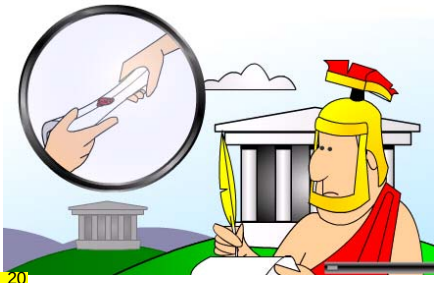
Capitolo 3



Crittologia classica

[20]

3.1 La storia ……...……44

3.2 L'eredità ……...……44

- Steganografia e Crittologia
- Codici e Cifrari
- Sostituzione e Trasposizione
- Robustezza di un Cifrario

3.3 Cifrari classici ……...……..47

- Sostituzione monoalfabetica di caratteri
- Trasposizione dei caratteri
- Sostituzione polialfabetica
- Il Cifrario di Vernam-Mouborgne e One-time pad

3.4 Sicurezza di un Cifrario e Teoria dell'informazione …..…....59

- Elementi di Teoria dell'informazione
- Sicurezza incondizionata
- Sicurezza computazionale

[20] Immagine tratta dalla presentazione "e-SecurityExperience" predisposta dalla ditta RSA e prelevabile anche dal sito del corso

3.1 La storia

La Crittografia classica si è occupata esclusivamente di **difesa della riservatezza** impiegando o **algoritmi segreti**, o **algoritmi simmetrici** noti a tutti e **chiavi segrete**. Le origini sono antichissime.

ESEMPI[21] – Il primo caso documentato è del 1900 A.C., riguarda gli scribi egizi e si riferisce all'uso di geroglifici non standard trovati su alcune tavole d'argilla; una formula cifrata per la verniciatura di vasellame è stata scoperta in Mesopotamia e risale al 1500 A.C.; nel 500 A.C. gli ebrei impiegano il cifrario ATBASH per consentire ai soli iniziati la lettura del libro di Geremia; nello stesso periodo gli spartani inventano ed usano il primo meccanismo per aiutare l'uomo nella cifratura e nella decifrazione (la scitale di cui parla Plutarco e che descriveremo tra poco); i romani impiegano il Cifrario circa 400 anni dopo (ne parla Svetonio nella "Vita di Cesare").

Il Kama Sutra (IV secolo D.C.) include la pratica di nascondere il significato di scritti e di parole tra le 64 arti che uomini e donne devono conoscere. Il primo testo di crittografia è stato scritto da Abu al Yahmadi nel 725 D.C.

Il primo vero salto di qualità nell'arte delle "scritture segrete" si verifica tra il XV° ed il XVI° secolo con l'invenzione ed il perfezionamento dei Cifrari a sostituzione polialfabetica e delle macchine che ne semplificano l'impiego (i contributi più significativi sono stati dati da Leon Battista Alberti, Tritemius, Bellaso, Giovan Battista della Porta e Vigenère). Per quasi tre secoli i Cifrari polialfabetici sono stati ritenuti inviolabili e questa errata convinzione determina un lungo stallo nell'evoluzione delle tecniche crittografiche.

Il secondo salto di qualità si verifica solo nel 1917, quando Gilbert S. Vernam inventa una macchina polialfabetica che impiega una chiave scelta a caso e lunga quanto il testo.

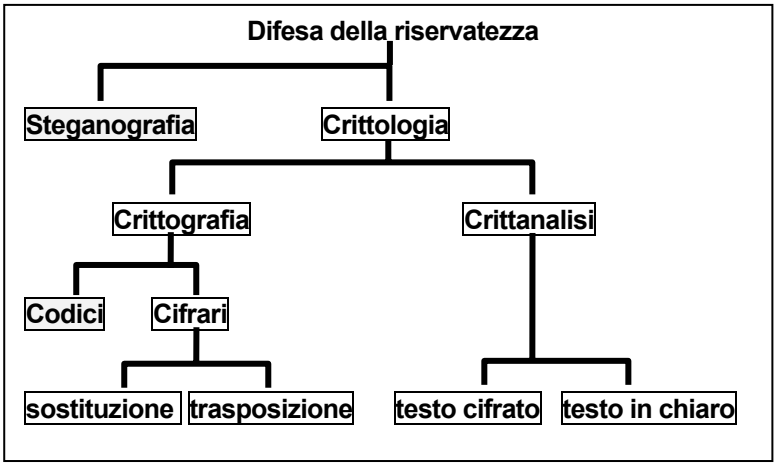
Contributi importanti all'attuale Crittologia sono successivamente stati dati sia dalle macchine cifranti usate nella II[a] Guerra Mondiale, sia dalle macchine crittanalitiche parallelamente sviluppate per violare la segretezza dei loro testi cifrati.

ESEMPI – La macchina cifrante più nota è Enigma (v. il simulatore disponibile nel sito del corso), inventata da Arthur Schrebius nel 1929 ed usata dalle Armate tedesche nella II[a] Guerra Mondiale. Si stima che a metà conflitto le macchine in servizio furono più di 100.000. Gli Alleati usavano la più robusta macchina di Hagelin.

Il matematico inglese Alan Touring, un professore di Cambridge già famoso per i suoi studi teorici sulle funzioni computabili, ha guidato, per conto dei Servizi Segreti inglesi, il progetto e la realizzazione delle macchine elettromeccaniche che sono riuscite a "rompere" Enigma. La versione a valvole termoioniche, resa disponibile dagli americani verso la fine del conflitto e nota con il nome di Colossus, deve essere considerata a pieno diritto uno dei progenitori degli attuali calcolatori elettronici.

3.2 L'eredità

Da tutta questa lunga storia discende la tassonomia indicata in figura.



La difesa della riservatezza può essere affidata a due differenti metodologie: la **Steganografia** e la **Crittologia**.

La Crittologia, la Scienza che intendiamo qui studiare, si articola in **Crittografia** ed in **Crittanalisi**.

La Crittografia classica ha impiegato due differenti strumenti: il **Codice** ed il **Cifrario**. I Cifrari, i soli ad interessare la Crittografia moderna, si avvalgono di due sole operazioni: la **sostituzione** e la **trasposizione**.

La Crittanalisi classica si è basata sulla conoscenza o di **solo testo cifrato**, o di **coppie di testo cifrato e di testo in chiaro** corrispondente.

Il significato di alcuni dei termini che abbiamo impiegato verrà chiarito nei prossimi paragrafi. Una volta creato questo quadro di riferimento, potremo più agevolmente esaminare come erano fatti i Cifrari classici e come sono stati violati. L'intendimento è duplice:

- inquadrare con casi semplici le complesse modalità realizzative dei Cifrari moderni;
- sottolineare l'importanza dell'uso di un calcolatore nei contesti della Crittografia e della Crittanalisi.

[21] da Table A1: History of Cryptography in M.E. Whitman, H.J. Mattord: "*Principles of information security*" Thomson Course Technology 2003

3.2.1 Steganografia e Crittologia

Il problema della **comunicazione in presenza di avversari** può essere affrontato facendo ricorso a due differenti metodologie:

- nascondere l'informazione da proteggere all'interno di un'altra informazione non riservata (**Steganografia**);
- rendere il messaggio inintelligibile per chiunque non ne sia il legittimo destinatario (**Crittologia**).

La Steganografia studia dunque come comunicare **senza che altri se ne accorgano**.

La Crittologia studia invece come comunicare **senza che altri capiscano**.

L'argomento della Steganografia esula dai limiti di questo corso. Per dare almeno una prima idea può però servire qualche esempio proveniente dal passato.

ESEMPI - Erodoto narra che Demerato, dovendo avvertire Sparta dell'imminente invasione della Grecia da parte di Serse e volendo impedire che il suo messaggio fosse intercettato, adoperò le allora usuali tavolette di legno ricoperte di cera, togliendo però inizialmente la ricopertura, incidendo il messaggio nel legno, rimettendo poi la cera ed incidendo infine su questa un messaggio insignificante.

Nelle Storie si parla anche di Istieo, che fece rapare uno schiavo, incise un messaggio segretissimo sulla cute del suo cranio, aspettò che gli ricrescessero i capelli e lo mandò poi ad attraversare le linee nemiche.

Molti secoli dopo la Steganografia ha scoperto ed usato gli inchiostri "invisibili" (come il latte, l'aceto, il succo di limone e l'urina, sostanze tutte che presentano la proprietà di scurirsi se avvicinate ad una fonte di calore).

Il primo trattato di Steganografia è stato scritto da Johannes Tritemius (1518).

Più importante è segnalare che la Steganografia è ancora di grande attualità.

Recentemente è stato avanzato il sospetto che i "video" di Bin Laden trasmessi per televisione contenevano anche messaggi nascosti indirizzati agli adepti di Al Quaeda disseminati in tutto il mondo.

L'uso più normale è però oggi quello di sancire e di difendere i *diritti d'autore* di un documento multimediale (un testo e/o un'immagine e/o un suono): a tal fine all'interno delle stringhe di bit che lo codificano è nascosto un messaggio invisibile, che svolge praticamente lo stesso ruolo della "filigrana" (*watermark*) nelle banconote.

ESEMPIO - Consideriamo un'immagine bit-map formata da 680x480 pixel singolarmente colorabili, tramite 8 bit, con un colore scelto tra 256 possibili. Sottraendo al suo uso normale il bit meno significativo del colore di ogni pixel si ottiene sia un'immagine in pratica non distinguibile da quella originaria, sia circa 300 Kbit ai quali è possibile affidare la codifica di un messaggio invisibile.

3.2.2 Codici e Cifrari

La Crittografia classica ha mostrato l'esistenza di due differenti metodi di trasformazione dei messaggi:

- il **Codice**,
- il **Cifrario**.

I campi di applicazione sono diversi e dipendono dall'assunzione o meno di vincoli sul testo in chiaro.

Esistono alcune situazioni in cui la stringa del messaggio da comunicare è a priori vincolata all'impiego di un alfabeto formato soltanto da simboli "complessi" (ad esempio determinate parole o frasi nel caso di un testo, determinate figure geometriche nel caso di un disegno). In tali situazioni il compito di rendere riservato il contenuto informativo può essere affidato ad un **Codice**, una specie di **dizionario** che associa ad ogni simbolo del testo in chiaro un simbolo di tutt'altro significato.

La non generalità dei testi che è così possibile proteggere e l'onerosità di uso e di gestione del dizionario (è relativamente voluminoso, deve essere segretamente consegnato a tutti i corrispondenti, deve essere periodicamente modificato) hanno però da sempre fortemente delimitato il ricorso ai Codici.

E' sicuramente vero che anche i Cifrari classici trasformano i simboli di un insieme nei simboli di un altro insieme, ma con due punti di forza che li hanno resi praticamente insostituibili, oggi come allora:

- l'oggetto della trasformazione sono i simboli "elementari" del testo in chiaro, accorgimento che consente di proteggere qualsiasi messaggio;
- il metodo di trasformazione è un algoritmo con chiave, cioè una cosa poco ingombrante e soprattutto facile da tenere segreta e da modificare, come abbiamo già segnalato nel precedente capitolo.

3.2.3 Sostituzione e Trasposizione.

Il Cifrario è formato da due algoritmi: il primo trasforma il testo in chiaro (una stringa di simboli appartenenti ad un certo alfabeto) in testo cifrato (una stringa di simboli che possono appartenere o allo stesso alfabeto del testo in chiaro o ad un diverso alfabeto); il secondo esegue la trasformazione inversa.

Di norma, per semplicità, si usa lo stesso alfabeto e si trasforma un simbolo alla volta: un **carattere** nella Crittografia classica, un **pacchetto di bit** nella Crittografia moderna.

La Crittografia classica ha mostrato che la difesa della riservatezza può essere conseguita con due soli tipi d'operazione:

- la **sostituzione**, che trasforma ogni simbolo d'ingresso in un differente simbolo d'uscita,
- la **trasposizione**, che attribuisce ad ogni simbolo d'ingresso una diversa posizione nella stringa d'uscita.

Inizialmente è stata impiegata o l'una, o l'altra; successivamente si è capito che devono invece essere applicate entrambe, più e più volte.

Tutte e due le operazioni sono invertibili e definiscono un enorme insieme T di trasformazioni possibili.

Per valutarne la cardinalità, consideriamo un testo in chiaro formato da **m** simboli, tutti appartenenti ad un alfabeto che ne contiene **n**.

Nel caso della sostituzione esistono

$$|T| = n!$$

possibili permutazioni dell'alfabeto: per l'alfabeto inglese (26 caratteri) si ha $\mathbf{26!} \cong \mathbf{4 \times 10^{26}}$.

Se, per maggiore sicurezza, si assume che il sostituto di ogni simbolo non possa mai essere il simbolo stesso, le trasformazioni a disposizione sono un po' di meno:

$$|T| = \sum_{j=0}^{n} (-1)^j \frac{n!}{j!} \cong (n-1)!$$

Arrivati a questo punto, dobbiamo ritornare sulle considerazioni fatte nel capitolo 2. La scelta di una delle trasformazioni possibili può essere fatta con una **tabella a due righe** (questa è la **chiave segreta**) concordata dai due corrispondenti: nella riga in alto (eventualmente sottintesa) sono elencati ordinatamente i simboli del testo in chiaro, nella seconda quelli che li devono sostituire nel testo cifrato.

Agli albori della Crittografia ci si è illusi che fosse possibile limitare il numero di sostituzioni per poterle esprimere con una regola facile da imparare a memoria.

ESEMPIO - Il **Cifrario di Giulio Cesare** era privo di chiave: ogni carattere del testo in chiaro era sostituito dal carattere che lo seguiva di tre posti nell'ordinamento alfabetico: A con D, B con E,.., U con A, V con B, Z con C. La decifrazione si otteneva scambiando i caratteri coinvolti nelle 21 sostituzioni operate in cifratura (D con A, ecc.).

de bello gallico $\rightarrow$ il glqqt nfqqpht $\rightarrow$ de bello gallico

E' però possibile dotarlo di una chiave con 20 valori giocando sulla possibilità di associare alla A una qualsiasi delle restanti lettere, alla B la lettera successiva e così via: lo spazio delle chiavi è comunque troppo piccolo per dare sicurezza.

Il **Cifrario ATBASH** usava l'alfabeto invertito: la prima lettera dell'alfabeto del testo in chiaro era sostituita dall'ultima, la seconda dalla penultima e così via.

Nel caso della trasposizione di un testo di **m** simboli valgono ancora le formule precedenti, una volta che **n** è stato sostituito da **m**. Anche in questo caso la chiave può essere vista come una tabella: la prima riga indica, ad esempio in ordine crescente, le posizioni dei simboli nel testo in chiaro e la seconda riga quelle in cui devono essere trasferiti nel testo cifrato.

Limitando il numero di trasposizioni possibili si perde in sicurezza e si guadagna in semplicità d'uso.

ESEMPIO - La **scitale degli spartani** (un bastone con un certo diametro su cui veniva disposta a spirale una striscia di pergamena) prevedeva che il testo in chiaro fosse scritto su generatrici consecutive del cilindretto procedendo dall'alto verso il basso. Lungo la striscia appariva così alla fine un testo cifrato per **trasposizione**; per rimetterlo in chiaro occorreva un bastone di eguali dimensioni.

Θεμιστοκλεσ μεν ταυτα ενεγραφε $\rightarrow$

Θ	τ	σ		α	γ
ε	ο		τ		ρ
μ	κ	μ	α	ε	α
ι	λ	ε	υ	ν	φ
σ	ε	ν	τ	ε	ε

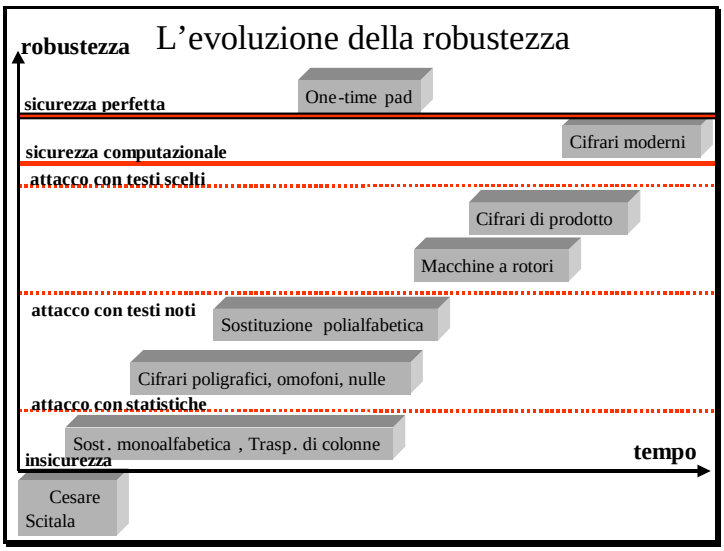
$\rightarrow$ Θτσ α γεο τ ρμκμαεαιλευνφσεντεε $\rightarrow$

Θ	τ	σ		α	γ
ε	ο		τ		ρ
μ	κ	μ	α	ε	α
ι	λ	ε	υ	ν	φ
σ	ε	ν	τ	ε	ε

Per introdurre il concetto di chiave basta pensare a bastoni con diametro e lunghezza diversi.

3.2.4 Robustezza di un Cifrario

L'intruso che intercetta messaggi cifrati può tentare di "decrittarli", termine di gergo che indica o l'individuazione della chiave, o la ricostruzione del testo in chiaro. Il successo dell'attacco dipende dalle vulnerabilità presenti nel Cifrario e dalla quantità di conoscenza necessaria per sfruttarle (v. pag. 27). La robustezza di un Cifrario può dunque essere misurata facendo riferimento ad una lista di attacchi a crescente pericolosità.



In figura sono mostrate sia la griglia degli attacchi predisposta dalla Crittanalisi, sia la collocazione dei Cifrari progressivamente messi a punto dalla Crittografia.

Come vedremo meglio tra poco, i Cifrari classici erano veramente poco robusti, essendo stato possibile violarli o con solo testo cifrato, o, al più, con testo in chiaro noto.

Si noti che i Cifrari moderni sono invece posti al di sopra della linea che certifica l'impossibilità **computazionale** di violarli, anche se si dispone di coppie testo cifrato/testo in chiaro scelte a piacere.

Si noti infine l'esistenza di un solo Cifrario, "one-time pad", posto al di sopra della linea della **sicurezza perfetta**: il testo cifrato intercettato, tutta la conoscenza che si vuole e risorse di calcolo illimitate non consentono, infatti, in questo caso ad un avversario di imparare nulla che non sapesse già.

3.3 Cifrari classici

3.3.1 Sostituzione monoalfabetica di caratteri

I Cifrari simmetrici che eseguono solo un'operazione di sostituzione sui singoli simboli di testo (in chiaro o cifrato) sono detti a sostituzione monoalfabetica di caratteri.

1 - Chiave, cifratura e decifrazione

Crittografia classica:
la sostituzione monoalfabetica

regola di sostituzione (o chiave)
A B C D E F G H I L M N O P Q R S T U V Z
| | | | | | | | | | | | | | | | | | | | |
Q E M R F Z T B L U P O N H A S C G V D I

testo in chiaro:**CRITTOGRAFIA**
testo cifrato: **MSLGGNTSQZLQ**

In figura è preso in considerazione l'alfabeto latino (21 caratteri): abbiamo già messo in evidenza che le permutazioni possibili sono 21! e che occorrono quindi altrettanti valori di chiave per consentire alla sorgente ed alla destinazione di scegliere ed eseguire una trasformazione ben precisa.

Ogni stringa di 21 caratteri diversi definisce l'alfabeto in uso nel testo cifrato ed è dunque una chiave.

Per cifrare si sostituisce ogni carattere del testo in chiaro con quello che ha la stessa posizione nel nuovo alfabeto.

Per la decifrazione ogni carattere del crittogramma è sostituito dal carattere che ha la stessa posizione nell'alfabeto originario.

Il Cifrario di Cesare ed il Cifrario ATBASH sono casi particolari (e particolarmente deboli) di sostituzione monoalfabetica.

ESEMPIO - **Codifica in Java dell'algoritmo di Cesare**[22]

```
public class AlgoritmoCesare {

  public AlgoritmoCesare() {
  }

  public String esegui(String input,String alfabeto){
    String res = "";
    String alfMin = "";
    String alfMai = "";
```

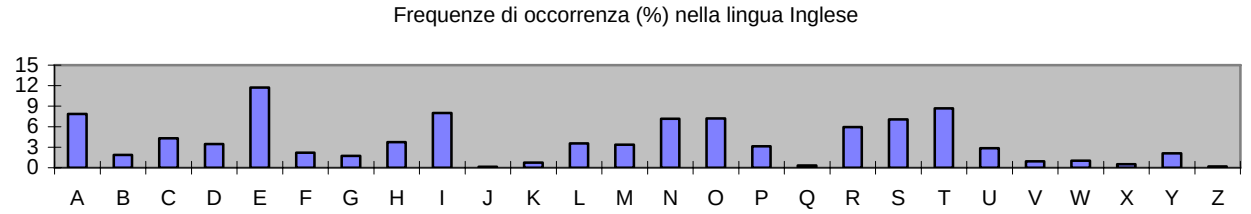
[22] Codice sviluppato da Anna Riccioni per il tool CryptoTest. L'eseguibile e gli altri sorgenti sono disponibili nel sito del Corso

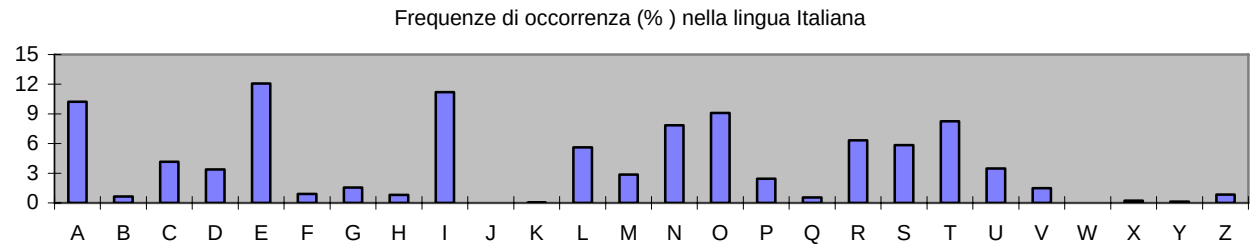
```
    int lunghAlf = 0;
    if (alfabeto.equals("Inglese")){
      alfMin = "abcdefghijklmnopqrstuvwxyz";
      alfMai = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
      lunghAlf = 26;
    }
    else{
      alfMin = "abcdefghilmnopqrstuvz";
      alfMai = "ABCDEFGHILMNOPQRSTUVZ";
      lunghAlf = 21;
    }
    char c1, c2;
    int idx;
    for (int i=0; i < input.length(); i++){
      c1 = input.charAt(i);
      c2 = c1;
      for (int j=0; j < lunghAlf; j++){
        if (c1 == alfMai.charAt(j)){
          idx = (j + 3) % lunghAlf;
          c2 = alfMai.charAt(idx);
        }
        if (c1 == alfMin.charAt(j)){
          idx = (j + 3) % lunghAlf;
          c2 = alfMin.charAt(idx);
        }
      }
      res = res.substring(0, i) + c2;
    }

    return res;
  }
}
```

2 - Crittanalisi

Nel caso di messaggi in linguaggio naturale, la semplice sostituzione di un carattere alla volta è una facile preda per il cosiddetto **attacco statistico** (*statistical attack*).

Ogni linguaggio naturale impiega i caratteri dell'alfabeto con frequenze non solo diverse, ma anche largamente indipendenti dalla semantica.





Per ogni lingua si può dunque parlare di una **probabilità di occorrenza** di ciascun carattere all'interno di un testo e, disponendo di abbastanza testo da analizzare, approssimarla con la **frequenza di occorrenza**.

Elementi caratteristici di ogni linguaggio sono le occorrenze dei singoli caratteri nell'intero testo, all'inizio ed alla fine di ciascuna parola, nell'ambito di stringhe formate da due o da tre caratteri consecutivi (*digrammi* e

trigrammi). Stringhe più lunghe non sono invece di alcuna utilità: le probabilità di occorrenza sono piccolissime, differiscono di poco una dall'altra e dipendono inoltre dal contesto.

ESEMPI - Nella lingua inglese il digramma più usato, TH, ha un'occorrenza pari al 3,16%, ed è seguito da IN (1,54%), ER (1,33%), RE (1,3%), ecc.; per quanto riguarda i trigrammi, THE ha una percentuale di occorrenza del 4,72, ING del 1,42 e così via.

Un linguaggio naturale è dunque *ridondante*. La disponibilità di statistiche sull'uso dei simboli alfabetici[23] ed il calcolo della frequenza di occorrenza dei simboli presenti nel testo cifrato consentono di rompere agevolmente una sostituzione monoalfabetica di caratteri. Il motivo è presto detto:
"**le proprietà statistiche di ogni carattere del testo in chiaro sono trasferite immutate sul carattere che lo sostituisce nel testo cifrato**".

ESEMPIO - Supponiamo che il messaggio cifrato

UNUFT OST, SII QNUF RBU GQFIO, HQDWXRUF KURGQFVL SKO BQLR ...

sia il risultato della trasformazione di un testo inglese con una sostituzione monoalfabetica di singoli caratteri.

Supponiamo inoltre di aver ottenuto i seguenti risultati dall'analisi frequenziale dei caratteri del testo cifrato:

- U, R e S sono i caratteri più ricorrenti (rispettivamente con 15,3%, 9,8% e 7,8%)
- i digrammi con maggiore occorrenza, tutti con 3,3%, sono UF, FU e RB.
- il trigramma più ricorrente è RBU (3,5%), seguito da USV (2%).
- S esiste più volte da solo ed ha un'occorrenza del 13% come prima lettera di altre parole

Sulla base delle occorrenze più alte è facile ipotizzare che U corrisponda a ***E,*** R a ***T***; una conferma è data dal trigramma RBU che è quindi ***THE,*** evidenziandosi così l'ulteriore corrispondenza tra B e ***H***.

Applicando queste prime e molto probabili sostituzioni si ha:

EN***E***FT OST, SII QN***E***F ***THE*** GQFIO, HQDWX***TE***F K***ET***GQFVL SKO ***H***QL***T*** .....

Dalle statistiche discende anche che è molto probabile che S corrisponda ad ***A*** e che i digrammi UF, FU siano in chiaro ***ER, RE*** (cioè che F corrisponda a ***R***).

Applicando queste ulteriori sostituzioni si ha:

EN***ER***T O***A***T, ***A***II QN***ER THE*** GQ***R***IO, HQDWX***TER*** K***ET***GQ***R***VL ***A***KO ***H***QL***T***..

Con uno sforzo modesto abbiamo dunque messo in chiaro 18 caratteri su un totale di 46, cioè circa il 40%. Statistiche non ancora prese in considerazione ed un diretto intervento dell'uomo porteranno abbastanza facilmente alla seguente conclusione:

EVERY DAY, ALL OVER THE WORLD, COMPUTER NETWORKS AND HOSTS…

3 - Accorgimenti difensivi

Per rendere meno pericoloso l'attacco statistico, la Crittografia classica ha richiesto a chi impiegava sostituzioni monoalfabetiche di adottare uno o più dei seguenti provvedimenti:

1. eliminare dal testo in chiaro **spaziature** e **segni di interpunzione**;
2. introdurre a caso nel testo cifrato dei caratteri non significativi (le cosiddette **nulle**);
3. impiegare nel testo cifrato un alfabeto più grande di quello usato nel testo in chiaro per poter poi sostituire con più simboli (gli **omofoni**) i caratteri più frequenti;
4. sostituire due o tre caratteri alla volta (Cifrari **poligrafici**).

ESEMPIO – Il Cifrario Playfair[24], inventato nel 1854 ed usato dagli inglesi nella I[a] Guerra Mondiale, trasforma digrammi di testo in chiaro in digrammi di testo cifrato. La cifratura e la decifrazione si basano su una matrice 5×5,

C	**I**	**A**	**O**	B
D	E	F	G	H
K	L	M	N	P
Q	R	S	T	U
V	W	X	Y	Z

che i corrispondenti riempiono dapprima con i caratteri diversi presenti in una concordata "parola chiave" (in figura CIAO) e poi, nell'ordine, con i restanti caratteri dell'alfabeto inglese (I e J sono considerati un solo carattere). In generale il digramma in chiaro identifica la diagonale di un "rettangolo": il digramma cifrato è formato dai caratteri posti alle estremità dell'altra diagonale (es. FU → HS). Se i caratteri in chiaro sono sulla stessa riga, i caratteri del cifrato sono quelli scritti nelle caselle sulla loro destra, con la precisazione che la prima casella di ogni riga è "alla destra" dell'ultima (es. EF → FG, SU → TQ). Se i caratteri in chiaro sono sulla stessa colonna, i caratteri del cifrato sono quelli scritti nelle caselle sottostanti, con la precisazione che la prima casella di ogni colonna è "al di sotto" dell'ultima (es. ON → GT, MX → SA).

Se infine il digramma in chiaro è una "doppia" o si elimina un carattere (es. GATTO = GATO → FOYG), o si inserisce un carattere improbabile (GATTO = GATXTO → FOSYYG).

[23] Oggi è semplicissimo ottenerle anche con un PC: provare per credere. Una volta era necessario consultare tavole computate a mano (v. ad es. M. Zanotti, "Crittografia" Manuali Hoepli,1928)

[24] Alessandro Dalla Torre ha ampliato il già citato CryptoTest con un eseguibile ed i relativi sorgenti.

Nei prossimi capitoli vedremo che la Crittografia moderna, simmetrica e asimmetrica, ricorre moltissimo alla sostituzione monoalfabetica ed è quindi giusto chiedersi subito qual'è l'accorgimento che fornisce robustezza all'attacco con statistiche.

La risposte è semplice e discende da quello che abbiamo appena detto: la sostituzione monoalfabetica viene applicata a blocchi formati da molti bit (come minimo 64, a cui corrispondono otto caratteri ASCII). Ciò è più che sufficiente per prevenire l'attacco statistico: la raccolta e la memorizzazione di 2^{64} dati statistici è computazionalmente impossibile; dati di questo tipo non sono più elementi identificativi del linguaggio naturale.

Utenti consapevoli della pericolosità dell'attacco con statistiche possono comunque far precedere la cifratura da una **compressione senza perdita** ed eseguire poi la trasformazione inversa dopo la decifrazione.

Indispensabile è comunque non dare all'avversario troppo materiale da analizzare.

- R24: "*non bisogna mai cifrare troppo testo in chiaro con la stessa chiave*"

3.3.2 Trasposizione dei caratteri

I Cifrari simmetrici che modificano solo la posizione dei simboli del testo in chiaro sono detti a trasposizione. Particolarmente semplice è la **trasposizione colonnare**, per la quale occorre un'operazione preliminare: si scrive il testo in chiaro da sinistra verso destra e dall'alto verso il basso all'interno di una tabella con **P** righe e **Q** colonne. Se il messaggio contiene più di **PxQ** caratteri, la preparazione della tabella deve essere ripetuta tante volte quante sono necessarie: l'ultimo blocco (o il primo e l'unico se il messaggio è corto) deve in generale essere completato con simboli di riempimento.

1 - Chiave, cifratura e decifrazione

Crittografia classica:
la trasposizione di colonne

Tabella **5 × 8** e chiave **76518234**:
Testo in chiaro: **ALLE PROSSIME ELEZIONI MI PRESENTO**

Cifratura:

7	**6**	**5**	**1**	**8**	**2**	**3**	**4**
A	**L**	**L**	**E**	**P**	**R**	**O**	**S**
S	**I**	**M**	**E**	**E**	**L**	**E**	**Z**
I	**O**	**N**	**I**	**M**	**I**	**P**	**R**
E	**S**	**E**	**N**	**T**	**O**	**X**	**X**

Testocifrato: **EEINRLIOOEPXSZRXLMNELIOSASIEPEMT**

I due corrispondenti scelgono come chiave un numero in cui compaiono, una volta sola ed in ordine qualsiasi, gli interi compresi nell'intervallo 1 ÷ Q.

Le chiavi possibili sono quindi **Q!**.

Una volta scritta la chiave al di sopra della tabella, la sorgente trasmette le **Q** colonne della tabella nell'ordine indicato dalle cifre della chiave.

La destinazione sistema ogni gruppo di P caratteri ricevuti in colonne affiancate, contenenti in testa un indice di ricezione incrementato progressivamente a partire da 1.

Una volta scritta la chiave al di sopra di questa matrice, la destinazione sposta le colonne in modo che il loro indice di ricezione venga a trovarsi esattamente sotto ad una cifra della chiave di eguale valore.

La Scitale è un caso particolare di trasposizione senza chiave.

2 - Crittanalisi

Essendo stata fatta solo una trasposizione, ogni carattere del testo cifrato mantiene le proprietà statistiche che ha nel linguaggio naturale.

Il calcolo delle occorrenze dei singoli caratteri non è dunque di alcuna utilità per la crittanalisi. Le statistiche dei digrammi del linguaggio originario consentono invece all'intruso di individuare agevolmente quali sequenze di due simboli del cifrato non sono "naturali", ma discendono dalla scrittura in colonne del testo in chiaro.

Una volta raccolte tutte queste informazioni, bastano pochi tentativi per individuare quanto vale P e per costruire quindi una matrice del tutto simile a quella che impiega la destinazione. A questo punto diventa agevole scoprire la chiave permutando le colonne della matrice fino a quando non si trova una frase significativa.

3.3.3 Sostituzione polialfabetica

La sostituzione polialfabetica si pone l'obiettivo di rendere il più possibile **equiprobabile** l'occorrenza di ogni simbolo del testo cifrato. A tal fine si avvale di **diversi alfabeti** per poter scegliere tra **altrettanti omofoni** il simbolo che dovrà essere inserito nel testo cifrato ad ogni occorrenza di ciascuno dei simboli del testo in chiaro.

Diverse sono state le tecniche messe a punto per trasformare i caratteri del testo in chiaro con una regola dipendente dalla loro **posizione** all'interno della stringa. Tra queste vanno ricordate le corone circolari concentriche impiegate da Leon Battista Alberti e la tavola introdotta da Vigenère.

1 - Chiave, cifratura e decifrazione del Cifrario di Vigenère

Crittografia classica:
la sostituzione polialfabetica

ABCDEFGHIJKLMNOPQRSTUVWXYZ
BCDEFGHIJKLMNOPQRSTUVWXYZA
CDEFGHIJKLMNOPQRSTUVWXYZAB
DEFGHIJKLMNOPQRSTUVWXYZABC
.....
ZABCDEFGHIJKLMNOPQRSTUVWXY

Chiave: CIAO,
testo in chiaro : DOMANI NON POSSO
Cifratura:

C I A O C I A O C I A O C I
D O M A N I N O N P O S S O
F W M O P Q N C P X O G U W

I due corrispondenti preparano una tavola con le 26 possibili rotazioni dell'alfabeto e scelgono come chiave una parola qualsiasi, che imparano a memoria.

Per cifrare, la sorgente scrive ripetutamente la chiave sopra il testo in chiaro ed inserisce poi ordinatamente nel testo cifrato il carattere della tabella posto nella colonna che inizia con il carattere di testo e nella riga che inizia con il carattere di chiave.

Ogni carattere dell'alfabeto può dunque essere sostituito in 26 modi diversi.

Per decifrare, la destinazione scrive la chiave ripetutamente sopra al testo cifrato; per rimettere in chiaro un carattere, deve scegliere la riga della tabella che inizia con il carattere di chiave corrispondente, individuare la colonna in cui è contenuto e prenderne il carattere iniziale.

2 - Codifica in C[25]

Cifratura

Il parametro *chiave* è un puntatore ad un buffer in cui è contenuta la parola chiave
La struttura dati *_AlfabetoScelto* contiene i simboli dell'alfabeto ordinati in un vettore *carattere* di dimensione *dimensione_buffer_in*.

Il contatore del puntatore alla chiave viene incrementato solo se il carattere da codificare appartiene all'alfabeto scelto; una volta raggiunto il valore della lunghezza della chiave viene azzerato, in modo da rileggere il primo carattere; perciò all'interno di una sequenza di testo in chiaro di lunghezza DIMBUF, quella cioè passata a questo algoritmo dal resto dell'applicazione, la chiave è correttamente allineata.
Per fare in modo che ci sia allineamento anche oltre si è utilizzata la variabile globale index_chiave

```
extern int index_chiave;
void CodificaVigenere(int chiave, char *buffer_in,
int dimensione_buffer_in, char *buffer_out,
int dimensione_chiave)
{
        int index;
        char* ptr_chiave;
        (int)ptr_chiave= chiave;
        memset(buffer_out, 0, dimensione_buffer_in);

        for ( index=0 ; index < dimensione_buffer_in ; index++ )
                if(alfabeto_attivo[buffer_in[index]].proprieta & 1)
                {
buffer_out[index]= AlfabetoSceltoAttivo->carattere[
((alfabeto_attivo[ptr_chiave[index_chiave]].proprieta >> 24)+
        (alfabeto_attivo[buffer_in[index]].proprieta >> 24) ) %
AlfabetoSceltoAttivo->dimensione  ];
                        index_chiave++;
                        if(index_chiave >= dimensione_chiave)
                                index_chiave = 0;
                }
                else buffer_out[index]= buffer_in[index];
}
```

Decifrazione

La logica è la stessa che per l'algoritmo di codifica, come pure le valutazioni sull'allineamento della chiave.

```
void DecodificaVigenere(int chiave, dimensione_buffer_in,
dimensione_chiave;
char *buffer_in, *buffer_out)
{
        int index;
        char* ptr_chiave;
        (int)ptr_chiave= chiave;
        memset(buffer_out, 0, dimensione_buffer_in);
        for ( index=0 ; index < dimensione_buffer_in ; index++ )
                if(alfabeto_attivo[buffer_in[index]].proprieta & 1)
                {
buffer_out[index]=AlfabetoSceltoAttivo->carattere[
( AlfabetoSceltoAttivo->dimensione -
(alfabeto_attivo[ptr_chiave[index_chiave]].proprieta >> 24) +
```

[25] I sorgenti presentati nel seguito sono tratti dalla tesi di laurea di Marco Remondini (1999). Il testo della tesi, tutti i codici sorgenti e l'eseguibile Codec.exe sono disponibili nel sito del Corso.

```
(alfabeto_attivo[buffer_in[index]].proprieta >> 24) ) %
AlfabetoSceltoAttivo->dimensione ];
                        index_chiave++;
                        if(index_chiave >= dimensione_chiave)
                                index_chiave = 0;
                }
                else buffer_out[index]= buffer_in[index];
}
```

3 - Attacco con solo testo cifrato

Uno stesso carattere presente in posizioni diverse del testo in chiaro viene dunque in generale sostituito da caratteri diversi: per più di 200 anni si è ritenuto che ciò rendesse del tutto inutile il calcolo delle frequenze di occorrenza dei caratteri all'interno del crittogramma.

In tutto questo lungo periodo di tempo non si è dato peso ad un aspetto che rende di nuovo efficace l'attacco statistico: se la chiave è lunga **L**, caratteri del cifrato distanti **L** uno dall'altro discendono dall'uso di una stessa sostituzione monoalfabetica.

L'individuazione preliminare di **L** accomuna i due metodi[26] che hanno consentito di rompere il Cifrario.

4 - Il test di Kasiski (1863)

Kasiski, un ufficiale prussiano, partì dall'osservazione che due poligrammi identici, presenti nel testo in chiaro ad una distanza uno dall'altro pari ad un multiplo della lunghezza della chiave, debbono necessariamente essere sostituiti da poligrammi identici nel testo cifrato. Per individuare con buona probabilità la lunghezza della chiave basta dunque prendere nota di quanto distano tra loro, nel testo cifrato, sequenze identiche di tre o più simboli, fattorizzare i valori trovati e prendere in considerazione solo quelli che mostrano un fattore comune: tale massimo comun divisore è molto probabilmente la lunghezza della chiave segreta od un suo sottomultiplo.

5 - La formula ed il test di Friedman (1925)

Friedman[27], un colonnello dell'esercito americano, partì invece dalla **probabilità** di trovare due caratteri identici, scegliendo a caso la loro posizione all'interno in una stringa contenente **n** caratteri dell'alfabeto inglese.

Tale probabilità, detta **indice di coincidenza** e nel seguito indicata con **I**, può essere stimata facendo il rapporto tra il numero dei casi favorevoli e quello dei casi possibili.

I casi possibili sono **n × (n-1)/2**, cioè tutte le coppie di caratteri in posizione diversa.

Per valutare i casi favorevoli indichiamo con n(1), n(2),....n(26) le occorrenze di ciascun carattere nella stringa: le coppie formate da due caratteri uguali a **i** sono **n(i) × [n(i) – 1]/2** e la somma per ogni **i** fornisce quindi il numero complessivo di casi favorevoli. Si ha dunque:

$$I = \frac{\sum_{i=1}^{26} n(i)\left[n(i)-1\right]}{n(n-1)}$$

Consideriamo ora il caso in cui la stringa è un testo in lingua inglese ed indichiamo con p(1), p(2),....p(26) le probabilità di occorrenza dei 26 caratteri.

Con questa premessa possiamo calcolare un indice di coincidenza **rappresentativo di qualsiasi testo** osservando che la probabilità di trovare una coppia di simboli uguali a **i** è $\mathbf{p(i)^2}$. Il valore atteso dell'**indice di coincidenza del linguaggio** inglese (nel seguito lo indicheremo con I_l) è dunque espresso da:

$$I_l = \sum_{i=1}^{26} p(i)^2$$

ESEMPI – L'inglese ha 0,0667 come indice di coincidenza, l'italiano 0,0738, il russo 0,0529.

La formula precedente consente di valutare anche l'indice di coincidenza I_c **di una qualsiasi stringa di simboli scelti a caso**. Se vale l'ipotesi di equiprobabilità dei simboli (**p(i) = 1/26** per ogni **i),** si ottiene:

$$I_c = \sum_{i=1}^{26} 1/26^2 = 0{,}0384$$

Ritorniamo ora al caso che interessa la crittanalisi: la stringa di cui si calcola l'indice di coincidenza (lo chiameremo indice di coincidenza **effettivo** e lo indicheremo con ***I****) è un testo cifrato e deve essere molto lungo.

[26] Charles Babbage, pur non avendo formalizzato un metodo preciso, è stato il primo ad accorgersi del punto debole della polialfabetica.

[27] Friedman è noto anche per aver scritto il primo libro di Crittanalisi e per aver contributo alla rottura del Cifrario usato dai giapponesi durante la seconda guerra mondiale, evento che venne stranamente reso di pubblico dominio dalla stampa americana. Ancora più stranamente gli unici a non crederci furono i giapponesi, che continuarono ad usare il loro Cifrario.

Se il testo cifrato è stato ottenuto con una sostituzione monoalfabetica, o con una trasposizione, ci si deve aspettare che ***I**** sia uguale a I_l.

Se invece è stato ottenuto con una sostituzione polialfabetica ci si deve aspettare che il valore di I* sia minore di I_l e maggiore di I_C .

L'obiettivo della polialfabetica, come abbiamo detto in precedenza, è proprio quello di generare una distribuzione "piatta" delle probabilità d'occorrenza dei simboli del cifrato ed è tanto meglio conseguito, quanto più grande è la lunghezza L della chiave.

Consideriamo una certa stringa di testo cifrato intercettato, ipotizzando che sia formata da **n** caratteri, che i caratteri possibili siano quelli della lingua inglese e che sia stata ottenuta impiegando il Cifrario di Vigenère con una chiave casuale di lunghezza **L** (per semplicità supponiamo anche che n sia un multiplo intero di L).

Vogliamo valutare la probabilità **p** che due simboli scelti a caso all'interno della stringa siano distanti uno dall'altro un multiplo intero della lunghezza della chiave e siano stati quindi ottenuti dalla stessa sostituzione monoalfabetica.

Le possibili coppie di simboli sono **½ n(n-1)**.

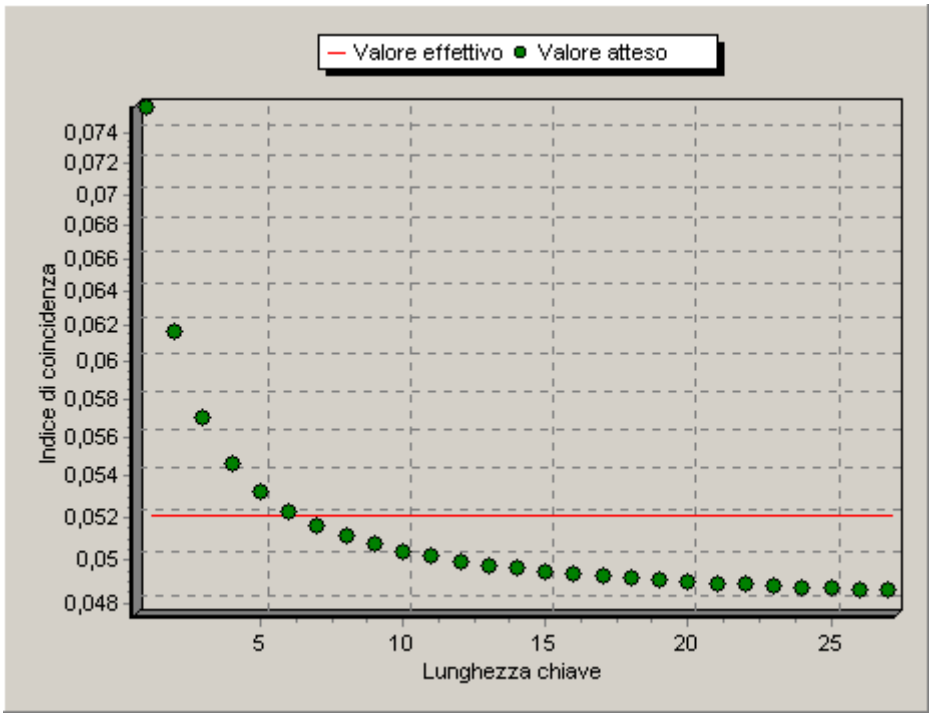
Per valutare i casi favorevoli, suddividiamo idealmente la stringa in **n/L** blocchi contenenti ciascuno **L** simboli. Le possibili coppie di blocchi sono ½ n/L(n/L-1) ed all'interno della coppia ci sono L simboli che distano un multiplo intero di L: i casi favorevoli sono dunque **½ n(n/L-1)**.

Facendo il rapporto tra i casi favorevoli ed i casi possibili si ha dunque:

$$p = \frac{1}{L} \times \frac{(n-L)}{(n-1)}$$

Il valore atteso dell'indice di coincidenza della stringa risulta espresso dalla formula:

$$E(I) = p \times I_l + (1-p) \times I_c$$



In figura è riportato l'andamento di *E(I)* al variare di L per testi cifrati ottenuti da testi in chiaro in lingua italiana. Si noti che per L=1 (sostituzione monoalfabetica) il valore di E(I) è circa uguale all'indice di coincidenza dell'italiano e che per L>25 tende a 0,0476, cioè a 1/21.

In figura è riportato anche, tramite una retta orizzontale, il valore dell'indice di coincidenza effettivo di una certa stringa, molto lunga, di caratteri cifrati.

I valori di lunghezza di chiave posti in un intorno dell'intersezione delle due curve sono i più probabili (nel caso specifico la stringa era stata ottenuta con L = 5); l'intervallo da prendere in considerazione è piccolo solo se l'intersezione avviene prima del tratto asintotico, cioè se la lunghezza della chiave di cifratura è molto piccola.

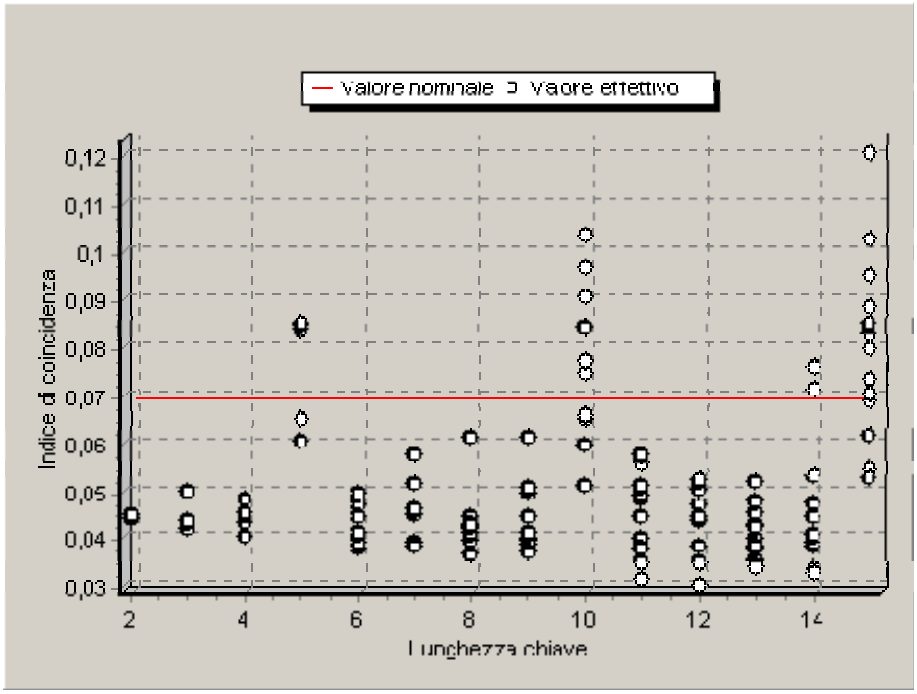
In questo caso il valore di L può essere ottenuto anche analiticamente (**formula di Friedman**). Una volta posto **I* = *E*(I)** si ottiene:

$$\mathbf{L = (I_l - I_c)/(I^* - I_c + (I_l - I^*)/n)}$$

In generale l'intervallo è grande ed i valori di chiave devono essere sottoposti al **test di Friedman**.

Tale test prevede di suddividere la stringa da crittanalizzare in **L** sottoinsiemi contenenti ciascuno **n/L** simboli posti ad una distanza multipla di **L** uno dall'altro e di calcolare poi l'indice di coincidenza **I*** di ciascun sottoinsieme.

Se **L** è proprio la lunghezza della chiave, ognuno di questi insiemi contiene simboli ottenuti con una sostituzione monoalfabetica e ci si deve quindi aspettare $\mathbf{I^* = I_l}$.



In figura ci si riferisce alla stringa considerata in precedenza: i "pallini" indicano i valori effettivi dell'indice di coincidenza **I*** dei sottoinsiemi al variare del valore di L all'interno dell'intervallo che si vuole analizzare.

Ai fini dell'individuazione della vera lunghezza di chiave si devono considerare solo i valori che hanno prodotto indici di coincidenza distribuiti attorno alla retta orizzontale corrispondente a I_l.

Si noti che valori di **I*** centrati attorno a I_l sono stati ottenuti solo per **L** uguale a 5, 10 e 15 (la vera lunghezza e due suoi multipli).

È naturale che per multipli crescenti della vera lunghezza della chiave i valori di **I*** tendano a sparpagliarsi sempre di più, dato che il numero di simboli su cui è calcolato cala sempre più. Bisogna dunque assumere per buona la lunghezza di chiave che determina il minimo scarto quadratico tra indice effettivo ed indice nominale.

6 - Codifica in C del test di Friedman

Calcolo di I_l - Il vettore *frequenze* contiene le frequenze relative *a priori* dei simboli del linguaggio

```
IL= 0;
for( index=0 ; index< dimensione ; index++)
        IL+= frequenze[index]*frequenze[index];
```

Calcolo di I*. Dopo avere creato il vettore delle frequenze ed avere verificato l'esistenza del file, vengono calcolate le frequenze relative dei vari simboli che compaiono nel testo cifrato.

Calcolo dell'indice di coincidenza effettivo.

```
float IndiceCoincidenzaEffettivo(char *NomeFile)
{
        char ch;
        int index;
        float I;
        float M;
        int *frequenze;
        FILE *f;
        frequenze= new int[DIMFREQ];
        for( index=0 ; index<DIMFREQ ; index++)
                frequenze[index]= 0;
        f= fopen(NomeFile, "rt");
        if(f == NULL)
                return -1;
        for( index=0 ; ; index++)
                if(fread(&ch, 1, 1, f))
                        frequenze[(char)ch]++;
                else break;
        M= DimensioneTesto(NomeFile);
        I=0;
        for( index=0 ; index<DIMFREQ ; index++)
                I+= frequenze[index]*(frequenze[index]-1);
        I/= M*(M-1);
        delete[] frequenze;
        fclose(f);
        return I;
}
```

Individuazione della lunghezza della chiave. Alla funzione vengono passati due parametri: il nome del file da elaborare e l'intervallo dei valori di lunghezza della chiave da prendere in considerazione.

```
float ICconLunghezzaChiave(char *NomeFile,
int LunghezzaChiave)
{
        int index;
        int Offset;
        char ch;
        float I;
        FILE *f;
        struct _frequenze {
                int frequenza[DIMFREQ];
                int M;
        } *frequenze;
        f= fopen(NomeFile, "rt");
        if(f == NULL)
                return -1;
        frequenze= new struct _frequenze[LunghezzaChiave];
        for(Offset= 0 ; Offset<LunghezzaChiave ; Offset++)
                for( index=0 ; index<DIMFREQ ; index++)
                {
                        frequenze[Offset].frequenza[index]=0;
                        frequenze[Offset].M= 0;
                }
        Offset= 0;
        for( ; ; )
                if(fread(&ch, 1, 1, f))
                {
                        frequenze[Offset].frequenza[(char)ch]++;
                        frequenze[Offset].M++;
                        Offset++;
                        if( Offset == LunghezzaChiave)
                                Offset= 0;
                }
                else break;
        for(Offset= 0 ; Offset<LunghezzaChiave ; Offset++)
        {
                I =0;
                for( index=0 ; index<DIMFREQ ; index++)
                        I+= frequenze[Offset].frequenza[index] *
(frequenze[Offset].frequenza[index]-1);
                if( frequenze[Offset].M > 1)
                {
                        I/= frequenze[Offset].M;
                        I/= (frequenze[Offset].M-1);
                }
                else I= 0;
                GraficoLCForm->Chart1->SeriesList->Series[1]->
AddXY(LunghezzaChiave, I, "",  clTeeColor);
        }
        delete[] frequenze;
        fclose(f);
        return 0;
}
```

Produzione del grafico riprodotto a pag. 53

Calcolo della lunghezza di chiave che determina il minimo scarto quadratico tra I_l e I*.

```
int TGraficoLCForm::CalcolaSQMmigliore()
{
      int index;
      int index2;
      int DimListaSQM;
      int LKcorrente;
      float *ListaSQM;
      float somma;
      float SQMminimo;
      float LKmigliore;

      DimListaSQM= Edit2->Text.ToInt() - Edit1->Text.ToInt() + 1;
      ListaSQM= new float[DimListaSQM];
      LKcorrente= Chart1->SeriesList->Series[1]->XValues->Value[0];
      index2= 0;
      somma= 0;
      for( index=0 ; ; index++ )
      {
      if(index >= Chart1->SeriesList->Series[1]->XValues->Count() )
      {
                              ListaSQM[index2]=
sqrt(somma/LKcorrente);
                              break;
      }
      if( Chart1->SeriesList->Series[1]->XValues->Value[index] !=
LKcorrente )
      {
      ListaSQM[index2]= sqrt(somma/LKcorrente);
      LKcorrente= Chart1->SeriesList->Series[1]->XValues-
>Value[index];
      index2++;
      somma= 0;
      }
      somma+= pow(Chart1->SeriesList->Series[1]->YValues-
>Value[index]-IClinguaggioAnalisi, 2);
      }
      SQMminimo= 9999999999999;
      for(index=0 ; index < DimListaSQM ; index++);
      if( ListaSQM[index] < SQMminimo)
      {
      SQMminimo= ListaSQM[index];
      LKmigliore= Edit1->Text.ToInt()+index;
}
    return LKmigliore;
    }
```

7 - Individuazione della chiave e del testo in chiaro

Una volta noto L, si può risalire alla parola chiave in tre modi (i primi due sono utili solo per testi brevi):

- confrontando il simbolo più frequente dell'alfabeto in chiaro con quello più frequente del sottoinsieme del testo cifrato preso in esame;
- confrontando la distribuzione di tutti i simboli dei due alfabeti;
- utilizzando l'indice mutuo di coincidenza (v. tesi).

Ottenuta un'**ipotesi** di chiave, si deve testarla

1. decifrando il testo cifrato
2. valutandone o il significato del testo in chiaro così ottenuto (cosa che richiede l'uomo), o l'ortografia e la grammatica (cosa che può fare la macchina da sola).

8 - Attacco con testo in chiaro noto

L'attacco alla polialfabetica **con testo in chiaro noto** è ancora più facile da fare e risulta efficace anche nel caso di chiavi lunghe.

Per dimostrarlo è utile una formulazione matematica del Cifrario di Vigenère.

Sia **L** la lunghezza della chiave e **k(1), k(2), .. k(i), .. k(L)** gli interi corrispondenti ai caratteri che la compongono (0=A, 1=B, ecc.).

Dopo aver suddiviso il testo in chiaro in blocchi di **L** caratteri ed aver sostituito a ciascuno carattere l'intero **m(i)** corrispondente, la cifratura di un blocco è espressa dalla formula:

$$c(i) = (m(i) + k(i)) \bmod 26 \text{ per } 1 \le i \le L$$

Per la decifrazione si ha:

$$m(i) = (c(i) + 26 - k(i)) \bmod 26 \text{ per } 1 \le i \le L.$$

Dalle formule precedenti discende immediatamente che

$$k(i) = (c(i) + 26 - m(i)) \bmod 26.$$

Se si conoscono tutti i **c(i)** ed i corrispondenti **m(i)** non c'è dunque problema a conoscere anche la chiave. Se non si conosce tutto il testo in chiaro o se si ha in mano soltanto un'ipotesi, l'attacco è un po' più brigoso, ma il successo è assicurato.

ESEMPI – ***Attacco con una testo cifrato ed il corrispondente testo in chiaro -*** Siano **C** e **P** una coppia testo cifrato/testo in chiaro di lunghezza **l < L** nota all'intercettatore. Il calcolo carattere per carattere di

$$(c(i) + 26 - m(i)) \bmod 26$$

consente di determinare agevolmente una stringa di simboli di chiave di eguale lunghezza.

Il frammento di chiave così individuato mette in chiaro tutti i frammenti di testo che distano multipli di L, in avanti e all'indietro, da quello inizialmente preso in considerazione; sfruttando la ridondanza del testo in chiaro è poi anche facile aumentare progressivamente la conoscenza sulla chiave.

Attacco con parola probabile - Le cose sono appena più complicate se si ritiene solo probabile che il testo cifrato contenga in qualche punto una certa parola **P***.

In questo caso occorre prendere via via in considerazione stringhe **C*** del testo cifrato di lunghezza pari a **P***. Operando su una coppia come indicato in precedenza si ottiene un frammento **K*** di *ipotesi di chiave*: per avere una conferma sulla validità dell'ipotesi si sfrutta la periodicità della regola di cifratura, andando a controllare se K* consente di mettere in chiaro altre parti del testo cifrato.

9 - Accorgimenti difensivi

Diversi accorgimenti possono rendere più robusta una sostituzione polialfabetica.

Il primo, già enunciato in R12, è che la chiave sia lunga e scelta a caso, ma nel periodo della Crittografia classica gli utenti dovevano poi ricordarsela e per questo motivo non è mai stato applicato bene.

Il secondo, più semplice da fare, interessa anche la Crittografia moderna:

- R25: "*per rendere più difficile la vita all'intruso che vorrebbe condurre analisi con testo noto, non bisogna mai archiviare insieme i testi cifrati ricevuti e le loro decifrazioni; la cosa più sicura è non archiviare affatto i testi in chiaro*"

Il terzo accorgimento riguarda il controllo d'accesso ed è sempre di attualità:

- R26: "*per impedire analisi con testo scelto, non bisogna mai lasciare incustodite macchine pronte per cifrare e decifrare".*

Un quarto accorgimento riguarda i Crittografi di allora e di oggi:

- R27: "*in un Cifrario robusto ogni simbolo del testo in chiaro deve influire sul valore di tutti i simboli del cifrato*".

La Crittografia classica ha tentato di eliminare la pericolosa forma di corrispondenza topologica tra i simboli in chiaro e cifrati della polialfabetica facendola seguire da un'altra trasformazione.

ESEMPIO - Durante la I^{a} Guerra Mondiale si ricorse alla **sopracifratura**, cioè all'uso di un Cifrario a trasposizione disposto a valle di quello a sostituzione polialfabetica: l'intrinseca debolezza delle due operazioni rese però solo appena un po' più laborioso il lavoro del crittanalista.

3.3.4 Il Cifrario di Vernam-Mouborgne e One-time pad

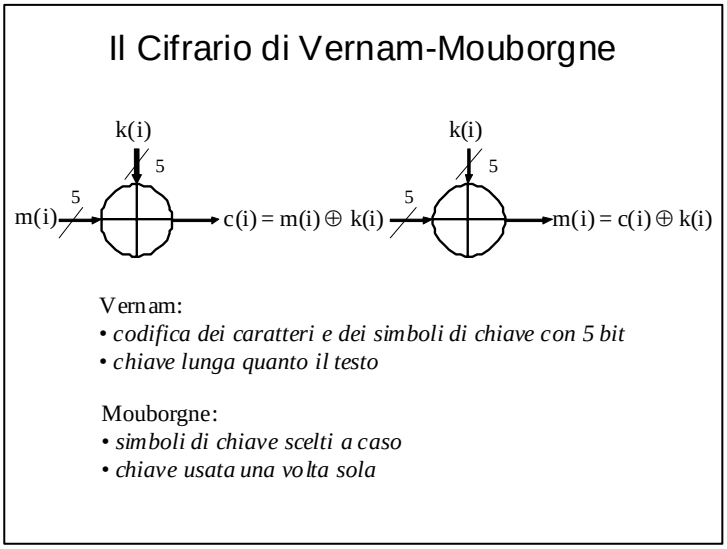
Il Cifrario di Vernam (1917), concepito per il telegrafo, operava una **sostituzione polialfabetica** di caratteri codificati con cinque bit (codice di Baudot) impiegando una chiave **lunga quanto il testo in chiaro**.

I caratteri di chiave erano ottenuti da un nastro che avanzava automaticamente di una posizione dopo la cifratura/decifrazione di un carattere di testo.

Le trasformazioni eseguite dalla sorgente e dalla destinazione si basavano sulla lettura di una tabella: per comodità ci limitiamo a tracciarla nel caso di simboli codificati con tre bit.

	Chiave							
Testo	000	001	010	011	100	101	110	111
000	000	001	010	011	100	101	110	111
001	001	000	011	010	101	100	111	110
010	010	011	000	001	110	111	100	101
011	011	010	001	000	111	110	101	100
100	100	101	110	111	000	001	010	011
101	101	100	110	111	001	000	011	010
110	110	111	100	101	010	011	000	001
111	111	110	101	100	011	010	001	000

Si noti che le righe della tabella contengono otto diverse permutazioni dei numeri di tre bit: ci troviamo dunque di fronte ad una sostituzione con otto alfabeti e le modalità d'impiego della tabella sono identiche a quelle di una tavola di Vigenère.



Con l'attuale tecnologia la tabella in trasmissione può essere sostituita dal calcolo dell'operazione di **somma modulo due** tra ogni bit di testo in chiaro ed il corrispondente bit di chiave:

c(i) = m(i) ⊕ k(i) per i = 1, 2, .., 5

La stessa cosa si può fare in ricezione. Si ha, infatti:

c(i) ⊕ k(i) = (m(i) ⊕ k(i)) ⊕ k(i) = m(i)

La rappresentazione schematica dei meccanismi di cifratura e di decifrazione è illustrata in figura.

Alcuni anni dopo J. Mouborgne (U.S. Army) intuisce che la massima sicurezza possibile si ottiene solo se la chiave:

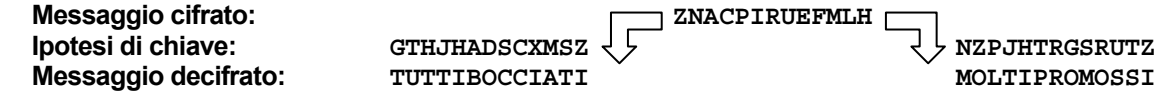
1. è formata da **simboli scelti a caso**,
2. è usata **una volta sola**.

La necessità di simboli di chiave casuali merita una breve discussione.

ESEMPIO - Il Cifrario di Vigenère è stato impiegato anche con chiavi lunghe quanto il testo ed usate una volta sola.

A tal fine i simboli di chiave erano progressivamente prelevati da un testo voluminoso a disposizione di entrambi i corrispondenti (metodo detto della ***running key***). La robustezza non è però aumentata: se, infatti, la chiave è una frase del linguaggio naturale del testo in chiaro, è ancora possibile "decrittare" sfruttando le statistiche dei caratteri (si devono cercare le situazioni in cui una lettera di chiave ha cifrato la stessa lettera di testo).

Il Cifrario di Vernam-Mouborgne, in seguito denominato *one-time pad*, non può essere violato con attacchi passivi. Questa importante proprietà sarà dimostrata nel prossimo paragrafo; per ora limitiamoci a verificarla osservando cosa succede se uno prova a decifrare un crittogramma con due diverse ipotesi di chiave:



Lo stesso cifrato può dunque fornire testi in chiaro di significato diverso e l'intruso non ha appigli per decidere se è buono il primo, o il secondo, o uno qualsiasi degli altri ottenibili con chiavi diverse.

Il modo di operare di questo Cifrario può però sollevare qualche perplessità: **per trasmettere un messaggio riservato su un canale insicuro è necessario impiegarne precedentemente uno sicuro per concordare un dato, la chiave, altrettanto lungo e segreto!**

Esistono inoltre notevoli oneri di gestione.

- I simboli di chiave devono essere generati con un PRNG sicuro.
- Per motivi di efficienza, la stringa di simboli di chiave concordata dai due utilizzatori deve avere una lunghezza sufficiente per diversi messaggi, cosa che ne rende poi pesante la memorizzazione.
- Il testo cifrato deve inoltre arrivare **tutto** ed **in ordine**.

One-time pad è quindi stato usato solo in situazioni particolari.

ESEMPI - **Seconda guerra mondiale -** I Servizi Segreti inglesi, che erano riusciti a rompere il cifrario di Enigma, impiegavano *one-time pad* (la denominazione è stata coniata in quel periodo) per trasmettere in modo sicuro da Bletchley Park a Londra la decifrazione dei messaggi degli U-Boat.
Guerra fredda tra Russia e Stati Uniti - Le spie russe operanti in territorio americano usavano one-time pad per non essere accusate di spionaggio anche se trovate in possesso di un testo cifrato ancora da inoltrare: una volta distrutta la porzione di chiave usata, non c'era, infatti, alcuna possibilità di dimostrare che il contenuto del messaggio fosse illecito. I Capi di Stato di Russia e Stati Uniti lo hanno usato per difendere la riservatezza delle loro comunicazioni sulla *linea calda*; corrieri diplomatici garantivano la disponibilità di sempre nuovi simboli di chiave.

Si noti infine che l'**inviolabilità** di OTP ad attacchi passivi non implica **robustezza** agli attacchi attivi: per ottenere questa proprietà occorre impiegare anche **meccanismi di autenticazione**.

ESEMPIO - Si consideri una missiva, codificata in ASCII, che porta al termine l'indicazione di chi l'ha scritta.
Chi ne intercetta il testo cifrato con *one-time pad* non può sicuramente metterlo in chiaro, ma non incontra alcuna difficoltà a modificare la firma, se la conosce. A tal fine basta, infatti, che la codifichi in ASCII, ne sommi modulo due i bit con la parte finale del messaggio ed utilizzi poi il frammento di chiave risultante per introdurre nel cifrato un nuovo nominativo di pari lunghezza.

3.4 Sicurezza di un Cifrario e Teoria dell'informazione

Si è detto in precedenza che il confine tra Crittologia classica e moderna è segnato dall'avvento dei calcolatori elettronici. In realtà il passaggio non è stato solo tecnologico: di fondamentale importanza teorica sono stati, infatti, gli studi di C.E. Shannon sulla sicurezza della comunicazione di un'informazione riservata[28].

L'ipotesi di partenza è che alle estremità del canale insicuro sia impiegato un Cifrario simmetrico con l'obiettivo di trasmettere testi cifrati che, se intercettati, non consentano di risalire ai corrispondenti testi in chiaro.

Nei paragrafi 3.4.1 e 3.4.2 vedremo come Shannon, impiegando un modello probabilistico del canale, sia riuscito a dimostrare l'esistenza del **Cifrario perfetto**, cioè atto a garantire l'assoluta impossibilità di decrittazione.

Nel paragrafo 3.4.3 riassumeremo le indicazioni date da Shannon per costruire un **Cifrario computazionalmente sicuro** a partire dalle classiche ed insicure trasformazioni di sostituzione e di trasposizione.

3.4.1 Elementi di Teoria dell'informazione

Variabile aleatoria X	Variabile **X** che assume valori sull'insieme di numeri $\{x_1, x_2, \ldots, x_n\}$, con probabilità $p(X = x_i) = p(x_i)$, ove $0 \le p(x_i) \le 1$ e $\sum_{i=1}^{n} p(x_i) = 1$
Informazione $I(x_i)$ fornita dall'evento $X = x_i$	La quantità d'informazione fornita dall'evento $X = x_i$ è per definizione $I(x_i) = \log_2 (1/ p(x_i)) = - \log_2 p(x_i)$
NOTE	- L'informazione trasportata da un evento che sicuramente si verifica è 0, quella di un messaggio che ha il 50% di probabilità di verificarsi è 1, quella di un messaggio cha ha il 25% di probabilità è 2. - Come unità di misura della quantità d'informazione è stato inizialmente impiegato il **bit**, pur con l'incongruenza di doverlo associare anche a valori non interi; oggi si usa il **shannon** (sigla **sh**), per onorare chi per primo ha introdotto questa misura. - Per esprimere la quantità di informazione fornita da una realizzazione di X si può impiegare la **funzione logaritmo** con base diversa da 2 (ad es. base 10, o e).

[28] Si veda ad esempio F. Fabris "Teoria dell'informazione, codici, cifrari", Bollati Boringhieri 2001 Quanto segue è tratto da [3], pp.75-84

Entropia H(X)	**Informazione media** trasportata da un valore di **X**. Per definizione di **valore atteso** di una variabile aleatoria si ha: $\mathbf{H(X) = E(I(X)) =} \sum_{i=1}^{n} p(x_i) \times I(x_i) = -\sum_{i=1}^{n} p(x_i) \times \log p(x_i)$
NOTE	- L'entropia misura anche l'**incertezza** che si ha sul valore di **X** prima che si verifichi. - L'entropia indica inoltre il numero atteso di bit richiesti per codificare i valori di **X**. - L'entropia è limitata inferiormente da **0** (cosa che capita quando un evento ha probabilità 1 e tutti gli altri 0) e superiormente da $\mathbf{\log_2 n}$ (cosa che capita quando tutti gli eventi sono equiprobabili): $\mathbf{0 \le H(M) \le \log_2 n}$
Coppia di variabili aleatorie X e Y	Variabili per i valori delle quali sono assegnate le probabilità p(x) e p(y), le probabilità condizionate **p(x|y)** e quindi anche le probabilità congiunte **p(x,y)** = p(x|y).p(y)
Entropia condizionata H(X|Y=y)	Incertezza che si ha sul valore di **X** a fronte dell'evento **Y = y**: $\mathbf{H(X|Y=y) =} -\sum_{x} p(x \mid y) \times \log p(x \mid y)$
Entropia condizionata H(X|Y)	Incertezza che si ha sul valore di X noto un qualsiasi valore **y** di **Y**: $\mathbf{H(X|Y) =} -\sum_{y} p(y) \sum_{x} p(x \mid y) \times \log p(x \mid y)$
NOTE	- L'entropia condizionata H(X|Y) è detta anche **equivocazion**e. - Tra l'entropia e l'entropia condizionata vale la disuguaglianza $\mathbf{H(X) \ge H(X|Y)}$
Entropia congiunta H(X,Y)	Per definizione è data da: $\mathbf{H(X,Y) =} -\sum_{y} \sum_{x} p(x,y) \times \log p(x,y)\,.$
NOTE	- Tra l'entropia e l'entropia congiunta vale la disuguaglianza $\mathbf{H(X,Y) \le H(X) + H(Y)}$ - L'eguaglianza si ha se e solo se le due variabili sono indipendenti. - Per l'entropia congiunta vale la proprietà: $\mathbf{H(X,Y) = H(X) + H(Y\|X) = H(Y) + H(X\|Y)}$ **-** Data anche una variabile aleatoria **Z** si ha: $\mathbf{H(X,Y|Z) = H(X|Z) + H(Y|X,Z)\ ) = H(Y|Z) + H(X|Y,Z)}$
Informazione mutua I(X;Y)	Quantità attesa d'informazione fornita da **Y** su **X.** $\mathbf{I(X;Y) = H(X) + H(Y) - H(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)}$

3.4.2 Sicurezza incondizionata

Ogni sorgente dispone di un certo alfabeto $\mathbf{M = \{m_1, m_2, \ldots, m_n\}}$ da cui sceglie in sequenza i simboli con cui comporre il testo in chiaro da inviare alla destinazione.

Supponiamo che ogni $\mathbf{m_i}$ sia codificato da una stringa di N bit e che sia quindi interpretabile come un intero appartenente all'intervallo $0 \div 2^N-1$. Supponiamo inoltre che la sorgente sia "priva di memoria" e che quindi ogni intero abbia una sua probabilità di essere scelto.

Con queste ipotesi il testo in chiaro è descrivibile con una sequenza di **u** variabili aleatorie con valori interi:

$$\mathbf{M^u = \{M_0, M_1, \ldots, M_{u-1}\}}$$

Prima di essere inoltrato sul canale insicuro il testo in chiaro deve essere cifrato con una chiave **k** scelta all'interno di un insieme di valori **K** su cui è definita la distribuzione di probabilità **p(K = k)**.

Supponiamo che anche il canale sia privo di memoria. Il testo cifrato è descritto dalla stringa

$$\mathbf{C^v = \{C_0, C_1, \ldots, C_{v-1}\}} \text{ con } \mathbf{C^v = E_k\,(M^u)}$$

Per semplificare le precedenti notazioni possiamo omettere gli apici: nel seguito **M** indicherà la variabile aleatoria "testo in chiaro", **K** la variabile aleatoria "chiave" e **C** la variabile aleatoria "testo cifrato".

Vogliamo ora valutare l'incertezza che ha l'intruso sul testo in chiaro e sulla chiave dopo aver intercettato un testo cifrato. L'entropia della coppia **K**, **M** condizionata dalla conoscenza di **C** è:

$$\mathbf{H(K,M|C) = H(K|C) + H(M|K,C) = H(M|C) + H(K|M,C)}$$

Essendo **H(M|K,C) = 0** (dato un cifrato e la chiave non c'è incertezza sul testo in chiaro), si ha:

H(K|C) = H(M|C) + H(K|M,C) ed anche
H(K|C) ≥ H(M|C).

L'attacco che individua la chiave di cifratura è dunque in generale più difficile dell'attacco che individua il testo in chiaro. Si noti che **H(K|M,C**) indica il valore medio dell'incertezza su **K** condizionata dalla conoscenza di **M e** di **C**, cioè esprime la pericolosità dell'**attacco con testo in chiaro noto**.

In un Cifrario robusto tutte e tre queste **equivocazioni** devono avere valore elevato.

Per valutare meglio quanta informazione **C** fornisce su **M**, è utile il concetto di informazione mutua:

I(M;C) = H(M) + H(C) – H(M,C) = H(M) – H(M|C)

Mettendo **H(K)** al posto di **H(M|C) si ottiene**

I(M;C) ≥ H(M) – H(K)

Siamo così arrivati ad una conclusione intuitiva: l'incertezza sulla chiave unitamente all'informazione che il testo cifrato fornisce sul testo in chiaro è maggiore o uguale all'incertezza sul testo in chiaro.

Shannon ha definito **perfettamente sicuro** un Cifrario per il quale si ha **I(M;C) = 0** ed ha dimostrato che

H(K) ≥ H(M)

è condizione necessaria per impedire all'intruso di risalire dal testo cifrato al testo in chiaro, qualsiasi sia la potenza di calcolo di cui dispone. Se la chiave è formata da **n** bit scelti a caso ed in modo indipendente uno dall'altro e se la stessa condizione (è il caso peggiore) vale per gli **u** bit del testo in chiaro, la segretezza perfetta si ottiene con

n ≥ u.

Rimane ancora aperta una questione: **come si costruisce un Cifrario perfetto**?

Per rispondere notiamo che la segretezza perfetta si ha anche con **H(M) = H(M|C)**: le probabilità *a posteriori* che un crittogramma intercettato rappresenti certi testi in chiaro devono essere uguali alle probabilità *a priori* degli stessi testi in chiaro prima dell'intercettazione. In un Cifrario perfetto deve dunque valere l'eguaglianza:

p(m|c) = p(m) per ogni **m** e per ogni **c**.

Per i teoremi di Bayes deve inoltre essere:

p(m) × p(c|m) = p(c) × p(m|c)

- **T1**: "condizione necessaria e sufficiente per la segretezza perfetta è **p(c|m) = p(c)** per ogni **m** e **c**".

In altre parole **p(c|m)** deve essere indipendente da **m**, cosa che si può ottenere se e solo se la probabilità totale di tutte le chiavi che trasformano un messaggio $\mathbf{m_i}$ in un certo crittogramma **c** è uguale a quella di tutte le chiavi che trasformano $\mathbf{m_j}$ nello stesso **c**, per ogni $\mathbf{m_i}$, $\mathbf{m_j}$ e **c**.

Da ciò consegue che il numero di chiavi diverse deve essere almeno pari al numero di messaggi:

- **T2**: "tra le cardinalità degli spazi **M**, **K** di un Cifrario perfetto deve valere la diseguaglianza **|K| ≥ |M|**"

Il caso più semplice si ha quando il numero di chiavi è esattamente uguale al numero di messaggi.

- **T3**: "un Cifrario con **|M| = |K| = |C|** è perfetto se e solo se c'è esattamente una chiave che trasforma ciascun messaggio **m** in ciascun crittogramma **c** e se tutte le chiavi **k** sono **equiprobabili**"

ESEMPIO – Consideriamo l'insieme di messaggi $\{\mathbf{m_1, m_2, m_3, m_4, m_5}\}$, l'insieme di testi cifrati $\{\mathbf{c_1, c_2, c_3, c_4, c_5}\}$ e l'insieme di chiavi **equiprobabili** $\{\mathbf{k_1, k_2, k_3, k_4, k_5}\}$. Un Cifrario perfetto deve trasformare ogni **m** in ogni **c**, cosa che può essere indicata con una matrice le cui righe sono i messaggi, le cui colonne sono i crittogrammi ed in cui ogni incrocio indica la chiave che opera la trasformazione riga-colonna: $\mathbf{c_i = E(m_i , k_k)}$

	$\mathbf{c_1}$	$\mathbf{c_2}$	$\mathbf{c_3}$	$\mathbf{c_4}$	$\mathbf{c_5}$
$\mathbf{m_1}$	$\mathbf{k_1}$	$\mathbf{k_2}$	$\mathbf{k_3}$	$\mathbf{k_4}$	$\mathbf{k_5}$
$\mathbf{m_2}$	$\mathbf{k_5}$	$\mathbf{k_1}$	$\mathbf{k_2}$	$\mathbf{k_3}$	$\mathbf{k_4}$
$\mathbf{m_3}$	$\mathbf{k_4}$	$\mathbf{k_5}$	$\mathbf{k_1}$	$\mathbf{k_2}$	$\mathbf{k_3}$
$\mathbf{m_4}$	$\mathbf{k_3}$	$\mathbf{k_4}$	$\mathbf{k_5}$	$\mathbf{k_1}$	$\mathbf{k_2}$
$\mathbf{m_5}$	$\mathbf{k_2}$	$\mathbf{k_3}$	$\mathbf{k_4}$	$\mathbf{k_5}$	$\mathbf{k_1}$

La matrice sopraindicata è una di tante possibili e costituisce un esempio di "quadrato latino", una struttura il cui studio ha appassionato Eulero: in questo caso si ha **i = (j + k – 1) mod 5**.

Si noti che essendo **p(k) = 1/5** si ha anche **p(m|c) = p(c) = 1/5**.

One-time pad, se impiegato correttamente, ha tutte queste proprietà ed è quindi il più semplice dei Cifrari con perfetta segretezza. E' dunque l'unico Cifrario da impiegare quando si vuole difendere la riservatezza di un messaggio? Ci si può fidare di chi afferma di averne fatto una versione più semplice e parimenti sicura?

Questa risposta è di B. Schneier (CRYPTO-GRAM, October 2002)
"…*One-time pads are useless for all but very specialized applications, primarily historical and non-computer. And almost any system that uses a one-time pad is insecure. It will claim to use a one-time pad, but actually use a two-time pad (oops). Or it will claims to use a one-time pad, but actually use a stream cipher. Or it will use a one-time pad, but won't deal with message re-synchronization and re-transmission attacks. Or it will ignore message authentication, and be susceptible to bit-flipping attacks and the like. Or it will fall prey to keystream reuse attacks. Etc., etc., etc. One-time pads may be theoretically secure, but they are not secure in a practical sense. They replace a cryptographic problem that we know a lot about solving -- how to design secure algorithms -- with an implementation problem we have very little hope of solving. They're not the future. And you should look at anyone who says otherwise with deep and profound suspicion.*"

3.4.3 Sicurezza computazionale

Date le difficoltà di costruzione e d'uso del Cifrario perfetto, ci si deve in pratica accontentare di difendere la riservatezza con Cifrari simmetrici dotati di una chiave di lunghezza fissa e relativamente piccola.

Con questo meccanismo esistono sicuramente correlazioni tra M, K e C: deve essere quindi cura del progettista attribuire alle incertezze H(M|C), H(K|C) e H(K|M,C) valori elevati. Shannon ha quantificato questa esigenza introducendo il concetto di sicurezza computazionale.

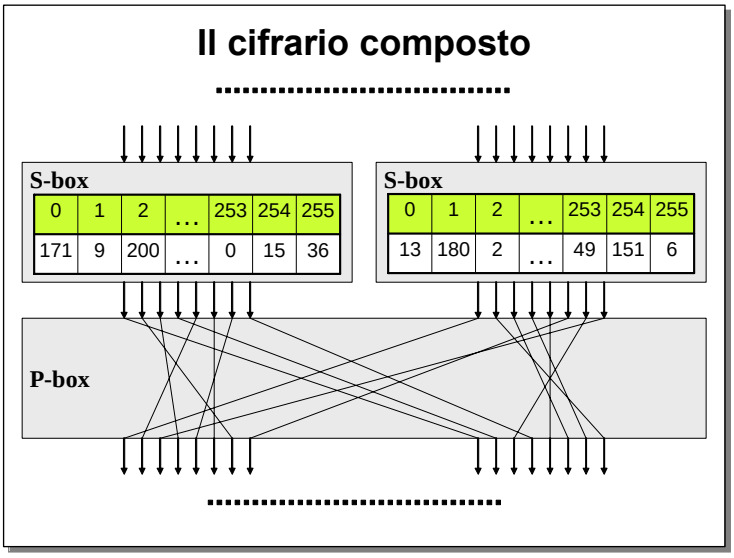
- **Sicurezza computazionale** – Un Cifrario è detto **computazionalmente sicuro** se il calcolare **m** da **c** è possibile, ma richiede una potenza di elaborazione superiore a quella che si ipotizza essere a disposizione dell'attaccante.

Per dare sicurezza computazionale ad un Cifrario non perfetto è necessario che la trasformazione di cifratura generi **confusione** e **diffusione**.

La **confusione** si ottiene imponendo al testo cifrato di dipendere in modo complesso dalla chiave e dal testo in chiaro, al punto che è difficile prevedere che cosa accadrà al cifrato modificando anche un solo simbolo del testo in chiaro.

La **diffusione** si ottiene imponendo ad ogni simbolo del testo in chiaro di influire sul valore di molti, se non tutti, i simboli del cifrato, al punto che è difficile prevedere quanti e quali di questi modificano il loro valore se si modifica anche un solo simbolo del testo in chiaro.

Shannon ha infine indicato il modello del Cifrario **composto** (*product cipher*) come linea guida per conseguire tali proprietà: per determinare **confusione** il Cifrario deve avvalersi di operazioni di **sostituzione**; per generare **diffusione**, deve invece avvalersi di operazioni di **trasposizione** (o **permutazione**).



La struttura di un Cifrario composto alterna di conseguenza stadi di sostituzione (detti **S-box**) e di permutazione (detti **P-box**).

In figura, per fissare le idee, è mostrato un ipotetico stadio di sostituzione formato da due S-box, ciascuna con 8 bit d'ingresso e 8 bit d'uscita, seguito da uno stadio di permutazione in cui i 16 segnali cambiano di posizione.

Per rendere imprevedibile all'intruso il risultato della trasformazione, almeno uno dei due stadi deve essere dotato di **chiave**.

Nelle S-box la chiave definisce la tabella che mette in corrispondenza le configurazioni d'ingresso con quelle d'uscita; nelle P-box la chiave definisce come deve essere cambiata la posizione dei bit.

Shannon ha dato anche un'ultima indicazione, recepita poi da tutti i Cifrari simmetrici moderni: per ottenere un'adeguata quantità di confusione e di diffusione è sufficiente prevedere l'uso iterato di una sola struttura **S-P,** a patto che ad ogni ***round*** sia impiegata una chiave diversa.