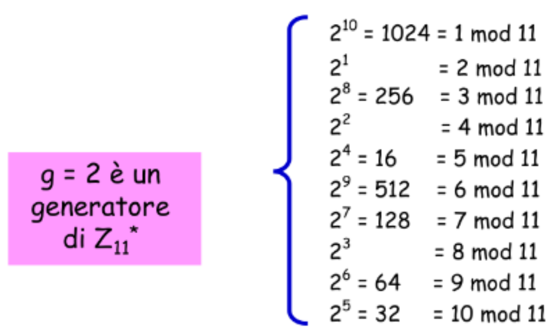
4.2.1 Concetto di logaritmo discreto

- Definiamo un gruppo finito G con operazione $\oplus$
- Definiamo H il sottogruppo generato da a, ovvero a^i con $i < 0$
- Definiamo $a \in G$ e $b \in H$
- L'unico intero x tale che $a^x = a \oplus a \oplus \cdots \oplus a = b$ viene chiamato logaritmo discreto di b in base a

Possiamo definire anche come $a^x = b \ mod \ N$
$\Rightarrow$ se N è primo, diventa difficile invertire il calcolo

4.2.2 Generatori

- definiamo $Z_p = \{0, \ldots, p-1\}$
- definiamo Z_p^* i numeri di Z_p che sono coprimi con p
- g è un generatore di Z_p^* se elevandolo a tutte gli elementi di Z_p si ottengono tutti gli elementi che compongono Z_p stesso



4.2.3 Generazione delle chiavi

Il crittosistema di El-Gamal si base sul concetto di logaritmo discreto; il processo per la generazione delle chiavi è il seguente:

- ogni utente genera un numero primo p molto grande e calcola:
  - un generatore g di Z_p^*
  - un α casuale compreso tra 1 e $p-1$
- la chiave pubblica è (p, g, β), con $\beta = g^\alpha \ mod \ p$
- la chiave privata è α

4.2.4 Cifratura

- codificare il messaggio come un intero m in Z_p
- scegliere un intero k compreso tra 1 e $p-1$
- calcolare $y_1 = g^k \ mod \ p$
- calcolare $y_2 = m\beta \ mod \ p$

$\Rightarrow$ il messaggio cifrato corrisponde alla coppia (y_1, y_2)

4.2.5 Decifratura

- calcolare $z = y_1^\alpha \ mod \ p$
- calcolare $m = z^{-1} \cdot y_2 \ mod \ p$, con z^{-1} inverso modulare di z