

M U Z I C A
B I B L I O T E C A

Andrea Sgarro

Crittografia

Tecniche di protezione
dei dati riservati

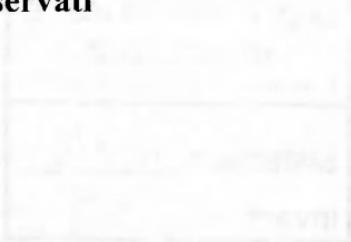


M U Z Z I O B I B L I O T E C A

Andrea Sgarro

Crittografia

Tecniche di protezione dei dati riservati



Franco Muzzio Editore

Andrea Sgarro, triestino, si è laureato in matematica a Trieste, con successiva specializzazione in Calcolo automatico a Pisa. È ordinario di Informatica generale all'università di Trieste. È uno tra i primi studiosi italiani che si siano attivamente dedicati alla nuova crittografia; i suoi risultati in questo campo sono pubblicati su riviste specializzate europee e americane. È membro dell'International Association for Cryptologic Research. Passa parte del suo tempo libero suonando il flauto traverso barocco a una sola chiave.

Nuova edizione: ottobre 1993

ISBN 88-7021-640-3

© 1985, 1993 franco muzzio & C. editore spa
Riviera A. Mussato 39, 35141 Padova
Stampa Legoprint s.r.l., Trento

Tutti i diritti sono riservati

Indice

Presentazione		pag.	VII
Prefazione alla seconda edizione		"	X
1 Cifrari manuali			
1 Messaggi e crittogrammi		"	1
2 Sostituzioni e trasposizioni		"	4
3 Cifrari a sostituzione monoalfabetica		"	6
4 Crittografia tattica e crittografia strategica		"	9
5 Cifrari a sostituzione polialfabetica		"	11
6 Cifrari a trasposizione		"	14
2 Crittanalisi statistica			
1 Struttura statistica dell'italiano scritto		"	19
2 Un esempio di decrittazione semiautomatica		"	23
3 Crittanalisi dei cfrari polialfabetici		"	28
4 Note consuntive sui cfrari manuali		"	34
3 Cifrari a rotore			
1 Macchine per cifrare		"	37
2 Alcune massime del buon crittografo		"	39
4 Il Data Encryption Standard			
1 Cifrari composti		"	41
2 La nuova crittografia		"	43
3 Dati binari, codici e cfrari		"	44
4 Un po' di matematica binaria		"	47
5 L'algoritmo di cifratura e decifrazione del DES		"	49
6 Critiche rivolte al DES		"	60

5 Un cifrario perfetto

1	Un risultato sorprendente	"	63
2	Un modello matematico delle sorgenti d'informazione	"	64
3	Il cifrario a chiave non riutilizzabile	"	67
4	Prestazioni del cifrario a chiave non riutilizzabile	"	69
5	Una digressione sulle aritmetiche circolari	"	72
6	L'alto prezzo della perfezione	"	73
7	Cifre casuali e pseudocasuali	"	75

6 Alla ricerca della perfezione perduta

1	Registri a scorrimento con retroazione lineare	"	77
2	Registri a periodo massimo	"	80
3	Proprietà di casualità	"	83
4	Uso crittografico dei registri lineari	"	85
5	Attacchi crittanalitici con testo in chiaro	"	86
6	Uno sguardo nel buio: i registri non lineari	"	88
7	Cifrari a blocco e cifrari a flusso	"	90

7 Crittografia a chiave pubblica

1	Chiave segreta e chiave pubblica	"	93
2	Le funzioni unidirezionali e il logaritmo finito	"	94
3	Il logaritmo finito e la distribuzione delle chiavi	"	98
4	Il problema del fusto	"	100
5	Il cifrario del fusto	"	103
6	Il teorema di Fermat-Eulero	"	107
7	Generazione di numeri primi elevati	"	109
8	Il cifrario RSA	"	114
9	Autenticazione e firme numeriche	"	118
10	Osservazioni critiche	"	120
11	Prospettive future	"	123

Bibliografia

" 127

Glossarietto crittografico inglese-italiano

" 131

Presentazione

La crittografia è oggi una disciplina in gran voga negli ambienti dell'informatica e delle telecomunicazioni. La ragione è evidente: ormai un numero enorme di messaggi viaggia su ogni sorta di canali, dalla posta tradizionale al telefono, alla radio, al telex e alle linee di trasmissione dei dati ad alta velocità; altrettanto enorme è la quantità di informazioni che viene immagazzinata nelle memorie dei calcolatori e nelle banche di dati. Spesso tali informazioni sono preziose e riservate: è difficile, se non impossibile, proteggerle da orecchi e occhi indiscreti senza ricorrere alle tecniche che la crittografia ci mette a disposizione. Se da un lato il progresso tecnico, specie quello dell'informatica e della microelettronica, agevola l'intercettazione dei dati, dall'altro esso facilita anche la loro protezione crittografica: non fa meraviglia che lo studio e la pratica delle "scritture segrete" (dei cifrari) abbiano avuto di recente un grande impulso.

La crittografia è una scienza antichissima (nelle società primitive qualunque tipo di scrittura è di per sé "magico" e "segreto"): se gli usi in passato sono stati soprattutto di tipo militare, il suo campo d'azione è oggi ben più vasto. Citeremo due esempi emblematici: si pensi da una parte all'esigenza di proteggere la privatezza delle cartelle cliniche, che ormai vengono spesso memorizzate nelle banche di dati degli ospedali, dall'altra parte ai problemi scottanti di trasferimento elettroni-

co del denaro legati all'impiego sempre più diffuso delle carte di credito.

Interessarsi di crittografia, oggigiorno, non è dunque un ozioso passatempo da patiti dell'enigmistica, per quanto neppure l'aspetto ricreativo sia da trascurare, così come non si possono dimenticare i risvolti magici e fantastici su cui ha insistito tanta letteratura (ci limiteremo a ricordare lo *Scarabeo d'oro* di Edgar Allan Poe).

I primi tre capitoli di questo libro sono dedicati alla crittografia "storica": il lettore troverà citati i nomi illustrati di Leon Battista Alberti e di Girolamo Cardano. Non bisogna lasciarsi ingannare; se i cifrari descritti sono di per sé superati, i principi che stanno alla base della loro costruzione sono spesso validi ancor oggi: in effetti il modo migliore per capire la crittografia del ventesimo secolo è proprio quello di compiere all'inizio una breve escursione nel passato e di studiare nella loro essenzialità certe idee di fondo di cui non potremo mai fare a meno. Gli ultimi quattro capitoli sono dedicati alla crittografia contemporanea, sia a quella a chiave segreta, relativamente tradizionale, sia a quella a chiave pubblica, decisamente innovativa. Conclude il volume un glossarietto inglese-italiano dei termini crittografici.

Il livello di preparazione matematica che si richiede per la lettura del libro è stato contenuto al minimo: il lettore più ambizioso dovrà rivolgersi a uno dei numerosi testi disponibili in inglese, per esempio all'ottimo *Cipher Systems. The Protection of Communications* di H. Beker e F. Piper, o agli altri titoli citati nella bibliografia. Sin dal Rinascimento l'Italia ha avuto una gloriosa tradizione crittografica che è stata tenuta viva soprattutto a Venezia ed è giunta fino a epoche recenti: il *Manuale di crittografia* di Luigi Sacco, pubblicato a più riprese fra le due guerre, è considerato un classico del suo genere ed è stato di recente ristampato... negli Stati Uniti. Per rimanere alla scuola del Sacco, facilmente accessibile in Italia è il manuale Hoepli *Crittografia. Le scritture segrete* di Mario Zanotti, stampato nel 1928 e ristampato nel '76 dall'Istituto Editoriale Cisalpino-Goliardica.

PRESENTAZIONE

Il lettore che avesse in antipatia i calcoli, se non farà fatica a scorrere il manuale dello Zanotti, avrà invece qualche fastidio nella lettura del libro che gli sta davanti, anche se gli impedimenti non saranno mai tali da costringerlo a sospendere il cammino. Viceversa i lettori che hanno la passione del calcolatore domestico troveranno nella crittografia numerosi spunti per servirsi del loro strumento in maniera proficua e piacevole.

Posto di fronte a un tale scontro di gusti, l'autore di questo libro rinuncia a prendere posizione, ma non si sottrae al gradito obbligo di ringraziare Mari Allocca e Susanna Crevatin, cui sono dovuti il corredo di programmi e le sperimentazioni al calcolatore che in esso figurano.

EARQD OHZZAUD!

Trieste, 1985

Andrea Sgarro

P.S.: Il lettore che volesse decifrare il crittogramma potrà leggere con profitto le *Vite dei Cesari* di Svetonio, oppure, nel primo capitolo... basta, si è già detto troppo!

Prefazione alla seconda edizione

Dalla prima edizione di questo libro sono passati otto anni, molti per una disciplina scientifica di punta com'è la crittografia, otto anni di febbrale attività di ricerca documentata da migliaia e migliaia di pagine pubblicate, da nuove idee, nuovi brevetti e nuove attuazioni ingegneristiche. Al lettore sembrerà una stravaganza, ma a noi piace cominciare ricordando invece di questo periodo il ritrovamento nell'Archivio Ottomano di Istanbul del *Risalah fi Istikhraj al-Mu'amma* ("Un manoscritto sulla decifrazione dei messaggi crittografici") di al-Kindi, o, per essere più precisi, di Abu Yusuf Ya'qub ibn Is-haq ibn as-Sabbah ibn 'omran ibn Ismail al-Kindi, un testo di mille e cento anni or sono in cui vengono descritte mature tecniche *statistiche* di crittanalisi. La statistica scientifica non sarebbe dunque nata in Europa con le ricerche demografiche di John Graunt (1620-1674) o con quelle attuariali del contemporaneo Johannn De Witt: anch'essa, alla pari dell'algebra, sarebbe una gemma della cultura medievale araba, tramandata poi ai crittografi italiani del Rinascimento come Leon Battista Alberti. Perché abbiamo voluto ricordare un fatto di storia della scienza in un libro che è dedicato ad argomenti di grande attualità? Perché esso ribadisce che la crittografia non è solo, come più o meno tutti sanno, una disciplina di interesse militare, non è solo, come molti cominciano a vedere, un'arma a doppio taglio, che potrà venir usata sia per difendere la nostra privatezza sempre

PREFAZIONE

più minacciata nella società telematica, sia per aggredirla e metterci alla mercè del Grande Fratello di turno, ma è anche, nel senso profondo di questa parola, *cultura*: dopo il “triennio eroico” che va dal 1976, data della pubblicazione del manifesto della nuova crittografia a chiave pubblica firmato da Diffie e Hellman, e il 1978, data di nascita del cifrario di Rivest, Shamir e Adlemann, la cui sicurezza è basata sulla difficoltà di calcolare i fattori primi di un numero intero elevato, la crittografia è sempre più strettamente legata a quel concetto-cardine della cultura scientifica e filosofica contemporanea che è la *complessità*: hanno origine crittografica alcune delle idee più interessanti e profonde che sono emerse di recente in questo campo, come, per fare un esempio, quella delle *dimostrazioni a conoscenza zero* di Goldwasser, Micali e Rackoff (*zero-knowledge proofs*: come dimostrare un teorema senza fornire informazione alcuna sul metodo di dimostrazione). Assai fruttuoso, e altrettanto sorprendente, è stato l'incontro con la più pura e difficile delle discipline matematiche, la teoria dei numeri: un bello sgarbo a quei matematici burbanzosi, sedicenti puristi, per i quali il mondo reale e le sue esigenze applicative sarebbero, per la matematica, solo un incidente di percorso. Su un altro versante, la crittografia quantistica ha fatto appello a idee fondamentali della fisica moderna. Rispetto al 1985, siamo insomma sempre più convinti che contribuire a disseminare la cultura crittografica anche in Italia sia un compito importante: è per questo che abbiamo accolto di buon grado l'invito a curare una seconda edizione riveduta e corretta di questo volume.

Trieste, giugno 1993

Andrea Sgarro

1 Cifrari manuali

1 Messaggi e crittogrammi

Alcuni dei cifrati di cui parleremo in questo capitolo sono spesso adoperati sui banchi di scuola da ingegnosi ragazzetti che vogliono proteggere i loro messaggi riservati (si fa per dire!) dalle intercettazioni di un'arcigna maestra. Ciò non implica affatto che vogliamo iniziare il nostro dialogo partendo dalla crittografia ricreativa: i cifrari che descriveremo, sia pure ridotti al rango di comportamenti di sistemi complessi, sono assai importanti ancor' oggi: per convincersene basterà aspettare fino al capitolo 4.

In un sistema crittografico il *testo in chiaro*, o il *messaggio*, viene trasformato secondo certe regole nel *testo in cifra*, o *crittogramma*; tale operazione si chiama *cifratura*. Il crittogramma viene quindi spedito a destinazione tramite un opportuno canale di trasmissione, per esempio via radio o, più romanticamente, legandolo alla zampetta di un piccione viaggiatore. Del canale di trasmissione non ci si può tuttavia fidare: lungo il percorso è appostata una spia la quale può *intercettare* il crittogramma e tentare di decifrarlo. Anche il destinatario legittimo decifra il crittogramma e riottiene il testo in chiaro: se il sistema di cifra, o *cifrario*, è ben congegnato l'operazione di *decifrazione* o *decifratura* deve risultare piuttosto semplice al destinatario legittimo, ma di complessità proibitiva alla spia. Ciò è possibile perché il destinatario legittimo conosce certe informazioni che devono rimanere del tutto inaccessibili alla spia e

CRITTOGRAFIA

che perciò non devono venire affidate al canale poco sicuro usato per trasmettere i crittogrammi. Tali informazioni costituiscono la *chiave del cifrario*.

Il modello che abbiamo delineato è schematizzato in figura 1.1; esso potrà sembrare piuttosto semplicistico al lettore, specie per il nostro incauto accenno ai piccioni viaggiatori: ne integreremo e approfondiremo la descrizione un po' alla volta, man mano che ci avvicineremo a situazioni tipiche dei giorni nostri. Si noti che nella figura abbiamo distinto fra decifrazione e *decrittazione*: quest'ultima è l'operazione illegittima in cui non ci si può avvalere della chiave (in realtà nell'uso corrente spesso non si distingue e si parla di decifrazione in entrambi i casi).

Un'obiezione che ci potrebbe venire rivolta è la seguente: dalla figura appare ovvio che la pur segretissima chiave va resa nota al destinatario e dunque gli deve venire in qualche modo trasmessa. Il sistema allora funziona solo se disponiamo di un *canale speciale* assolutamente fidato (non soggetto a intercettazioni) destinato alla trasmissione della chiave: nello schema il canale speciale corrisponde al collegamento in alto. Ma se così è, perché non trasmettere sul canale anche il crittogramma? Tanto varrebbe anzi, visto che non c'è possibilità di intercettazione, trasmetterci subito il messaggio in chiaro!

In realtà le cose non sono così semplici, o per dir meglio lo sono solo quando non c'è nessun bisogno della crittografia. Altre volte invece l'ipotesi dei due canali non è affatto artificiosa. Le "dimensioni" della chiave sono di norma molto più

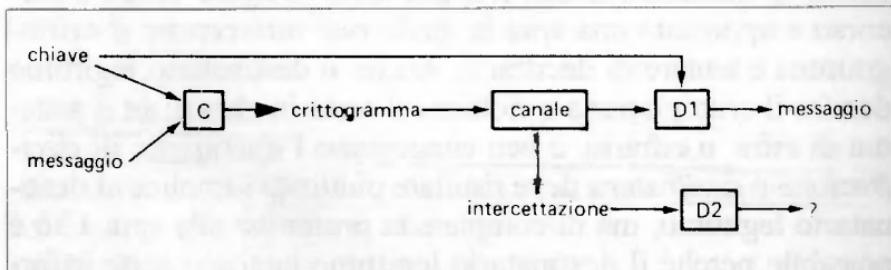


Fig. 1.1 Cifratura (C), decifrazione (D1) e descrittazione (D2).

ridotte di quelle del testo in chiaro: un'unica chiave relativamente breve viene impiegata per smaltire un traffico di messaggi piuttosto intenso. Ora l'uso del canale speciale che non è soggetto a intercettazioni potrebbe essere costoso, oppure il canale potrebbe essere disponibile solo per intervalli di tempo troppo brevi per trasmettere l'intero messaggio ecc. Si pensi al seguente caso estremo: tutte le volte che il mittente vuole rivelare una nuova chiave al destinatario lo richiama in sede, gliela comunica a viva voce e lo rimanda a destinazione.

Nella terminologia abituale della crittografia si è soliti dire che la chiave viene comunicata al destinatario tramite un *corriere* (incorrottibile): è una terminologia suggestiva, ma si badi che oggi i corrieri non cavalcano più a briglia sciolta perché di solito sono costituiti da prosaici congegni elettronici.

Il problema della distribuzione delle chiavi, come si può già intuire, è di importanza cruciale per il buon funzionamento di un sistema crittografico; per il momento ci accontenteremo tuttavia dello schema rappresentato nella figura 1.1, che è sufficiente in molte delle situazioni più tradizionali in cui viene usata la crittografia. Aggiungiamo un'osservazione di natura psicologica: qui e in seguito prenderemo sempre le parti degli utenti legittimi e mai quelle dell'intercettatore; dunque se parliamo di "situazione favorevole" intendiamo dire che l'intercettatore si trova nei pasticci. Del resto già la scelta di termini come "legittimo" o "spia" tradisce quali siamo le nostre simpatie. Tutto ciò, va da sé, serve solo per capirci meglio: potrebbe darsi infatti, che l'intercettatore operi a fin di bene per mandare a monte le trame losche dei cosiddetti "utenti legittimi". D'altra parte non si possono studiare i metodi per costruire un cifrario senza studiare insieme gli eventuali metodi per demolirlo; in termini più tecnici non ci si può occupare *di crittografia* (la parte costruttiva) senza occuparsi di *crittanalisi* (la parte distruttiva); insieme crittografia e crittanalisi costituiscono una disciplina unitaria che si chiama *crittologia*. Critografi e crittanalisti sono dunque le stesse persone, e non fa differenza prendere le parti degli uni piuttosto che degli altri. (Nell'uso corrente si dice spesso "crittografia" là dove si dovrebbe dire

“crittologia”: anche noi ci renderemo colpevoli di questa imprecisione.)

2 Sostituzioni e trasposizioni

In questi primi capitoli i messaggi, per il rispetto dovuto all’età dei cifrari che stiamo descrivendo, saranno costituiti da frasi in buon italiano (possiamo pensare a una lettera riservata); i messaggi in binario dovranno aspettare fino al capitolo 4. Una forzatura che ci permetteremo è quella di eliminare i segni d’interpunzione, gli accenti, gli apostrofi, gli “a capo”, i “ritorni di carrello” ecc.; scopriremo perfino quegli utilissimi “simboli speciali” che sono gli spazi, e non staremo a distinguere fra minuscole e maiuscole. Queste decisioni sono del tutto arbitrarie, non fanno molta differenza in quello che stiamo per dire e hanno il solo scopo di rendere più scorrevole l’esposizione. Per chiarezza continueremo a “scrivere” gli spazi nei messaggi in chiaro, ma nell’eseguire la cifratura il lettore dovrà ignorarli; sempre per chiarezza le lettere dei critogrammi verranno compattate a gruppi di cinque.

Un messaggio da cifrare potrebbe essere:

FUGGI PRIMA DEL TRAMONTO

Fissiamo una *permutazione* dell’alfabeto italiano, che ha 21 lettere, come quella che segue:

A B C D E F G H I L M N O P Q R S T U V Z (1.1)
E S F O A N T B C Q U D P G H R I M Z V L

(nella seconda riga compaiono tutte le 21 lettere, ognuna una sola volta).

La regola di cifratura è semplice: si sostituisce ogni lettera del messaggio in chiaro con la lettera corrispondente che sta nella riga in basso della permutazione, le A con altrettante E, le B

con altrettante S ecc. Il crittogramma risultante è:

NZTTC GRCUE OAQMR EUPDM P

La regola di decifrazione è ovvia, basta usare la *permutazione inversa*, che si ottiene “capovolgendo” la permutazione diretta:

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

ossia, riordinando la prima riga (ciò che in realtà non è affatto necessario):

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

Poiché abbiamo preso le parti del crittografo ci può far piacere notare che le permutazioni di un alfabeto di 21 lettere sono $21! = 1 \times 2 \times 3 \times \dots \times 21$, che è un numero di 19 cifre decimali ($21! \approx 5 \times 10^{19}$).

Il principio che abbiamo illustrato è quello della sostituzione. Anche il principio di trasposizione è basato su una permutazione: in questo caso però il crittografo deve fissare un numero intero P, che si dice periodo della trasposizione, e scegliere una permutazione degli interi da 1 a P; un esempio per P = 7 è:

$$\begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 4 & 6 & 3 & 5 & 7 & 1 & 2 \end{array} \quad (1.2)$$

Il messaggio viene allora spezzato in blocchi di lunghezza P e le P lettere di ciascun blocco vengono rimescolate, o anagrammate, in base alla permutazione: in ogni blocco in cifra si scrivono la quarta, la sesta, la terza, ..., la seconda lettera del corrispondente blocco in chiaro. Il crittogramma corrispondente a

FUGGI PR/IMA DEL T/RAMONTO

è allora

GPGIR FUDLA ETIMO TMNOR A

Beninteso le barre che separano i blocchi sono state scritte solo per nostra comodità; la scelta del periodo 7 è un po' disonesta perché ci esime dall'obbligo di completare il messaggio in maniera che la sua lunghezza sia un multiplo del periodo (non è affatto ovvio come convenga fare questo completamento per non offrire informazioni suppletive al crittanalista). Si noti che il crittogramma è composto dalle stesse lettere del messaggio, ognuna con la sua frequenza. Per la decifrazione si usa la permutazione inversa

4 6 3 5 7 1 2

1 2 3 4 5 6 7

ossia, riordinando la prima riga:

1 2 3 4 5 6 7

6 7 3 1 4 2 5

3 Cifrari a sostituzione monoalfabetica

L'attributo di *monoalfabetico* usato nel titolo di questo paragrafo si riferisce al fatto che i cifrari che stiamo per illustrare adoperando un unico *alfabeto sostitutivo*, vale a dire un'unica permutazione del genere della (1.1).

Cifrario di Cesare

Ogni lettera viene sostituita da quella che la segue di tre posizioni nell'ordinamento normale dell'alfabeto (prolungato per *circularità*: si conviene che la Z sia seguita dalla A, come in

ciascun dei quattro anelli della figura 1.2). La permutazione circolare corrispondente è allora:

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

Cifrario a rotazione o cifrario additivo

Le ragioni di questo secondo nome verranno chiarite più avanti. Stavolta sono disponibili tutte le 21 permutazioni circolari: la rotazione può essere di 0, 1, 2, ..., 19 o 20 posizioni. Beninteso, poiché il cifrario è monoalfabetico, una volta che l'ampiezza della rotazione sia stata scelta essa va conservata per tutta l'operazione di cifratura. Una relativa "meccanizzazione" del cifrario a rotazione (abbastanza modesta da non contraddirre il titolo del capitolo) si può ottenere mediante due dischi concentrici che possono ruotare l'uno rispetto all'altro, come nella figura 1.2.

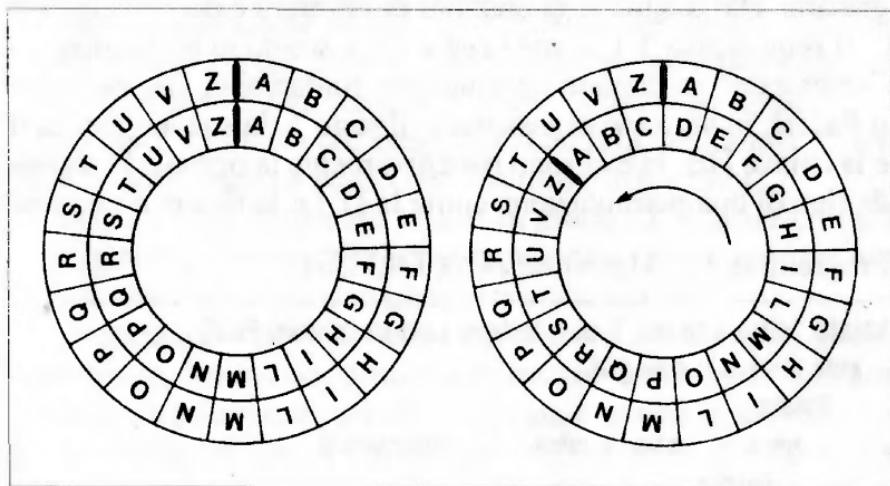


Fig. 1.2 Dischi per il cifrario a rotazione

Supponiamo che la rotazione scelta sia nuovamente quella che porta la A sulla D: per cifrare il messaggio basta ruotare il disco piccolo in senso antiorario di tre posizioni e usare il congegno dall'esterno verso l'interno; per decifrare il crittogram-

ma lo si usa dall'interno verso l'esterno.

In un cifrario a rotazione la permutazione impiegata può venire assegnata specificando qual è la lettera che nel crittogramma corrisponde alle A in chiaro (nell'esempio tale lettera è la D). Si noti che fra le rotazioni ammesse c'è anche quella, assai poco attraente a dire il vero, di zero posizioni, che lascia il messaggio tale quale.

Cifrario completo

È facile capire che il cifrario a rotazione, in cui sono disponibili tutte le 21 permutazioni circolari, è un cifrario un po' più astuto di quello legato al nome, pur illustre, di Giulio Cesare. Ma anche 21 è un numero troppo piccolo: nel cifrario completo sono invece disponibili *tutte* le $21! \approx 5 \times 10^{19}$ permutazioni dell'alfabeto. Naturalmente si perde la comoda possibilità di attuarlo con i dischi della figura 1.2. Poco importa: il lettore in possesso di un calcolatore non farà fatica a scrivere un programma che esegua le operazioni di cifratura e decifrazione.

Il programma 1.1, come i seguenti, è scritto in un linguaggio "strutturato" (e dunque agevolmente traducibile) che richiama il Pascal. In ingresso si assegnano il testo T, la sua lunghezza n e la chiave RS. R e S sono rispettivamente la prima e la seconda riga di una permutazione come la (1.1); la R non è vincola-

Programma 1.1 Algoritmo *cifra*(T,n,R,S)

inizial "cifra il testo T di n lettere con la chiave RS"

per i:=1 **a** n **esegui**

inizial

se t_i=' ' **allora** *cifra*_i=' ' **altrimenti**

inizial

j:=1

finché t_i≠r_j **esegui** j:=j+1

*cifra*_i:=s_j

fine

fine

fine

ta a seguire l'ordinamento alfabetico normale: questa maggiore flessibilità ci tornerà comoda nel prossimo capitolo.

Per semplicità abbiamo rinunciato a compattare le lettere del crittogramma a gruppi di cinque: gli spazi, dei quali va tenuto conto quanto si computa n , vengono lasciati al loro posto.

4 Crittografia tattica e crittografia strategica

La crittografia di cui si parla in questo libro è molto ambiziosa.

Talora basta proteggere le informazioni riservate da intercettatori più o meno occasionali e per periodi piuttosto brevi: non ha molta importanza che il crittogramma venga decrittato se per farlo l'intercettatore ci impiega, diciamo, un paio d'ore (o di giorni, a seconda dei casi).

Basta allora usare un cifrario relativamente debole: per esempio i metodi di *rimescolamento* del segnale telefonico sono appunto sistemi crittografici di tipo *tattico*, anche se le loro prestazioni (e il loro costo) possono essere molto diversi. Invece la crittografia di cui ci occupiamo noi ha fini *strategici*: la protezione cui si mira dovrebbe avere, almeno in linea di principio, durata illimitata, o, più realisticamente, durata "praticamente" illimitata (ad esempio fra un bilione d'anni e l'eternità di solito non ha senso fare cavillose distinzioni).

Dopo queste premesse siamo in grado di chiarire con più precisione il concetto di *chiave* (non a caso questo termine non è mai stato usato nei due paragrafi precedenti).

Supponiamo di voler usare un cifrario a rotazione: questa decisione, beninteso, è nota a entrambi i terminali legittimi della comunicazione, quello in entrata e quello in uscita.

La scelta del *tipo* di cifrario prescelto non è sufficiente: ci si deve ancora accordare sulla permutazione circolare che si vuole impiegare, tra le 21 disponibili.

Questa seconda decisione è, per così dire, meno impegnativa della precedente, essa può venir modificata senza cambiare il sistema di cifra e dunque senza dover sostituire le apparecchiature che in concreto lo attuano (per esempio i dischi della figura 1.2).

Per decifrare correttamente il crittogramma intercettato una spia deve: 1) scoprire il tipo di cifrario che è stato adoperato, nel nostro caso un cifrario a rotazione, 2) identificare la permutazione circolare che è servita a cifrare il messaggio. In base alle argomentazioni del paragrafo 1.1 si potrebbe pensare che il concetto di chiave copra globalmente sia l'informazione segreta del punto 1 sia quella del punto 2. *Invece è solo a quest'ultima che si dà il nome di chiave*: in altri termini il cifrario a rotazione offre la possibilità di scegliere fra 21 chiavi diverse, ognuna delle quali corrisponde a una delle 21 permutazioni circolari dell'alfabeto. In modo analogo il cifrario di Cesare ha un'unica chiave e quello completo ha 21! chiavi.

Tutto ciò potrà sembrare una pura questione di nomi, ma sotto c'è molto di più. L'esperienza ha infatti insegnato ai crittologi un'amara lezione: una spia ben preparata, con tempo e denaro a disposizione, riesce di regola a scoprire qual è il sistema di cifra usato. Ciò non è dovuto soltanto a ragioni "teoriche" legate all'esistenza di tecniche crittanalitiche particolarmente efficienti per discriminare sistemi diversi. Il punto principale è questo: nella crittografia strategica il sistema di cifra *non* deve risultare irreparabilmente compromesso se il congegno di cifratura o quello di decifrazione finiscono nelle mani dell'avversario; d'altra parte risalire dal congegno di cifratura/decifrazione al tipo di cifrario di cui il crittologo si è servito non è un compito difficile per un professionista della crittanalisi. I crittologi hanno ormai un punto di vista radicale a questo proposito: la sicurezza offerta da un sistema crittografico usato a fini strategici è affidata *esclusivamente* alla segretezza della chiave (all'informazione segreta del punto 2).

Vediamo che cosa succede dei cifrari del paragrafo 3 se accettiamo questo punto di vista. Per il cifrario di Cesare non c'è niente da fare: come ogni cifrario *degenera*, in cui è disponibile una sola chiave, esso non ha nessun senso strategico: *tanto varrebbe trasmettere il messaggio in chiaro*. Il cifrario a rotazione non è degenero, ma 21 chiavi sono veramente poche: all'intercettatore, senza bisogno di profonde tecniche crittanalitiche, basterà tentare, una per una, tutte le 21 possibilità: il

metodo della *ricerca esauriente*, pur così rozzo, è stavolta adeguato. Anche il cifrario a rotazione va dunque scartato.

Per quanto oggi i calcolatori rendano possibili controlli di una complessità incredibile di fronte alle $21! \approx 5 \times 10^{19}$ chiavi del cifrario completo la ricerca esauriente appare davvero troppo primitiva; nel prossimo capitolo vedremo tuttavia che, sotto i colpi di metodi crittanalitici meglio congegnati, anche il cifrario a sostituzione completa è destinato a crollare, perlomeno quando viene usato da solo, e non viene inserito come componente in un sistema crittografico più complesso.

Nonostante le nostre argomentazioni, il lettore forse sarà ancora perplesso: ritenere che la sicurezza strategica di un sistema crittografico sia interamente affidata alla segretezza della chiave equivale in sostanza a ritenere che la spia sia sempre e comunque al corrente del tipo di cifrario che viene usato.

Può parere un eccesso di scrupolo che i crittologi valutino la validità dei loro sistemi partendo da un'ipotesi così pessimistica, invece l'accettazione del *principio di Kerckhoffs*, come viene talvolta chiamato dal nome di un crittologo franco-olandese del secolo scorso, è un assioma indiscusso della crittologia contemporanea.

Va anche detto che nelle moderne "reti telematiche" è spesso *impossibile* tenere segreto il sistema di cifra: anzi, la descrizione dettagliata del suo funzionamento viene resa pubblica dallo stesso crittografo... ma stiamo anticipando! Questo capitolo è dedicato ai cifrari manuali ed è a essi che dobbiamo tornare.

5 Cifrari a sostituzione polialfabetica

In un cifrario a sostituzione monoalfabetica la chiave è costituita da un'unica permutazione dell'alfabeto.

Un'idea abbastanza naturale è quella di rendere la vita più difficile al crittanalista adoperando *parecchie* sostituzioni a turno. Un esempio di nobili tradizioni che generalizza i cifrari a rotazione è quello del *cifrario di Vigenère* (Blaise de Vigenère visse nel XVI secolo; l'idea dei cifrari polialfabetici era stata anti-

cipata nel '400 da Leon Battista Alberti, umanista e architetto, e, sia pur di pochi anni, da Giovan Battista Della Porta, commediografo e inventore della camera oscura). Per afferrare il principio basterà riferirsi al quadro della figura 1.3.

La chiave del cifrario di Vigenère è costituita da una parola speciale, per esempio la parola *FLAUTO*. Supponiamo di dover cifrare il messaggio

PARIGI VAL BENE UNA MESSA
(FLAUTO FLA UTOF LAU TOFLA)

La riga fra parentesi, in cui la *parola-chiave* viene ripetuta “quanto basta”, serve per la cifratura: la P viene cifrata usando la riga che nel quadro comincia con una F ($P \rightarrow U$), per la A si usa la riga L ($A \rightarrow L$), per la R si usa la riga A ($R \rightarrow R$), ..., per la A si usa la riga A ($A \rightarrow A$). Il crittogramma è:

ULRFC ZDLLV ACLGN UGSAE A

Se si conosce la parola-chiave è facile decifrare il crittogramma: basta ripeterla sotto le lettere che lo compongono e servirsi delle permutazioni inverse. Scrivere un programma che implementi il cifrario di Vigenère è particolarmente semplice se si ricorre agli strumenti dell’aritmetica “circolare” (si veda il paragrafo 5.5): non occorre neppure memorizzare il quadro della figura 1.3.

Naturalmente si potrebbero escogitare cifrari polialfabetici molto più complessi; preferiamo piuttosto spendere qualche parola di commento sul cifrario di Vigenère. Più precisamente cercheremo di rispondere alla seguente domanda: quante sono le chiavi del cifrario? Saperlo è essenziale per poter valutare la sua attendibilità. In un certo modo le chiavi del cifrario sono in un numero infinito: basta permettere al crittografo di usare parole-chiave che non abbiano un senso preciso, ad esempio *TRSTKRF* invece di *FLAUTO* (dichiarando la nuova parola-chiave priva di senso abbiamo contato sul fatto che il lettore

CIFRARI MANUALI

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

Fig. 1.3 Il quadro di Vigenère.

non conosca lo sloveno o il croato, e non sia quindi portato a pensare a un'invitante crociera da Trieste, ossia Trst, a Corfù, ossia Krf).

Una volta concessa questa libertà non ci sono ragioni di principio perché la parola-chiave non abbia lunghezza 30, o 100, o 10.000 ecc. In effetti questa strada ci porterebbe molto lontano, come scopriremo nel capitolo 5, tanto lontano che ci fermiamo subito, rammentandoci che in fondo il cifrario di Vigenère ha parecchi secoli d'età e che i problemi di *attuazione* erano allora ben più seri di oggidì: i corrieri fidati viaggiavano davvero a cavallo ed era essenziale che la chiave fosse breve e facile da ricordare. Nel caso del cifrario di Vigenère, insomma, se non si vuole snaturarlo, la parola-chiave deve trovarsi nel vocabolario. Anche così, beninteso, non è chiaro quante siano esattamente le chiavi ammesse (chi può dire quante siano le parole della lingua italiana?), ma perlomeno sappiamo che il loro numero è finito.

Per lungo tempo il cifrario di Vigenère è stato considerato inattaccabile, finché nel secolo scorso un ufficiale prussiano di nome Kasiski ideò un metodo crittanalitico per cercare di determinare il *periodo* del cifrario (ossia la lunghezza della parola-chiave): da allora forzare il cifrario di Vigenère non è molto più difficile che forzare il suo corrispondente monoalfabetico, ossia il cifrario a rotazione. Di cifrari a sostituzione storici ce n'è tanti, ma ormai non si imparerebbe più nulla di veramente nuovo. Passiamo piuttosto ai cifrari a trasposizione.

6 Cifrari a trasposizione

Di un cifrario a trasposizione bisogna innanzitutto specificare il periodo P , ossia la lunghezza dei blocchi di lettere che vengono anagrammati. Stavolta partiremo dai cifrari completi.

Cifrario a trasposizione completo di periodo P

Le chiavi di un cifrario completo di periodo P sono tante quante sono le permutazioni dei numeri interi da 1 a P . Per esempio per $P=7$ le chiavi sono $7!=1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 = 5040$.

Il metodo della ricerca esauriente in questo caso è ancora ragionevole, specie se si dispone di un calcolatore. Basta però aumentare il periodo di poche decine per mettersi al riparo da tecniche crittanalitiche così brutali. Come il lettore avrà già indovinato, ne esistono però di più raffinate: anche i cifrari a trasposizione sono inadeguati alle esigenze di sicurezza strategica cui miriamo, a meno che P non sia enorme: in questo caso tuttavia il cifrario rischia di diventare troppo complesso persino per gli utenti legittimi (torneremo su questo punto alla fine del prossimo capitolo). Si potrà obiettare che la chiave del cifrario è costituita non solo da una permutazione dei numeri 1, 2, ..., P ma anche dal periodo P che si è scelto: escludere P dalla chiave e considerarlo parte del sistema equivale a supporre che il crittanalista conosca P, o comunque possa facilmente stimarlo, cosa per nulla ovvia. L'obiezione è fondata, ma nel dubbio preferiamo insistere nel nostro atteggiamento di pessimismo; d'altronde ciò potrebbe avere una giustificazione "fisica": cifratura e decifrazione potrebbero venire attuate mediante congegni che non consentono di alterare liberamente il periodo, per cui sarebbe inevitabile pensare a quest'ultimo come parte del sistema e non della chiave.

Oltre al cifrario completo ne esistono altri più deboli, che presentano comunque un certo interesse storico. In questi il periodo è determinato dalla lunghezza del messaggio (il cifrario viene adoperato per un unico periodo: qui è palese che il periodo *non* fa parte della chiave).

Cifrario a inferriata

Il messaggio viene scritto su due righe "scendendo e salendo":

I	M	N	O	F	T	O	S	A	E
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
L	O	D	E	A	T	A	C	L	

Nel crittogramma si scrive la riga in alto e poi quella in basso, così:

IMNOF TOSAE LODEA TACL

CRITTOGRAFIA

Nell'esempio il periodo è $P=19$, che è anche la lunghezza del messaggio, e la chiave usata è:

1	3	5	7	9	11	13	15	17	19	2	4	6	8	10	12	14	16	18

Rispetto alla formula (1.2) si è omessa la prima riga, che non porta alcuna informazione utile. Purtroppo il cifrario a inferriata, che ha un'unica chiave, è un cifrario degenere.

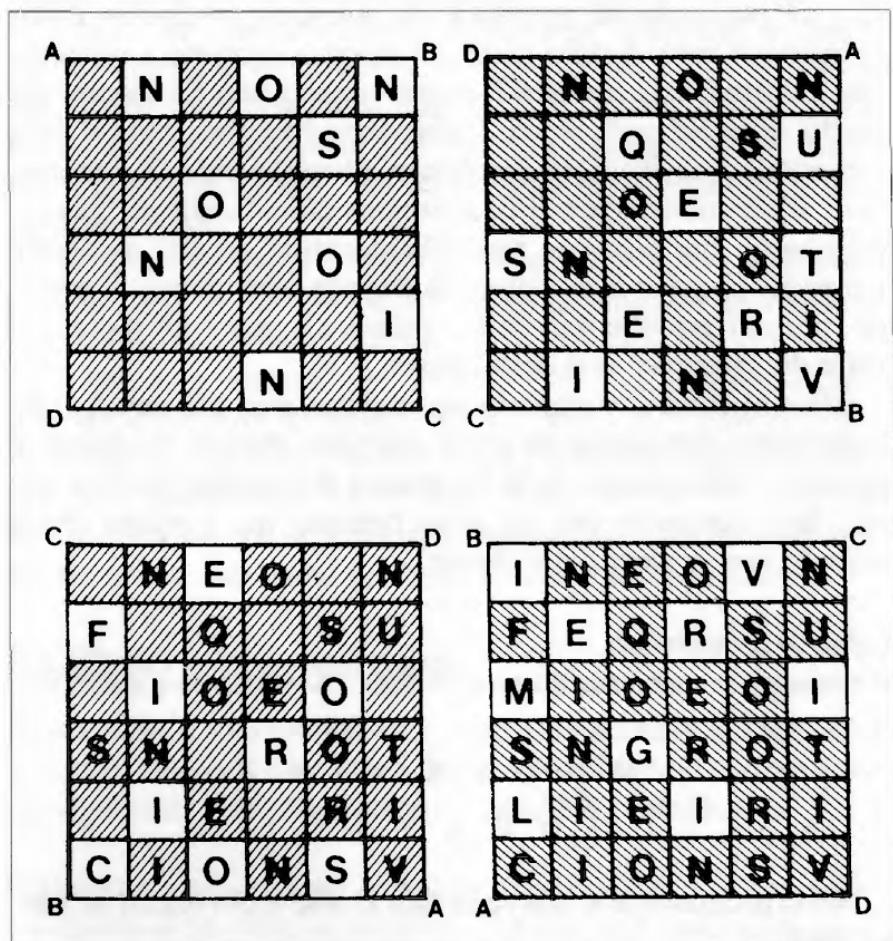


Fig. 1.4 Un cifrario a griglia.

Cifrari a griglia

Nella figura 1.4 è riportato l'esempio di un cifrario a griglia quadrata: il "congegno di cifra" è una griglia con 36 quadretti, 9 dei quali aperti; ruotando via via la griglia di un quarto di giro tutte le 36 posizioni risultano successivamente scoperte e su di esse si scrive il messaggio, da sinistra a destra, dall'alto verso il basso. Abbiamo cifrato il distico del Tasso

NON SONO IN QUESTE RIVE
FIORI COSÌ VERMIGLI

il crittogramma è:

INEOV NFEQR SUMIO EOISN GROTL IEIRI CIONS V

Nonostante i loro illustri precedenti storici (furono studiati da Girolamo Cardano nel '500) e letterari (agli appassionati di Jules Verne sarà senz'altro venuto in mente il *Mattia Sandorf*) i cifrari a griglia sono degeneri.

Cifrario a trasposizione colonnare

Il messaggio viene scritto a blocchi di lunghezza fissa, un blocco per riga, come segue (nell'esempio i blocchi hanno 5 lettere ciascuno; la R in basso è una "zeppa"):

S	E	N	O	N
C	I	F	O	S
S	E	B	I	S
O	G	N	E	R
E	B	B	E	I
N	V	E	N	T
A	R	L	O	R

Se si fissa poi una permutazione da 1 a 5, per esempio

5 1 2 4 3

Nel crittogramma le colonne vengono scritte orizzontalmente nell'ordine specifico dalla permutazione (la quinta colonna viene scritta per prima, la prima per la seconda, poi viene scritta la seconda, poi la quarta e infine la terza); si ottiene, nel solito formato:

NSSRI TRSCS OENAE IEGBV ROOIE ENONF BNBEL

In questo caso il periodo è $7 \times 5 = 35$, che è la lunghezza del messaggio incrementato della zeppa. Non conviene specificare la chiave come una permutazione dei numeri da 1 a 35, ma piuttosto come del resto si è fatto, fissando il numero delle colonne, che qui è 5, e la permutazione che va applicata.

Molti altri cifrari di tipo sostitutivo e traspositivo sono stati storicamente proposti, ma ormai i principi essenziali della sostituzione e della trasposizione dovrebbero essere abbastanza chiari. Nel prossimo capitolo cercheremo di demolire quanto finora abbiamo costruito.

2 Crittanalisi statistica

1 Struttura statistica dell'italiano scritto

Si è già detto che per forzare un cifrario a sostituzione esistono tecniche molto più efficienti della ricerca esauriente; noi saremo particolarmente malevoli con il crittanalista e lo costringeremo a misurarsi con un cifrario *completo*: a priori può essere usata una qualsiasi delle $21! \approx 5 \times 10^{19}$ chiavi possibili. Tuttavia, come esige il nostro pessimismo di crittografi, supporremo che egli sappia fin dal principio, per esempio grazie alla soffiata di una spia, che il cifrario usato è per l'appunto un cifrario a sostituzione monoalfabetica, e supporremo anche che gli sia nota la natura della sorgente dei messaggi che vengono cifrati, ossia, per essere più esplicativi, che egli sappia che i messaggi in chiaro sono scritti in italiano. Rispetto al crittografo, dunque, ci sono solo due dati che il crittanalista ignora: quale sia il particolare messaggio che è stato cifrato e quale sia la chiave che è stata impiegata per cifrarlo.

A questo punto, se il crittanalista sa un po' di statistica e se il crittogramma non si riduce a una manciata di lettere, il cifrario a sostituzione è pronto per venir forzato. Vediamo di capire come.

Supponiamo che il crittogramma consista di cento lettere, che la lettera T non vi compaia mai, mentre la L vi compaia 12 volte. Chi sarebbe disposto a credere che la chiave usata trasformi la A in T e la Q in L, ossia a credere che nel messaggio in chiaro la A non compaia mai e le Q abbia ben 12 presenze? Ciò è

possibile, naturalmente. Esistono frasi italiane di 100 lettere senza nessuna A e con molte Q: ma nella vita di ogni giorno come in crittografia non è saggio far troppo conto degli eventi logicamente possibili, ma del tutto inverosimili. Piuttosto la tabella 2.1, che riporta le frequenze percentuali delle lettere italiane (computate come diremo più avanti) ci porta a credere che la L del crittogramma provenga dalla A, o dalla E, o dalla I, mentre la T provenga dalla Q o dalla Z; tutto ciò ci parrà ancor più convincente se la L è la lettera più frequente del crittogramma, mentre la T è l'unica lettera che ne è assente. Infatti dalla tabella risulta che la A, la E e la I sono le lettere più frequenti in italiano (in testa la A); le lettere meno frequenti sono la Z e, buona ultima, la Q.

TABELLA 2.1 *Frequenze assolute e percentuali delle lettere singole (le frequenze percentuali sono arrotondate alla terza cifra decimale).*

A	1714	0,114
B	160	0,011
C	637	0,042
D	566	0,038
E	1658	0,111
F	141	0,009
G	272	0,018
H	160	0,011
I	1563	0,104
L	966	0,064
M	436	0,029
N	966	0,064
O	1486	0,099
P	421	0,028
Q	85	0,006
R	978	0,065
S	771	0,051
T	1024	0,068
U	528	0,035
V	343	0,023
Z	123	0,008
Totali	14998	0,998

Se teniamo conto di questo tipo di indicazioni, il numero delle chiavi da verificare (il numero delle chiavi *verosimili*) si riduce drasticamente, ma forse non tanto da potere intraprendere un controllo puntuale. D'altra parte l'indagine statistica è appena cominciata: oltre che della tabella 2.1, che riporta le frequenze delle lettere, possiamo servirci delle tabelle 2.2 e 2.3 relative ai digrammi e ai trigrammi (coppie e terne ordinarie di lettere).

TABELLA 2.3 *Trigrammi frequenti*

ENT	LLA	DEL	ALI	NTE	CHE	ELL	EDE	ION	ANO	CON
ITA	ONE	ICA	ALL	NTI	ATT	ESS	PER	TER	ZIO	MEN
TRA	OST	EDI	PAR	ERE	TTO	ARE	NON	ATO	ICI	QUE

Ci sono anche altre informazioni statistiche che possono tornare utili: ad esempio la frequenza della I e della L cresce in inizio frase per influenza degli articoli determinativi IL, LA, LO, mentre la frequenza delle vocali A, E, I e O cresce in fine di parola perché ben poche sono le parole italiane che terminano in U o in consonante.

Non basta. Nel decriptare il crittogramma, oltre alla *struttura statistica* della lingua interviene in maniera decisiva la sua *struttura semantica*: parole come BIRTO o REPOPO, pur essendo del tutto plausibili dal punto di vista statistico, non significano nulla e possono essere scartate; viceversa la parola ENIGMA, che statisticamente è poco convincente per la presenza dell'insolito gruppo GM, diventa del tutto credibile per chi capisca l'italiano e sappia che, nel vocabolario, essa ha pieno diritto di cittadinanza.

Sono opportune alcune parole di commento alle tabelle 2.1 e 2.2 che sono "fatte in casa". Ci siamo serviti di un testo di circa 15.000 lettere (14.998 per la precisione) ottenuto incollando un brano tratto da *Pinocchio* con uno di lunghezza doppia tratto da *Il nome della rosa*. Gli statistici più raffinati potrebbero accampare valide obiezioni alla nostra scelta: d'altra parte bisogna rassegnarsi all'idea che non esistono *le* frequenze delle let-

CRITTOGRAFIA

TABELLA 2.2 Frequenze assolute dei diagrammi.

	A	B	C	D	E	F	G	H	I	L	M	N	O	P	Q	R	S	T	U	V	Z
A	33	39	92	97	47	30	41	3	50	203	95	192	5	93	17	172	149	173	42	119	22
B	31	18	*	*	29	*	*	30	1	*	*	12	*	*	22	*	*	17	*	*	*
C	81	*	64	*	54	*	*	144	78	1	*	*	186	*	2	11	*	*	16	*	*
D	90	*	*	3	114	*	*	*	233	1	*	*	92	*	*	4	*	1	28	*	*
E	72	15	104	121	42	16	87	3	70	169	54	181	7	58	14	253	176	93	30	72	20
F	43	*	*	*	15	13	*	*	32	*	*	*	21	*	*	6	*	*	11	*	*
G	20	*	*	*	17	*	16	2	52	46	*	41	20	*	*	26	*	*	32	*	*
H	4	*	*	*	99	*	1	*	50	*	*	*	5	*	1	*	*	*	*	*	*
I	172	15	111	79	99	24	42	1	27	155	88	188	106	55	12	53	118	101	56	53	10
L	195	9	35	7	146	10	14	1	122	163	24	3	98	24	4	1	15	63	24	8	*
M	112	20	*	*	117	*	*	1	72	*	16	*	65	28	*	*	*	*	5	*	*
N	110	9	36	83	99	13	23	*	50	12	17	16	205	16	16	2	29	175	18	16	21
O	67	13	109	140	67	23	28	5	72	157	100	190	10	76	18	143	155	47	13	49	4
P	66	*	*	*	96	*	*	*	73	4	*	*	75	28	*	58	*	2	19	*	*
Q	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	85	*	*
R	201	8	27	25	172	6	9	*	142	20	24	23	148	13	1	38	32	46	20	13	10
S	48	4	40	*	119	3	5	*	140	1	5	1	104	22	*	*	79	149	49	2	*
T	175	*	*	*	157	*	*	*	155	*	1	*	210	*	*	141	*	126	59	*	*
U	64	10	18	11	64	3	6	*	24	33	12	131	29	8	*	42	18	48	1	3	3
V	95	*	*	*	102	*	*	*	63	*	*	*	66	*	*	6	*	*	3	8	*
Z	35	*	*	*	3	*	*	*	30	*	*	*	22	*	*	*	*	*	*	*	33

tere e dei diagrammi, esistono soltanto *stime* necessariamente arbitrarie; tutto ciò che si può fare è cercare di contenere il grado di arbitrarietà entro limiti ragionevoli (per fare un esempio, in questo libro la C ha di sicuro una percentuale di comparsa anormalmente elevata, specie in inizio di parola, per colpa della CRITTOGRAFIA, dei CIFRARI e delle CHIAVI). Per i nostri scopi non val la pena di fare troppo i sofistici; data la lunghezza del testo adoperato possiamo appellarcia alla *legge dei grandi numeri*: è estremamente improbabile che i dati ottenuti siano stravaganti e che una stima più accurata porti a risultati sensibilmente diversi.

Notiamo per inciso che sommando i numeri della tabelle 2.2 si trova 14.997: non è un errore!

2 Un esempio di decrittazione semiautomatica

Se vogliamo scrivere un programma per decrittare in maniera del tutto automatica crittogrammi provenienti da cifrari a sostituzione, converrà insegnare al calcolatore non solo un po' di statistica, ma anche un po' di semantica, cosa abbastanza problematica. La situazione si fa più semplice se ci accontentiamo di un aiuto parziale, anche se sostanzioso (è per questo che nel titolo abbiamo parlato di decrittazione *semiautomatica*).

Il programma che proponiamo fa uso, per così dire, della forza bruta: non si serve né della struttura semantica né di quella statistica del "secondo e terzo ordine" (dei digrammi e dei trigrammi). Le lettere del crittogramma vengono disposte in ordine decrescente di frequenza: poiché l'ordinamento decrescente naturale è noto (AEIOTRLNSCDUMPVGHBFZQ) si prende per buona la chiave che cifra le A in chiaro nella lettera più frequente del crittogramma, le E nella seconda in graduatoria, ..., le Q nella lettera meno frequente del crittogramma (il lettore esperto di statistica riconoscerà gli ovvi legami di questo procedimento con il principio di *massima verosimiglianza*). In realtà nella tabella 2.1 la L e la N, e la H e la B, hanno la stessa frequenza: ci sono dunque quattro ordinamenti "naturali", di cui uno è stato privilegiato in maniera del tutto arbitraria; tor-

neremo su questo punto più avanti.

Il programma 2.1 richiama il programma 1.1 di p. 18.

Programma 2.1 Algoritmo *decr(T,n)*

inizial "decrittta il testo T di n lettere"

$s_1 := 'A'$; $s_2 := 'E'$; $s_3 := 'I'$; ...; $s_{21} := 'Q'$ "frequenze decrescenti"

inizial "calcolo frequenze"

per $i := 1$ a 21 **esegui** $f_i := 0$

per $i := 1$ a n **esegui**

inizial

se $t_i \neq '$ allora

inizial

$j := 1$

finché $t_i \neq s_j$ **esegui** $j := j + 1$

$f_j := f_j + 1$

fine

fine

fine

inizial "ordinamento frequenze crittogramma"

per $i := 1$ a 21 **esegui** $r_i := s_i$

ripeti

$d := 0$ "indicatore scambi"

per $i := 1$ a 20 **esegui**

inizial

se $f_i < f_{i+1}$ allora

inizial

$x := f_i$; $f_i := f_{i+1}$; $f_{i+1} := x$;

$x := r_i$; $r_i := r_{i+1}$; $r_{i+1} := r_i$; $d := d + 1$

fine

fine

finché $d > 0$

fine

$decr(T,n) = cifra(T,n,R,S)$

fine

CRITTANALISI STATISTICA

DEMZREQUADMA C GRPSQAUC RADEMOCV EQQE MREIUCIICPDA A EQQE
FPDIARVELCPDA DAQQ CONPRUELCPDA IPDP VAFFBC H2EDMP CQ
QCDT2ETTCP A EQFZDA DAQDA IFRCMMZRA GCZ EDMCFBA E DPC
GARVADZMA C TARPTQCNCFC ATTCOLCEDC P O ASRECFF FOEIICFF
UPIMREDP IPQZLCPDC CDTATDPIA A FPDMREIMEDMC DC EQFZDC DC
HZAC GRPSQAUC, QE DPLCPDA DC RCOPODDELE IAUSRE AIARA IMEME
ENNARREME UPOMP GRAIMP FDP EQFZDA DAQDA IZA GCZ CUGPRMEDMC
CUGQCFCPDC FPUA OCUPIMRE EO AIAUGCP Q CDWADLCPDA DAQOE
IMADPIRENCE E RPUE DAO GROUP IAFFPQ EVEDMC FRCIMP EDFBA DE
IZNCFCALE DC ZD FPOCFA DC QZA ICUSPQC GAR RATCIMREFA Q
CONPRUELCPDA IAUSRE AIARA IMFME FFUGRATIE DE TRED MAUQP FPUA
IC G2P VAQARA DEQQ EQNESAMP FAQMCFP PTEU E ZD EQMRP EIGAMMP
DAQOE MREIUCIICPDA DAQQ CONPRUELCPDA NZ DAOCFEME UPQME
EMMADLCPDA TCE DAQOE SCSSCE QE IATRAMALLE DAQOE
FPUZDCFELCPDA ZDP DAC GRDUC FPOCFP FCNREMC A EMMRSZCMP E
TCZQCP FAIERA A UPOMC EQMRC IC DAOCFERPDP DAQ FPRIP DAC
IAFPQC EFOQ IMZQCP DAC FCNRERC QA FZC EGGOCFELCPDC UCOCMERC
IPDP AVCOADMC NZ GRPGRCP DAQ FPRIP DAQQA QZA TZARRA UDOCEQC
FBA HZAIMP IMZQCP RCFAVAMMA ZD TREDDA CUGZQIP PTTC HZAIMA
RCFARFBIA IPDP IMEMA RCGRAIA FPD RCDDPVEME QADE CD VCIME DC
CUGPRMEDMC EGGQCFCPDC FPUUARFCEQC A CDOZIMRCEQC OPVA A
NPDOEUADMEA EIICFZRERA QE GRVEMALLE DAQQ CONPRUELCPDA

TINMLIRUETNE A PLOFREUA LERINABA IRRI NLISUASSAOYE E IRRI
COTSELBIVAOYE DERR ATHOLUIVAOTE SOTO BECCZA QMITNO AR
RATGMIKGAO E IROMTE DERRE SCLANNMLE PAM ITNACZE I TOA
PELBETMNE A GELOGRAHACA EGAVAITA O R EFLIAOCO CRISSACO
UOSNLITO SORMVAOTA ATGETTOSE E COTNLISNITNA DA IROMTA DA
QMEA PLOFREUA RI TOVAOTE DA LADOTDITVI SEUFLI ESSELE SNINI
IHHELLINI UORNO PLESNO COT IROMTE DERRE SME PAM AUPOLNITNA
AUPRACIVAOYE COUE DAUDSNLI ID ESEUPAO R ATBETVAOTE DERRI
SNETOGLIHAI I LOUI TER PLAOU SECORO IBITNA CLASNO ITCZE RI
SMHACACTVI DA MT CODACE DA DME SAUFORA PEL LEGASNLLIE R
ATHOLUIVAOTE SEUFLI ESSELE SNINI COUPLESI DI GLIT NEUPO COUE
SA PMO BEDELE DIRR IRHIFENO CERNACO OGIU I MT IRNLO ISPENNO
DERRI NLISUASSAOYE DERR ATHOLUIVAOTE HM DEDACINI UORNI
INNETVAOTE GAI TERRI FAFFAI RI SEGLENEVVI DERRI
COUTACIVAOYE MTO DEA PLAUA CODACA CAHLINA E INNLAFMANO I
GAMRAO CESILE E UORNA IRNLA SA DEDACILOTO TER COLSO DEA
SECORA IRRO SNMDAO DEA CAHLILA RE CMA IPPRACIVAOYA UARANILA
SOTO EBADETNA HM PLOPLAO TER COLSO DERRE DME GMELLE UOTDAIRA
CZE QMESNO SNMDAO LACEBENNE MT GLITD AUPMRSO OOGA QMESNE
LACELCZE SOTO SNINE LAPLESE COT LATTOBINI RETI AT BASNI DA
AUPOLNITNA IPPRACIVAOYA COUUELCAIRA E ATDMSNLAIRA DOBE E
HOTDIUETNIRE ISSACMLILE RI PLABINEVVI DERR ATHOLUIVAOTE

Esempio 2.1 Un crittogramma prima e dopo la sua (poco felice) decrittazione.

Abbiamo fatto un esperimento (esempio 2.1): abbiamo cercato di decrittare un crittogramma di un migliaio di lettere. Il risultato sembra davvero deludente (oltre tutto abbiamo un po' barato perché gli spazi del messaggio originale sono stati mantenuti al loro posto); a posteriori si vedrà cheabbiamo correttamente decifrato sette lettere su ventuno.

Non bisogna tuttavia scoraggiarsi: vediamo di riaccomodare le cose "a mano". Nel crittogramma decrittato compaiono le parole ESSENE e OGGA: poiché la R e la A e la I hanno frequenze abbastanza vicine è verosimile che si debba leggere ESSERE e OGGI. Se la A e la I vengono scambiate CODACA e CODACE diventano CODICI e CODICE, e ciò è incoraggiante. Per effettuare gli scambi possiamo usare il programma di cifratura del capitolo 1: il "messaggio da decifrare" è il crittogramma (malamente) decrittato e la "chiave di cifratura" è quella che lascia fisse tutte le lettere tranne la R, la N, la A e la I: $A \rightarrow I$; $I \rightarrow A$; $N \rightarrow R$; $R \rightarrow N$.

Ecco il risultato:

TANMRALUETNE I PROFLEUI RELANIBI ALLA NRASUSSIOTE E ALLA COTSERBAVIOTE DELL ITHORUAVIOTE SOTO BECCZI QMATNO IL LITGMAGGIO E ALCMTI DELLE SCRINNMRE PIM ATNICZE A TOI PERBETMNE I GEROGLIHICI EGIVIATI O L EFRAICO CLASSICO UOSNRATO SOLMVIOTI ITGEGTOSSE E COTNRASNATNI DI ALCMTI DI QMEI PROFLEUI LA TOVIOTE DI RIDOTTATUA SEUFRA ESSERE SNANA AHHERRANA UOLNO PRESNO COT ALCMTI DELLE SME PIM IUPORNATNI IUPPLICAVIOTI COUE DIUOSNRA AD ESEUPIO L ITBETVIOTE DELLA SNETOGRAPHIA A ROUA TEL PRIUO SECOLO ABATNI CRISNO ATCZE LA SMHHICETUA DI MT CODICE DI DME SIUFOLI PER REGISNRARE L ITHORUAVIOTE SEUFRA ESSERE SNANA COUPRESA DA GRAT NEUPO COUE SI PMO BEDERE DALL ALHAFENO CELNICO OGAU A MT ALNRO ASPENNO DELLA NRASUSSIOTE DELL ITHORUAVIOTE HM DEDICANA UOLNA ANNETVIOTE GIA TELLA FIFFIA LA SEGRENENVA DELLA COUTMICAVIOTE MTO DEI PRIUI CODICI CIHRANI E ANNRIFFMINO A GIMLIO CESARE E UOLNI ALNRI SI DEDICAROTO TEL CORSO DEI SECOLI ALLO SNMDIO DEI CIHRARI LE CMI APPLICAVIOTI UILINARI SOTO EBIDETNI HM PROPRIO TEL CORSO DELLE DME GMERRE UOTDIALI CZE QMESNO SNMDIO RICEBENNE MT GRATDE IUPMLSO OGGI QMESNE RICERCZE SOTO SNANE RIPRESE COT RITTOBANA LETA IT BISNA DI IUPORNATNI APPLICAVIOTI COUERCIALI E ITDMNRIALI DOBE E HOTDAUETNALE ASSICMRARE LA PRIBANEVVA DELL ITHORUAVIOTE

Esempio 2.2 Un nuovo tentativo di decrittazione.

Non ci vuole molto a capire (basterà usare ancora un paio di di volte il programma di cifratura) che il vero messaggio (tratto da *Teoria dell'informazione*, di Giuseppe Longo, Boringhieri editore) è il seguente, cifrato con la chiave già usata nel paragrafo 1.2.

NATURALMENTE I PROBLEMI RELATIVI ALLA TRASMISSIONE E ALLA CONSERVAZIONE DELL'INFORMAZIONE SONO VECCHI QUANTO IL LINGUAGGIO E ALCUNE DELLE SCRITTURE PIÙ ANTICHE A NOI PERVENUTE I GEROGLIFICI EGIZIANI O L'EBRAICO CLASSICO MOSTRANO SOLUZIONI INGENUOSE E CONTRASTANTI DI ALCUNI DI QUEI PROBLEMI LA NOZIONE DI RIDONDANZA SEMBRA ESSERE STAFA FERRATA MOLTO PRESTO CON ALCUNE DELLE SUE PIÙ IMPORTANTI IMPLICAZIONI COME DIMOSTRA AD ESEMPIO L'INVENZIONE DELLA STENOGRAFIA A ROMA NEL PRIMO SECOLO AVANTI CRISTO ANCHE LA SUFFICIENZA DI UN CODICE DI DUE SIMBOLI PER REGISTRARE L'INFORMAZIONE SEMBRA ESSERE STAFA COMPRESA DA GRAN TEMPO COME SI PUÒ VEDERE DALL'ALFABETO CELTICO OGAM A UN ALTRO ASPECTO DELLA TRASMISSIONE DELL'INFORMAZIONE FU DEDICATA MOLTA ATTENZIONE GIA NELLA BIBBIA LA SEGRETEZZA DELLA COMUNICAZIONE UNO DEI PRIMI CODICI CIFRATI È ATTRIBUITO A GIULIO CESARE E MOLTI ARABI SI DEDICARONO NEL CORSO DEI SECOLI ALLO STUDIO DEI CIRRARI LE CUI APPLICAZIONI MILITARI SONO EVIDENTI FU PROPRIO NEL CORSO DELLE DUE GUERRE MONDIALI CHE QUESTO STUDIO RICEVETTE UN GRANDE IMPULSO OGGI QUESTE RICERCHE SONO STATE RIPRESE CON RINNOVATA LENA IN VISTA DI IMPORTANTI APPLICAZIONI COMMERCIALI E INDUSTRIALI DOVE È FONDAMENTALE ASSICURARE LA PRIVATEZZA DELL'INFORMAZIONE

Esempio 2.3 La decrittazione definitiva.

Beninteso, perché il programma serva a qualcosa la lunghezza del crittogramma deve essere piuttosto elevata; d'altronde si è già detto che esso è estremamente rozzo. Ecco qualche suggerimento per migliorarlo e renderlo meno indipendente da contributi manuali. Innanzitutto oltre che dell'ordinamento strettamente decrescente si potrebbe tenere conto anche di quelli che ne differiscono "di poco" (nel nostro caso non possiamo dimenticare che ci sono quattro ordinamenti "naturali"; anche certe lettere del crittogramma potrebbero avere la stessa frequenza: col nostro programma queste verrebbero ordinate in un unico modo, del tutto arbitrariamente). Si potrebbe scomodare la struttura statistica del secondo e del terzo ordine, per esempio insegnando al programma a servirsi dei diagrammi e dei tri-

grammi fortemente probabili (come DI, ER, NO, TO, AL, ENT, LLA).

Infine, facendo ricorso a un pochino di semantica, si potrebbe tenere in memoria un repertorio di parole notevoli: questo accorgimento è tanto più ragionevole se a priori si ha qualche idea del contenuto del messaggio e ci si aspetta di trovarci determinate parole (per esempio BOMPRESSO, PARROCCHETTO e TRINCHETTO, oppure, in un diverso contesto, DIVORZIO, ALIMENTI, VELENO).

Tutto ciò può servire per discriminare in maniera automatica fra ordinamenti “grosso modo decrescenti”; il programma si complica di molto, ma il gioco potrebbe valere la candela.

Chi ha in uggia il calcolatore, può rinunciarci e in questo caso farà meglio a non trascurare nulla, né sul piano statistico né su quello semantico: la crittanalisi manuale non è certo un’attività divertente, tuttavia anche critogrammi di *poche decine di lettere* alla fine possono venire decifrati.

3 Crittanalisi dei cifrari polialfabetici

I dati della tabella 2.1, che riporta le frequenze dell’italiano scritto, possono essere resi più suggestivi se vengono rappresentati mediante l’*istogramma* della figura 2.1.

Anche a partire da un crittogramma si può costruire l’istogramma delle frequenze: per esempio quello della figura 2.2 è basato sul crittogramma di circa mille lettere che abbiamo decifrato nel paragrafo precedente.

Il confronto dei due istogrammi è assai costruttivo per il crittanalista: in effetti è proprio su tale confronto che si basa la tecnica statistica di cui ci siamo serviti nel programma 2.1. Naturalmente l’operazione di decriptazione sarebbe più difficile se l’istogramma della figura 2.2 fosse più appiattito: al limite, nel caso di un ipotetico crittogramma in cui tutte le lettere avessero la stessa frequenza, il confronto non darebbe nessuna informazione. L’indicazione per il crittografo è chiara: bisogna escogitare qualche sistema per appiattire l’istogramma della figura 2.2.

Una semplice analisi teorica mostra che i cifrari a sostituzio-

CRITTANALISI STATISTICA

ne polialfabetica portano appunto a histogrammi appiattiti; ciò è tanto più vero quanto più lunga è la parola-chiave, specie se essa è "poco regolare" (VANDALO va meglio di BARBARO).

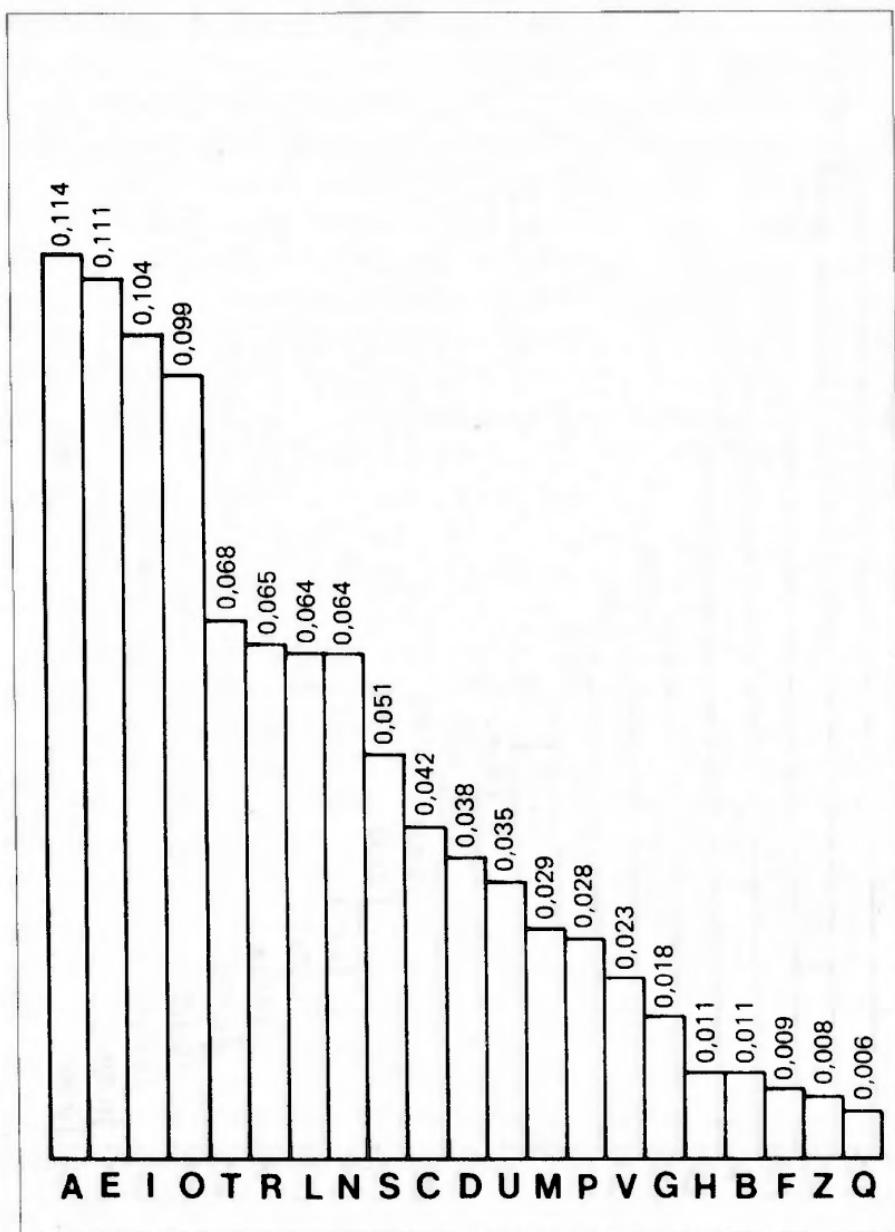


Fig. 2.1

CRITTOGRAFIA

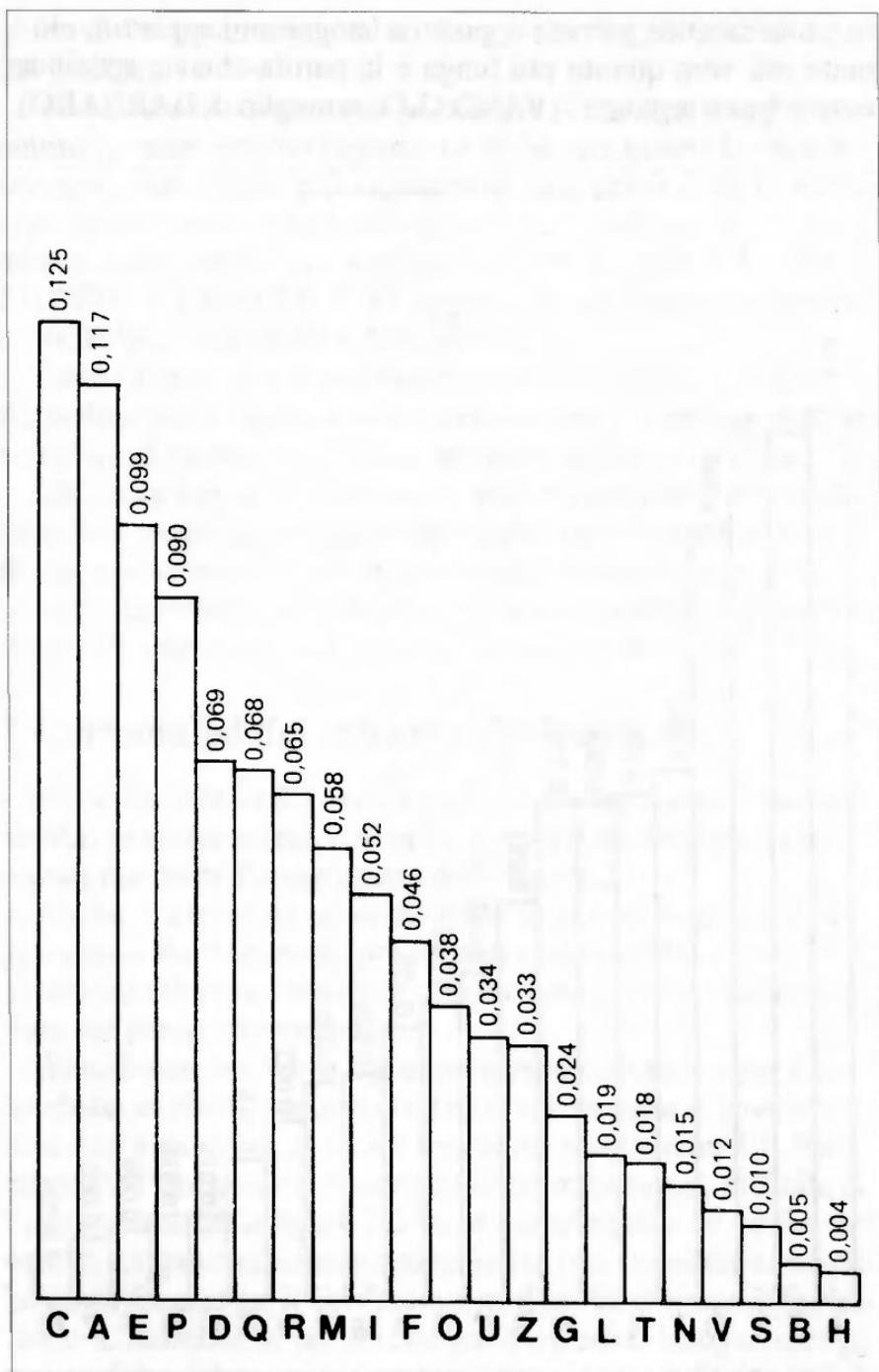


Fig. 2.2

Lasceremo la cura di provare questo asserto a chi si diletta di calcolo delle probabilità; al lettore più impaziente proponiamo l'istogramma di figura 2.3, che è stato ottenuto cifrando con il metodo di Vigenère e con la parola-chiave FLAUTO il brano tratto da *Teoria dell'informazione*.

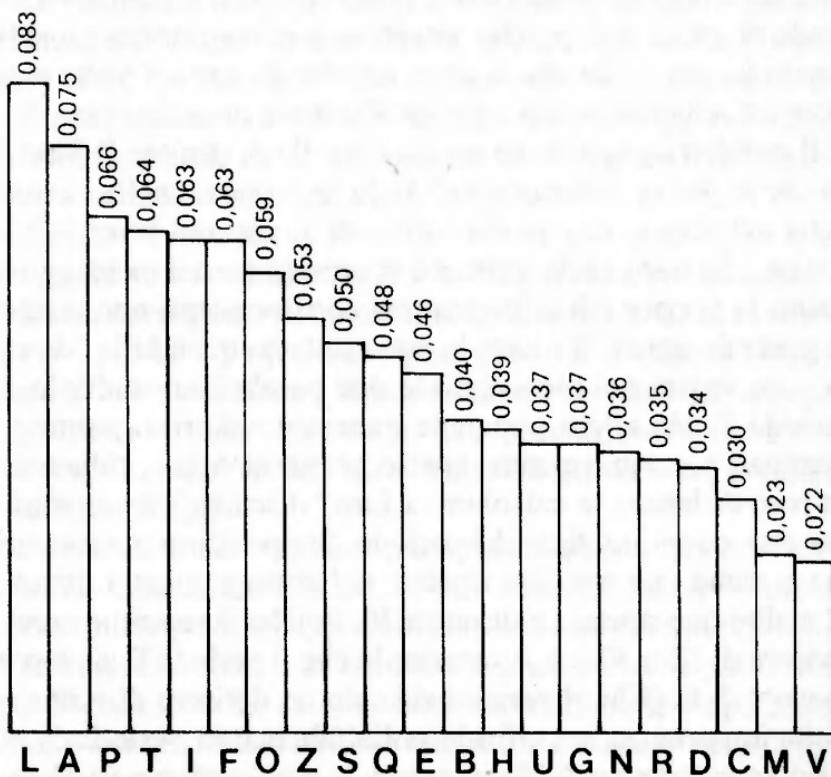


Fig. 2.3

Il compito del crittanalista è ora assai più impegnativo: i cifrari a sostituzione polialfabetica sono stati a lungo (e a torto) ritenuti inattaccabili. Poniamoci di nuovo dal punto di vista del crittanalista e supponiamo che egli abbia scoperto in qualche modo quale sia il periodo P del cifrario polialfabetico usato; per esempio $P=6$. Egli può allora decomporre il crittogramma originario in sei "crittogrammi parziali", ognuno dei quali proviene da un cifrario a sostituzione *monoalfabetica*: quello formato dalla prima, settima, tredicesima, ..., $(6n-5)$ -esima lettera del crittogramma completo, quello formato dalla seconda, ottava, quattordicesima, ..., $(6n-4)$ -esima lettera del crittogramma, ..., e finalmente quello formato dalla sesta, dodicesima, ..., $(6n)$ -esima lettera del crittogramma completo di lunghezza $Pn=6n$. Forzare il cifrario polialfabetico si riduce dunque a forzare sei cifrari monoalfabetici, compito che il crittanalista è in grado di affrontare, purché beninteso il crittogramma sia abbastanza lungo, e stavolta occorre una lunghezza sei volte superiore a quella che bastava per un cifrario monoalfabetico.

Il problema che rimane aperto è quello di stimare P . Storicamente la prima soluzione valida fu la proposta nella seconda metà del secolo scorso dall'ufficiale prussiano Friedrich W. Kasiski. Se due parole identiche si presentano nel messaggio in chiaro le porzioni di crittogramma corrispondenti non saranno in generale uguali fra loro: lo sono tuttavia quando la "distanza" che separa nel messaggio le due parole è un multiplo del periodo P . Ma allora conviene ricercare nel crittogramma le eventuali porzioni ripetute, specie se queste non si riducono a un paio di lettere, e calcolare la loro "distanza": è verosimile che essa sia un multiplo del periodo. Supponiamo per esempio che ci siano due porzioni ripetute nel crittogramma a distanza 12 e altre due ripetute a distanza 30. Poiché il massimo comun divisore di 12 e 30 è 6, è verosimile che il periodo P sia 6 o un divisore di 6. (Che il periodo sia solo un divisore di 6 non ha molta importanza: un cifrario polialfabetico di periodo 3, per esempio un cifrario di Vigenère con la parola-chiave BUE, può sempre esser visto come un cifrario di periodo 6 con la parola-chiave BUEBUE.) A questo punto il crittanalista può mettersi

al lavoro ipotizzando che il periodo del cifrario sia proprio $P=6$: naturalmente la sua ipotesi potrebbe poi rivelarsi sbagliata. In realtà oggi sono noti metodi statistici per determinare P più efficaci di quello (non statistico) di Kasiski: ne risulta che tutti i cifrari polialfabetici sul tipo del cifrario di Vigenère o delle sue varianti, basate sull'uso di parole-chiave facili da ricordare, fanno ormai parte della crittografia storica.

Coglieremo l'occasione per menzionare un altro espediente piuttosto antico (fu usato alla corte dei Gonzaga nel '400) che serve ad appiattire l'istogramma incriminato; purtroppo, come si vedrà, esso porta a una non desiderata *espansione dei dati*.

Supponiamo che l'alfabeto delle lettere usate per scrivere il crittogramma sia più numeroso di quello usato per i messaggi; per esempio si potrebbe ricorrere alle $21^2=441$ copie ordinate di lettere normali. Una volta che le "lettere" sostitutive siano più numerose dello stretto necessario, ci si può permettere di associare molti sostituti alle lettere fortemente probabili, come la A, la E ecc., e di alternare il loro uso nel crittogramma in modo da evitare frequenze relative troppo elevate (i sostituti della stessa lettera in chiaro vengono detti *omofoni*). Per confondere ulteriormente le idee alla spia si può far uso anche di *zeppe o nulle*, inserendo eventuali lettere dell'alfabeto dei crittogrammi che non corrispondono a nessuna lettera dell'alfabeto dei messaggi in chiaro. I *cifrari omofonici* sono alla base dei *nomenclatori*, ossia dei sistemi di cifra che furono usati fino all'epoca napoleonica; essi tuttavia non hanno un interesse meramente storico: il principio dell'omofonia, sia pur rivisto, è valido e vitale ancor oggi.

Prima di chiudere questo paragrafo vogliamo ancora fare un accenno alla crittanalisi dei cifrari a trasposizione. Come si è visto, un istogramma appiattito è un'indicazione attendibile che il crittografo abbia usato un cifrario a sostituzione polialfabetica: discriminare fra i due sistemi, monoalfabetico e polialfabetico, è dunque relativamente facile, e ciò dà ragione alla pratica crittografica di valutare l'efficienza di un cifrario supponendo che il crittanalista sia a conoscenza del sistema di cifra usato e ignori soltanto la chiave. Anche discriminare fra i cifrari a

sostituzione e i cifrari a trasposizione è semplice, visto che in questi ultimi le frequenze del primo ordine (quelle delle lettere singole) rimangono *del tutto invariate*: le lettere del messaggio in chiaro nel crittogramma vengono soltanto rimescolate (trasposte, anagrammate), e non alterate (sostituite).

Ciò non è affatto piacevole dal punto di vista del crittanalista, perché implica che tali frequenze non gli siano più di nessun aiuto nella decrittazione del crittogramma: dovrà partire direttamente dai digrammi.

La prima cosa che viene in mente è quella di accostare le Q alle U; tuttavia nelle tabelle 2.2 e 2.3 sono racchiuse tante altre informazioni forse meno ovvie, ma non per questo meno utili.

Non entreremo nei dettagli delle tecniche crittanalitiche che possono venire utilizzate per forzare un cifrario a trasposizione: rispetto ai cifrari a sostituzione la loro “logica” è grosso modo la stessa, per cui non impareremmo nulla di davvero nuovo.

4 Note consuntive sui cifrari manuali

Dopo questa panoramica sui cifrari “manuali” è giunto il momento di chiederci quali strade siano rimaste aperte, ammesso che di strade aperte ce ne siano.

Si potrebbe ragionare così: anche se non si può essere tanto ingenui da credere che il nemico si imbarchi in un'estenuante verifica di tutte le chiavi a priori possibili, comprese quelle statisticamente inverosimili, possiamo sempre aumentare a dismisura il numero delle chiavi, in modo che non solo le ricerche esaurienti ma perfino quelle parziali diventino inattuabili.

Potremo per esempio ricorrere a cifrari a trasposizione o polialfabetici di periodo estremamente alto, oppure ai cifrari a sostituzione di tipo *poligrafico* (l’idea è semplice: basta pensare che il messaggio sia costruito su un “superalfabeto” le cui “superlettere” siano le coppie ordinate, o più in generale le ennuple ordinate, di lettere normali; il superalfabeto delle ennuple ha 21^n superlettere, e $(21^n)!$, che è il numero delle chiavi, fa traboccare i normali calcolatori già per $n=2$). Un’altra possibilità è quella dei cifrari composti di cui parleremo nel

capitolo 4 (per dirla in breve: diversi cifrari vengono usati "in serie", uno dopo l'altro).

Purtroppo queste proposte portano tutte a un aumento della *complessità* delle operazioni di cifratura/decifrazione (legittima), mentre un buon cifrario fra i suoi pregi deve avere quello della semplicità di attuazione. D'altra parte che cosa sia attuabile o inattuabile dipende dal livello "tecnologico" a cui si opera: vedremo nel prossimo capitolo quali siano stati gli effetti profondi che ha avuto sulla crittografia la rivoluzione industriale iniziata nella seconda metà del Settecento.

Vogliamo fare un'ultima osservazione prima di chiudere questo capitolo: i metodi crittanalitici di tipo statistico funzionano bene solo se il messaggio è "sufficientemente lungo". In realtà, come vedremo nel capitolo 6, è facile escogitare sistemi crittografici "perfetti" in cui il crittogramma non fornisca alcuna informazione all'intercettatore, se si sa che dopo poche lettere il cifrario verrà buttato alle ortiche. Di solito però questo è un lusso che non ci si può permettere: un cifrario "professionale" viene usato per smaltire una quantità di messaggi tutt'altro che trascurabile, ed è di tali cifrari che abbiamo l'intenzione di occuparci.

3 Cifrari a rotore

1 Macchine per cifrare

L'epoca delle macchine e dell'industrializzazione non ha risparmiato neppure la crittografia e ha consentito di realizzare sistemi di cifra estremamente complessi: a partire dalla fine del '700 cominciano a venire proposti e prodotti congegni di tipo meccanico e più tardi elettromeccanico che rendono automatiche le operazioni di cifratura dei messaggi e di decifrazione dei crittogrammi; il culmine ma anche il punto di rottura di tale sviluppo è rappresentato dalla Seconda guerra mondiale. I cifrari che vengono costruiti sono a sostituzione polialfabetica; le loro "componenti" monoalfabetiche sono tutt'altro che complete, ma ciò viene compensato, o almeno si spera, dai periodi estremamente lunghi che i nuovi congegni consentono di ottenere.

Un esempio tipico è l'*Enigma*, una famosa macchina per cifrare usata in varie versioni dai tedeschi, nella Seconda guerra mondiale. A dire il vero, più che famoso l'*Enigma* è malfamato, perché costituisce un caso emblematico di un cifrario ritenuto inviolabile dai suoi utenti e di fatto forzato, dai nemici. Ci accontenteremo di descriverlo in maniera sommaria e un po' semplificata.

La macchina per cifrare (e per decifrare) consiste essenzialmente in un certo numero di *rotori*, ossia di dischi adiacenti che ruotano su un perno comune. Ogni disco ha 26 contatti su una faccia e 26 sull'altra (26 sono le lettere dell'alfabeto italiano

completato da J, K, W, X e Y): i contatti delle due facce sono collegati fra di loro in maniera piuttosto complessa, ma comunque in modo che a ogni contatto di una faccia corrisponda uno e un solo contatto dell'altra. Le lettere in chiaro entrano a un capo del "banco" dei rotori (il primo e l'ultimo disco possono essere fissi, e allora più propriamente si chiamano *statori*), viaggiano attraverso i rotori ed escono all'altro capo come lettere in cifra. La chiave è determinata dai collegamenti all'interno dei dischi e dalla posizione iniziale dei rotori. Nel caso dell'Enigma (o almeno di una delle sue varianti) oltre ai due statori ci sono cinque rotori; ogni rotore può essere di una decina di tipi diversi a seconda della "configurazione" dei collegamenti interni. Il moto dei rotori è *odometro*: dopo ogni lettera il primo rotore avanza di una posizione fino a compiere un giro completo; è allora il secondo rotore che si mette in moto, e così via: i rotori si trovano dunque in posizione iniziale dopo $26^5 = 11.881.376$ passi, il che sembra un periodo davvero rispettabile.

Per forzare l'Enigma gli inglesi si avvalsero delle indicazioni dei loro collaboratori polacchi fuoriusciti. Il centro crittanalitico inglese, situato a Bletchley Park, nel Buckinghamshire, è dotato di uno staff eccezionale che includeva Alan Turing (1912-1954), il padre dell'informatica teorica, è entrato nella storia: è a Bletchley che vennero costruite, proprio per forzare i sistemi di cifra tedeschi, poderose macchine elettroniche di calcolo chiamate *Colossi*: i Colossi sono fra i primi calcolatori (siamo nel 1943), antecedenti di pochi mesi al Mark I dell'Università di Harvard e di un paio d'anni all'ENIAC della Moore School, che di solito passano per essere i capostipiti. (È tuttavia doveroso citare le macchine di calcolo a relè costruite nel 1941 dal tedesco Konrad Zuse: la sua genialità, per nostra fortuna, non fu messa a buon uso dalle autorità naziste.) Lo sviluppo accelerato della crittografia durante la Seconda guerra mondiale è dunque legato in maniera profonda alla nascita dell'informatica. Del resto anche le ricerche di Claude Shannon, il padre della teoria dell'informazione (ossia della teoria matematica della trasmissione dei dati) hanno il loro punto di partenza nel-

la crittografia: *Communication theory of secrecy systems*, che segna la nascita della moderna crittologia teorica, è di un anno posteriore all'ormai storico *A mathematical theory of communication*, che segna la nascita della teoria dell'informazione, solo perché la sua pubblicazione fu ritardata fino al 1949 per ragioni di segreto militare: una versione provvisoria era stata messa a punto già negli anni della guerra.

2 Alcune massime del buon crittografo

Poiché questo è l'ultimo capitolo dedicato alle rievocazioni storiche, non guasterà sentire alcune massime che sono frutto dell'esperienza crittografica "classica" (antecedente ai calcolatori), ma che non per questo sono superate. La prima ci è già nota:

1. *Quando si valuta la sicurezza di un cifrario bisogna supporre che il nemico sia al corrente del sistema usato*: è il principio di Kerckhoffs. La sicurezza del cifrario rimane affidata a quell'informazione segreta maneggevole e facile da alterare che è la chiave.
2. *E' solo il crittologo professionista che può giudicare la sicurezza garantita da un sistema di cifra*: le tecniche crittanalitiche sono molto più ingegnose di quanto gli inesperti possano supporre; in particolare sarebbe un'ingenuità limitarsi a contare il numero delle chiavi disponibili, quasi che l'unica risorsa del crittanalista fosse il metodo della ricerca esauriente.
3. *Un cifrario molto complicato non è necessariamente un cifrario molto efficace*: un aumento della complessità delle operazioni di cifratura/decifrazione (legittima) non implica un corrispondente aumento di complessità per la decriptazione illegittima; ritorneremo su questo punto nel prossimo capitolo. Un cifrario assai elaborato rischia di non venire ben capito dal crittografo che può non accorgersi di punti deboli, come la presenza di operazioni *involutive*, per cui ciò che a un certo punto viene fatto, più avanti viene disfatto.

4. *Gli errori del crittografo sono la miglior arma del crittanalista:* anche ammesso che la macchina non possa sbagliare, il binomio "uomo-machina" non è certo infallibile. Infrazioni frequenti a una corretta pratica crittografica sono (o erano): l'uso di una stessa chiave per un tempo troppo lungo, la trasmissione di un messaggio ripetuto cifrato con chiavi diverse, la trasmissione del messaggio in chiaro dopo che era già stato trasmesso in cifra (ciò potrebbe succedere se la linea è molto disturbata e il ricevitore chiede che il messaggio gli venga ritrasmesso), l'uso frequente di frasi e parole stereotipate, (ad esempio all'inizio e alla fine di un dispaccio), l'uso di chiavi prevedibili, ad esempio di parole-chiave legate al contesto in cui si opera (in tempo di guerra una parola-chiave come VITTORIA è di pessimo auspicio), l'uso di consonanti doppie e di combinazioni frequenti come CH o, peggio, QU: nel testo in chiaro che si vuole cifrare gli errori di ortografia, di grammatica e di proprietà stilistica sono un gran pregio crittografico!

4 Il Data Encryption Standard

1 Cifrari composti

Alla fine del capitolo 2 abbiamo accennato alle strade che la crittografia classica sembrava lasciare aperte a possibili sviluppi; nel capitolo 3 abbiamo visto come una di queste strade sia stata percorsa all'incirca fino alla metà di questo secolo. La situazione oggi è completamente cambiata per una ragione molto ovvia: è comparso il calcolatore, che è dotato di una capacità assai apprezzata dai crittografi e dai crittanalisti, quella di eseguire un numero sorprendente di operazioni ripetitive senza stancarsi mai. Ormai i rotori sono entrati nei musei della scienza; certi procedimenti di cifratura/decifrazione e certe tecniche crittanalitiche di complessità un tempo proibitiva sono oggi alla portata di tutti.

Fra le strade lasciate aperte dalla crittografia classica c'è quella dei *cifrari composti*, cui abbiamo già accennato di sfuggita. Si tratta di un'idea feconda e insieme semplice che permette di rafforzare un sistema di cifra senza doverlo fare a spese di una crescita "esplosiva" (esponenziale) della complessità delle operazioni (leggime) di cifratura/decifrazione: si fa ricorso a strutture di tipo seriale, nelle quali tale complessità cresce in modo più o meno proporzionale al numero dei cfrari "elementari" che le compongono. Si pensi ad esempio al sistema rappresentato nella figura 4.1: il messaggio m viene cifrato dal cfrario a rotazione R_1 ; il crittogramma c_1 risultante funge da "messaggio" di ingresso per il secondo blocco costituente, un

cifrario a trasposizione T1 che produce un nuovo crittogramma provvisorio c2; c2 viene cifrato dal cifrario a rotazione R2 che lo trasforma in c3; c3 viene finalmente trasposto da un cifrario a trasposizione T2 che produce il vero crittogramma, c4.

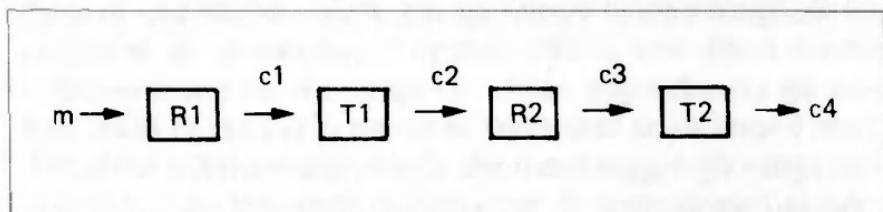


Fig. 4.1 Un cifrario composto.

Beninteso la chiave del cifrario della figura è costituita dalle quattro chiavi dei cifrari componenti, R1, T1, R2 e T2, specificate nel loro giusto ordine. Non bisogna credere che i cifrari composti siano la panacea di tutti i mali: se non si presta sufficiente attenzione c'è il rischio che l'aumento di sicurezza sia del tutto illusorio. Ecco un paio di esempi ovvi: la concatenazione di due cifrari a rotazione è ancora un cifrario a rotazione, la concatenazione di due cifrari a trasposizione di periodo P è ancora un cifrario a trasposizione di periodo P; se i periodi sono P1 e P2 il risultato è un cifrario il cui periodo è pari al minimo comune multiplo di P1 e P2, per cui conviene che P1 e P2 siano numeri primi fra loro ecc. (l'ultima osservazione è utile a chi voglia attuare il cifrario di figura 4.1). L'analisi teorica dimostra soltanto che se si usano per i componenti della catena *chiavi scelte indipendentemente l'una dall'altra* le prestazioni del cifrario composto sono almeno pari a quelle del cifrario componente che nella catena è l'anello più solido: non viene garantito nessun guadagno, ma almeno si sa che non si può far peggio!

Il sistema illustrato in figura è un *cifrario a sostituzioni e trasposizioni*, proprio come il Data Encryption Standard, o DES, che è il protagonista di questo capitolo. Per ragioni di attuazione pratica i cifrari componenti in generale non sono completi (non lo sono nel DES: solo certe sostituzioni e certe permutazioni sono disponibili); questo ci ricorda che i problemi di

complessità sopravvivono anche nell'epoca dei calcolatori elettronici: da quanto diremo nell'ultimo capitolo sembra perfino che siano oggi più importanti che mai.

2 La nuova crittografia

Il cifrario di cui vogliamo parlare in questo capitolo non è più (finalmente!) un cifrario storico. Che in un libro come il nostro si parli in dettaglio di un cifrario *in uso* è di per sé segno evidente di una situazione "sociologica" assolutamente nuova in crittografia. A prima vista il cifrario che stiamo per descrivere, il Data Encryption Standard, non ha nulla di straordinario: si tratta di un cifrario a sostituzioni e trasposizioni; il fatto che il "sistema" sia reso pubblico, di modo che tutta la sicurezza resta davvero affidata alla segretezza della chiave, non sembra contrastare con quanto si è più volte ribadito.

Di fatto, se è vero che i crittografi nelle loro valutazioni partono sempre dal presupposto pessimistico che il nemico sia a conoscenza del sistema di cifra usato, è altrettanto vero che in passato nessuno aveva mai rinunciato ai vantaggi, reali o illusori che fossero, che si sperava di conseguire tenendo accuratamente nascosto tutto ciò che era possibile nascondere, a costo di ricorrere agli inchostri invisibili. Queste cautele avevano d'altronde una loro giustificazione: le situazioni in cui si faceva uso della crittografia erano caratterizzate da una rigida struttura gerarchica che legava gli utenti legittimi del sistema di cifra; in particolare il numero di tali utenti era perfettamente controllato e veniva mantenuto a livelli minimi. Tutto ciò è ancora vero, almeno entro certi limiti, per la crittografia di cui hanno bisogno militari e diplomatici (per tacere degli amanti segreti), e difatti anche oggi i militari e i diplomatici diffidano di un cifrario come il DES il cui funzionamento è stato propalato ai quattro venti.

Oggi tuttavia un'enorme quantità di informazioni riservate viene trasmessa all'interno di sistemi la cui struttura gerarchica è confusa, complessa, appiattita; sistemi che coinvolgono una collettività di utenti assai numerosa, mutevole e mal controllata;

bile a meno di non ricorrere a metodi di schedatura degni del 1984 di George Orwell.

Il DES è un cifrario pensato appunto per le *banche di dati*, e non per le cancellerie diplomatiche o per gli uffici cifra dell'esercito.

Un tale cifrario va descritto, e perfino *reclamizzato*, come si fa per un qualunque oggetto di consumo destinato alla vendita: chi comprerebbe un'automobile le cui caratteristiche tecniche venissero tenute segrete dalla casa costruttrice? (Naturalmente questo discorso rischia di portarci lontano: i costruttori di automobili o di cifrari possono ben decidere di tenere segrete le vere ragioni che li hanno convinti ad adottare certe soluzioni tecniche piuttosto di altre.)

Il DES, che è il frutto del lavoro di ricercatori dell'IBM, è stato adottato e reso pubblico nel 1977 dal National Bureau of Standards americano, che ne ha raccomandato l'uso per la protezione di dati riservati ma non "classificati" (come i segreti militari, i segreti di stato ecc.).

Il DES fa dunque parte di quella che con un'espressione piuttosto riduttiva si usa chiamare la *crittografia commerciale*: se l'espressione è riduttiva (tanto per fare un esempio non è certo di natura "commerciale" il problema di proteggere la privatezza delle cartelle cliniche), serve però a ricordarci che la crittografia è oggi un ramo dell'informatica assai redditizio e in rapidissima espansione. Finora il DES è rimasto sostanzialmente inviolato; ciononostante gli sono state rivolte critiche severe: ne parleremo nel paragrafo 6. I due paragrafi seguenti illustrano certe nozioni preliminari che ci torneranno utili: il DES verrà descritto nel paragrafo 5.

3 Dati binari, codici e cifrari

D'ora in poi supporremo che i messaggi in chiaro siano scritti usando l'alfabeto binario $\{0, 1\}$, cosa che farà piacere agli appassionati del calcolatore, ma che ci costringerà a rinunciare a esempi più o meno suggestivi e a sostituirli con aride sequenze di zero e di uno. La trascrizione, o *codifica*, di un messaggio

IL DATA ENCRYPTION STANDARD

in italiano normale può essere effettuata ricorrendo a uno dei vari *codici* in uso, come l'ASCII (*American standard code for information interchange*) che, in forma un po' ridotta, è descritto nella tabella 4.1.

TABELLA 4.1 *Il codice ASCII (ridotto)* [SP=spazio, RC=ritorno carrello, NR=nuova riga]

A	01000001	a	01100001	0	00110000
B	01000010	b	01100010	1	00110001
C	01000011	c	01100011	2	00110010
D	01000100	d	01100100	3	00110011
E	01000101	e	01100101	4	00110100
F	01000110	f	01100110	5	00110101
G	01000111	g	01100111	6	00110110
H	01001000	h	01101000	7	00110111
I	01001001	i	01101001	8	00111000
J	01001010	j	01101010	9	00111001
K	01001011	k	01101011	.	00101110
L	01001100	l	01101100	,	00101100
M	01001101	m	01101101	:	00111010
N	01001110	n	01101110	;	00111011
O	01001111	o	01101111	!	00100001
P	01010000	p	01110000	?	00111111
Q	01010001	q	01110001	+	00101011
R	01010010	r	01110010	-	00101101
S	01010011	s	01110011	=	00111101
T	01010100	t	01110100	(00101000
U	01010101	u	01110101)	00101001
V	01010110	v	01110110	"	00100010
W	01010111	w	01110111	/	00101111
X	01011000	x	01111000	SP	00100000
Y	01011001	y	01111001	RC	00001101
Z	01011010	z	01111010	NR	00001010

Insistiamo sul fatto che l'operazione di codifica *non* viene eseguita per ragioni di segretezza ma semplicemente perché l'alfabeto naturale dei calcolatori è quello binario. Nel linguaggio corrente la parola "codice" viene spesso usata come sinonimo di "cifrario" o di "firma numerica", e questo può creare confusione. Nell'uso specialistico i concetti di codice e di cifra-

rio sono ben distinti, e solo nel secondo è presente un aspetto di segretezza (l'espressione "codice segreto" al posto di cifrario è però corretta; chi ama il formalismo osserverà che i codici sono cifrari degeneri, a un'unica chiave). Certi codici "intelligenti" oltre a trascrivere i dati li *comprimono*, eliminando *ridondanza* (informazioni superflue). Un tipico esempio è quello dell'alfabeto Morse, o meglio *codice Morse*, che trascrive i dati utilizzando tre "simboli", il punto, la linea e la pausa: il codice Morse realizza una certa compressione dei dati associando a lettere molto probabili *parole di codice* il più possibile brevi e riservando le parole di codice lunghe a lettere improbabili.

Del resto anche la trascrizione dell'eloquio umano in simboli grafici come quelli che state leggendo avviene in base a un codice, anche se in questo caso si tratta di una trasformazione ben più complessa di quella escogitata da Samuel Morse nel 1837. Lo studio dei codici è oggetto *della teoria dell'informazione*, disciplina la cui data di nascita si fa risalire a un famoso articolo di Claude Shannon, "A mathematical theory of communication", cui abbiamo già accennato.

Più esattamente i codici che servono a trascrivere i dati, eventualmente comprimendoli, si chiamano *codici di sorgente*. A essi vanno affiancati i *codici di canale*, il cui scopo è di proteggere i dati dai *disturbi* che possono corromperli quando vengono trasmessi; tale protezione permette di rivelare gli errori o perfino di correggerli (*codici rivelatori* e *codici correttori d'errore*).

Si osservi che i disturbi cui ora ci riferiamo sono eventi "naturali", tipici del canale di trasmissione, e non derivano da un'opera di *sabotaggio* intenzionale da parte di un avversario malintenzionato, nel qual caso la protezione dei dati rientrerebbe di nuovo fra i compiti della crittografia. Il più comune codice rivelatore si serve di un *controllo di disparità*: i dati hanno un formato fisso, diciamo di *7 cifre binarie o bit*, e a ciascun blocco viene aggiunto un bit finale in modo che gli uno dei blocchi siano in numero dispari. Se durante la trasmissione uno dei blocchi binari viene affetto da un singolo errore, il ricevitore si accorge che qualcosa è andato storto perché il numero

degli uno non è più dispari, come dovrebbe essere (naturalmente se gli errori sono due il ricevitore non si accorge di nulla; d'altra parte, a meno che la qualità del canale non sia disastrosa, la maggioranza dei blocchi saranno ricevuti intatti, alcuni saranno affetti da un singolo errore, e solo pochissimi saranno affetti da due o più errori). Degli 8 bit di ciascun blocco i primi sette sono portatori d'informazione, l'ottavo bit non porta alcuna informazione, ed è allora ridondante: si tratta però di ridondanza "intelligente", o *sistematica*, destinata appunto alla protezione dei dati da disturbi di trasmissione.

Abbiamo dunque i codici di sorgente, i codici di canale e i codici segreti, o meglio cifrati: torniamo a questi ultimi che soli sono oggetto di questo libro.

4 Un po' di matematica binaria

Descrivendo l'algoritmo di cifratura del DES avremo bisogno della somma binaria, definita come segue: $0 \oplus 0 = 0 = 1, 0 \oplus 1 = 1 \oplus 0 = 1$ (il suo significato è ovvio se si interpreta 0 come "pari" e 1 come "dispari"). Si possono anche sommare blocchi di bit di uguale lunghezza: il risultato è un terzo blocco ottenuto sommando cifra per cifra, senza problemi di riporto, i due blocchi che fanno da addendi; per esempio $00110 \oplus 10100 = 10010$, più esplicitamente:

primo addendo	0	0	1	1	0	\oplus
secondo addendo	1	0	1	0	0	
risultato	0 \oplus 1	0 \oplus 0	1 \oplus 1	1 \oplus 0	0 \oplus 0	= 10010

Avremo anche bisogno della rappresentazione dei numeri interi in base 2 (invece che in base 10): $0 \rightarrow 0, 1 \rightarrow 1, 2 \rightarrow 10, 3 \rightarrow 11, 4 \rightarrow 100, 5 \rightarrow 101, 6 \rightarrow 110, 7 \rightarrow 111, 8 \rightarrow 1000, 9 \rightarrow 1001, 10 \rightarrow 1010, 11 \rightarrow 1011, 12 \rightarrow 1100, 13 \rightarrow 1101, 14 \rightarrow 1110, 15 \rightarrow 1111$. Il significato di questa rappresentazione è il seguente: se 3782 in base 10 significa $3 \times 10^3 + 7 \times 10^2 + 8 \times 10 + 2$, 1001 in base 2 significa $1 \times 2^3 + 0 \times 2^2 + 0 \times 2 + 1$, numero che

in base 10 si indica col simbolo 9, $(1001)_2 = (9)_{10}$. Talvolta è conveniente uniformare le lunghezze aggiungendo degli zero in testa: così i numeri da 0 a 15 si possono scrivere 0000, 0001, 0010, 0011, ..., 1111, mentre i numeri da 0 a 3 si possono scrivere 00, 01, 10, 11.

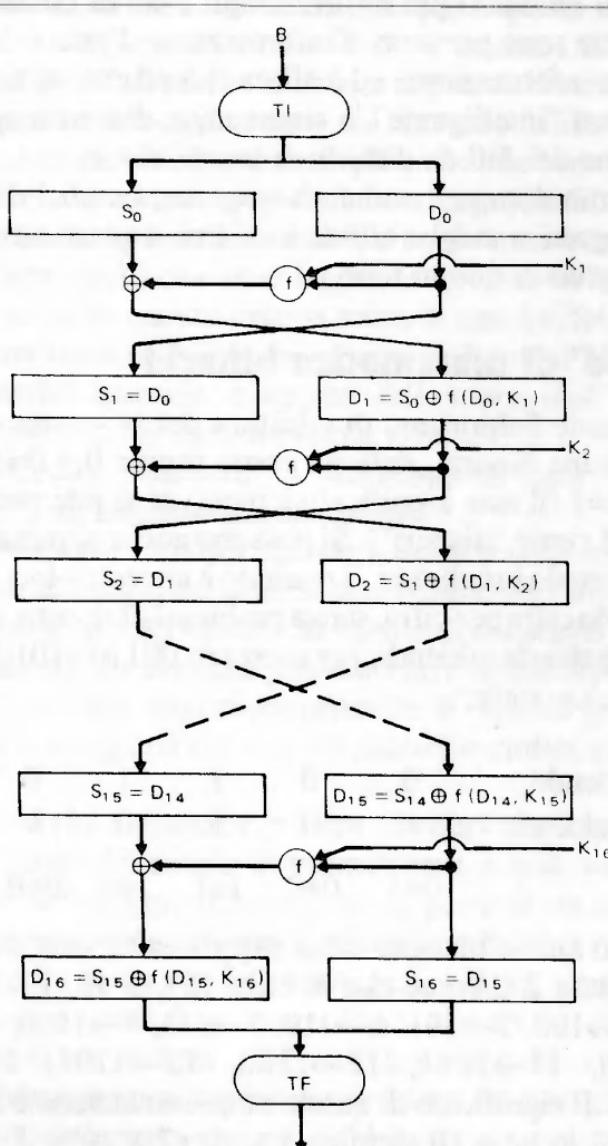


Fig. 4.2 L'algoritmo del DES.

5 L'algoritmo di cifratura e decifrazione del DES

La chiave del DES è un blocco di 64 bit suddiviso in 8 sottoblocchi (*byte*) di 8 bit ciascuno; l'ultimo bit di ogni sottoblocco è di controllo (si tratta di un controllo di disparità): i bit liberi sono allora $64 - 8 = 56$, per cui le chiavi del DES sono in tutto $2^{56} \approx 7 \times 10^{16}$. I dati binari che costituiscono il testo in chiaro da cifrare vengono anch'essi suddivisi in blocchi di 64 bit ciascuno ed entrano a turno nell'algoritmo di cifratura; beninteso l'algoritmo va preventivamente innescato mediante la chiave.

Passiamo ai dettagli. Il resto del paragrafo non è affatto divertente e può venir omesso dai lettori più impazienti.

L'algoritmo, che è lo stesso sia per la cifratura che per la decifrazione, è riassunto nella figura 4.2.

Il blocco *B* di ingresso, composto da 64 bit di testo in chiaro, viene dapprima trasposto mediante una trasposizione iniziale *TI* e diventa $T_0 = TI(B)$. Dopo esser stato sottoposto a 16 passate di una funzione *f* che descriveremo oltre, viene trasposto mediante una trasposizione finale *TF* che è l'inversa di quella iniziale *TI*; l'uscita della *TF* è il risultato finale (il crittogramma in cifratura e il messaggio in chiaro in decifrazione). Le trasposizioni *TI* e *TF* sono descritte nelle tabelle seguenti 4.2 *a* e *b*.

Queste tabelle, come le altre dello stesso tipo riportate più avanti, vanno lette da sinistra a destra e dall'alto verso il basso. Ad esempio *TI* traspone $B = b_1 b_2 b_3 \dots b_{64}$ in $T_0 = b_{58} b_{50} b_{42} \dots b_7$.

TABELLA 4.2a *La trasposizione TI*

58	50	42	34	26	18	10	2	40	8	48	16	56	24	64	32
60	52	44	36	28	20	12	4	39	7	47	15	55	23	63	31
62	54	46	38	30	22	14	6	38	6	46	14	54	22	62	30
64	56	48	40	32	24	16	8	37	5	45	13	53	21	61	29
57	49	41	33	25	17	9	1	36	4	44	12	52	20	60	28
59	51	43	35	27	19	11	3	35	3	43	11	51	19	59	27
61	53	45	37	29	21	13	5	34	2	42	10	50	18	58	26
63	55	47	39	31	23	15	7	33	1	41	9	49	17	57	25

TABELLA 4.2b *La trasposizione TF*

58	50	42	34	26	18	10	2	40	8	48	16	56	24	64	32
60	52	44	36	28	20	12	4	39	7	47	15	55	23	63	31
62	54	46	38	30	22	14	6	38	6	46	14	54	22	62	30
64	56	48	40	32	24	16	8	37	5	45	13	53	21	61	29
57	49	41	33	25	17	9	1	36	4	44	12	52	20	60	28
59	51	43	35	27	19	11	3	35	3	43	11	51	19	59	27
61	53	45	37	29	21	13	5	34	2	42	10	50	18	58	26
63	55	47	39	31	23	15	7	33	1	41	9	49	17	57	25

Fra la trasposizione iniziale e quella finale l'algoritmo effettua le 16 iterazioni della funzione f , funzione che opera sia trasposizioni che sostituzioni. Indichiamo con T_i il risultato della i -esima iterazione, con S_i e D_i rispettivamente il semiblocco di sinistra e il semiblocco di destra di T_i ; scriveremo $T_i = S_i D_i$ per indicare che T_i è la giustapposizione di S_i e D_i ; S_i e D_i hanno 32 bit ciascuno. In base all'algoritmo si ha:

$$S_i = D_{i-1}$$

$$D_i = S_{i-1} \oplus f(D_{i-1}, K_i)$$

dove K_i è una chiave parziale di 48 bit che descriveremo più avanti. Si noti che dopo l'ultima iterazione i due semiblocchi di 32 bit non vengono scambiati: il blocco $D_{16} S_{16}$ viene direttamente sottoposto alla trasposizione finale TF. Ciò è necessario perché l'algoritmo possa essere usato sia per cifrare che per decifrare. Passiamo alla descrizione di alcuni elementi costitutivi dell'algoritmo.

La funzione f e le funzioni di sostituzione Z

La figura 4.3 dà un'idea di come opera la funzione $f(D_{i-1}, K_i)$.

Dapprima il semiblocco D_{i-1} viene espanso a un blocco di 48 bit, $E(D_{i-1})$, usando la tabella 4.3.

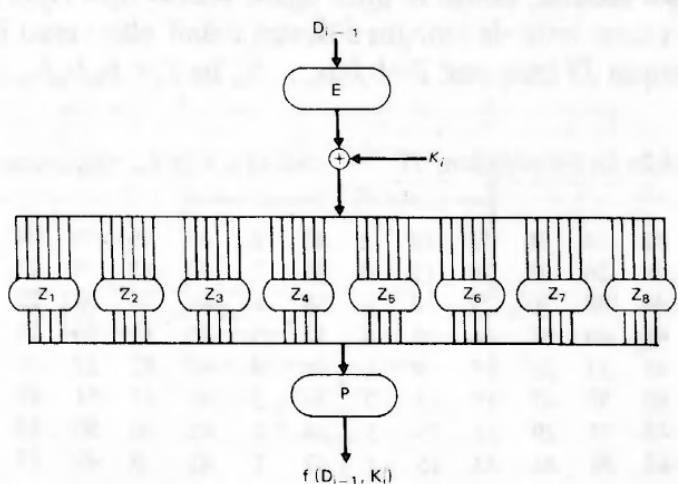


Fig. 4.3 La funzione f .

e espanso a un blocco di 48 bit, $E(D_{i-1})$, usando la tabella 4.3.

Questa tabella va interpretata come le precedenti con la riserva che certi bit di D_{i-1} vengono replicati più volte in $E(D_{i-1})$: dunque se $D_{i-1} = d_1d_2d_3\dots d_{32}$, $E(D_{i-1}) = d_{32}d_1d_2\dots d_{12}d_1$

TABELLA 4.3 *L'espansione E*

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Successivamente si calcola $E(D_{i-1}) \oplus K_i$ e si spezza il risultato in 8 blocchetti di 6 bit, blocchetti che indichiamo con i simboli $\beta_1, \beta_2, \dots, \beta_8$; dunque $E(D_{i-1}) \oplus K_i = \beta_1\beta_2\dots\beta_8$.

Ciascun blocchetto β_j viene usato come ingresso per una funzione di sostituzione Z_j che restituisce un blocchetto più corto, di 4 bit, $Z_j(\beta_j)$. I blocchetti di 4 bit vengono giustapposti dando luogo a una stringa di 32 bit che viene poi trasposta mediante la trasposizione P indicata nella tabella 4.4. Dunque il blocco restituito dalla $f(D_{i-1}, K_i)$ è $P[Z_1(\beta_1), Z_2(\beta_2), \dots, Z_8(\beta_8)]$.

TABELLA 4.4 *La trasposizione P*

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Ciascuna funzione di sostituzione Z_j trasforma il blocchetto di 6 bit $\beta_j = x_1x_2\dots x_6$ in un blocchetto di 4 bit in base alla tabella 4.5. La trasformazione si effettua come segue: il numero intero

CRITTOGRAFIA

che corrisponde a x_1x_6 individua una riga nella tabella, mentre quello che corrisponde a $x_2x_3x_4x_5$ ne individua una colonna.

TABELLA 4.5 *Le funzioni di sostituzione Z*

Righe	Colonne															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Il valore di $Z_j(\beta_j)$ è allora la rappresentazione in base 2, lunga 4 bit, dell'intero che sta all'incrocio fra quella riga e quella colonna. Per esempio se $\beta_1=010011$, allora Z_1 restituisce il valore all'incrocio fra la riga 1 (01 in base 2) e la colonna 9 (1001 in base 2), ossia il numero 6 che in base 2 si scrive 0110.

Calcolo delle chiavi parziali

Ciascuna iterazione i fa uso di una diversa chiave parziale K_i ricavata dalla chiave completa K . Come ciò avvenga è illustrato nella figura 4.4.

K , lo si ricorderà, è un blocco di 64 bit con 8 bit di controllo nelle posizioni 8, 16, 24, ..., 64; tali bit vengono ignorati dalla trasposizione TK della tabella 4.6 che indica come trasporre i 56 bit rimanenti. Il risultato $TK(K)$ viene spezzato in due semi-blocchi di 28 bit ciascuno, s_0 e d_0 (s per "sinistra" e d per

TABELLA 4.6 *La trasposizione TK*

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

"destra"). Questi sottoblocchi sono poi "rotati a sinistra" per 16 volte; si ottengono via via i sottoblocchi s_1d_1 , $s_2d_2, \dots, s_{16}d_{16}$. La chiave parziale K_i usata durante l'iterazione i -esima è $K_i=CK(s_id_i)$, $i=1,2,\dots,16$, dove la "trasposizione compressiva" CK è descritta nella tabella 4.7 (di 56 bit la CK ne traspone solo 48). Rimane da spiegare come agiscano le rotazioni a sinistra, ossia come si ottenga s_id_i a partire da $s_{i-1}d_{i-1}$ ($i=1, 2, \dots, 16$).

L'ampiezza delle rotazioni da usare nelle varie iterazioni è indicata nella tabella 4.8. Supponiamo di essere giunti al passo $i=4$ e di avere $s_3=s_1\ s_2\ s_3\dots\ s_{28}$, $d_3=s_{29}\ s_{30}\ s_{31}\dots\ s_{56}$; la tabella

dice che l'ampiezza della rotazione per $i=4$ è di 2 posizioni a sinistra; allora $s_4 = s_3 s_4 \dots s_{28} s_1 s_2$ e $d_4 = s_{31} s_{32} \dots s_{56} s_{29} s_{30}$.

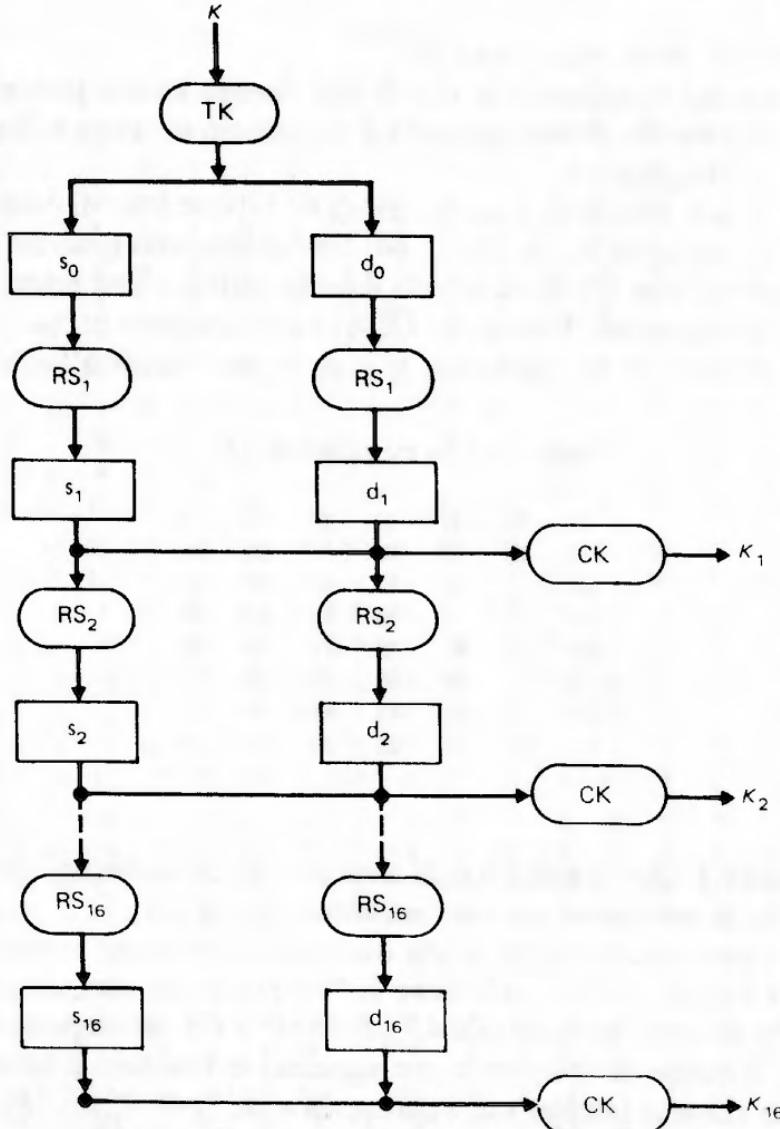


Fig. 4.4 Calcolo delle chiavi parziali.

IL DATA ENCRYPTION STANDARD

TABELLA 4.7 *La compressione CK*

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

La decifrazione viene compiuta usando lo stesso algoritmo, salvo che K_{16} è usato al primo passo, K_{15} al secondo, ..., K_1 al sedicesimo. Ciò avviene perché una trasposizione finale TF è inversa di quella iniziale TI , e perché $D_{i-1} = S_i$, $S_{i-1} = D_i \oplus f(S_i, K_i)$. Va sottolineato che mentre l'ordine delle chiavi parziali è rovesciato, l'algoritmo di per sé non lo è.

TABELLA 4.8 *Le rotazioni a sinistra RS*

Iterazione	Aampiezza della rotazione a sinistra
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Il DES è stato attuato sia in hardware che in software. Nel secondo caso si ottengono tempi di cifratura dell'ordine di parecchi milioni di bit al secondo.

Proponiamo un esempio numerico: ci fermeremo dopo la prima passata, ma ciò sarà sufficiente per farci un'idea chiara di come funziona l'algoritmo.

Cominciamo scegliendo "a caso" la chiave K : il lettore può pensare che per generarla si sia lanciata 56 volte una moneta (in realtà ci siamo serviti di un elenco telefonico; si veda l'ultimo paragrafo del prossimo capitolo). Abbiamo scritto le otto cifre di controllo, quelle più a destra in ciascun blocco (in ciascuna riga), anche se queste non vengono mai usate (i numeri 8, 16, 24, ..., 64 non compaiono nella tabella 4.6):

00001110
10010100
10010001
10101101
00110010
01100010
10001010
01110110

Dobbiamo calcolare K_i . La chiave anagrammata $TK(K) = s_0 d_0$ è la seguente (ci si serva della tabella 4.6):

0100111
0101000
0010111
0001001
1111000
1100010
1101001
0010110

Le prime quattro righe corrispondono a s_0 , le ultime quattro a d_0 . Nella prima iterazione le rotazioni a sinistra da applicare a s_0

e d_0 hanno ampiezza 1 (tabella 4.8). Si ottiene s_1d_1 :

1001110
1010000
0101110
0010010
1110001
1000101
1010010
0101101

e subito dopo $CK(s_1d_1)$ (tabella 4.7), che è proprio la chiave parziale K_1 da usare nella prima iterazione:

000111
000101
010101
101100
001000
111101
000101
010110

Supponiamo che il messaggio da cifrare sia il seguente (DOBRUGIA trascritto in ASCII):

01000100
01001111
01000010
01010010
01010101
01000111
01001001
01000001

Cominciamo con l'anagrammarlo mediante la trasposizione iniziale TI (tabella 4.2 a); si ottiene T_0 :

11111111
00011000
00110011
11110010
00000000
00000000
01000010
00101110

Le prime quattro righe corrispondono a S_0 , le ultime quattro a D_0 , che poi è anche S_1 . Calcoliamo $f(D_0, K_1)$. Prima di tutto ci serve $E(D_0)$, che traspone ed “espande” D_0 (tabella 4.3):

000000
000000
000000
000000
001000
000100
000101
011100

Il risultato della somma binaria $E(D_0) \oplus K_1$ è il seguente:

000111
000101
010101
101100
000000
111001
000000
001010

I blocchi di 6 bit sono rispettivamente $\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6$. Riscriviamoli come coppie di numeri decimali, in base all'algoritmo:

$$(1,3) (1,2) (1,10) (2,6) (0,0) (3,12) (0,0) (0,5)$$

Queste coppie ci rimandano alla tabella 4.5 (il primo di ogni coppia è un indice di riga, il secondo è un indice di colonna). Si ottiene, in decimale:

$$4, 4, 5, 7, 2, 6, 4, 15$$

e in binario:

0100
0100
0101
0111
0010
0110
0100
1111

A questi 32 bit va applicata la trasposizione P (tabella 4.4); ciò che si ottiene è proprio $f(D_0, K_1)$:

10001100
01110011
10011000
10111000

(abbiamo cambiato il formato in vista dell'operazione successiva, scrivendo di seguito il primo e il secondo blocco di 4 bit, il terzo e il quarto ecc.).

Per ottenere D_1 , a $f(D_0, K_1)$ dobbiamo sommare S_0 ; tenendo presente che S_1 coincide con D_0 il "crittogramma parziale" $S_1 D_1$ che esce dalla prima passata è allora:

00000000
00000000
01000010
00101110
01110011
01101011
10101011
01001010

Sconsigliamo il lettore di continuare i calcoli: evidentemente il DES non è un cifrario manuale e il nostro esempio ha uno scopo puramente didattico.

6 Critiche rivolte al DES

Il DES, per quanto possa essere ingegnoso, non chiude affatto il problema della crittografia ed è stato oggetto di critiche severe; del resto già il fatto che non sia destinato alla protezione di dati "segretissimi" è indicativo. Veniamo alle critiche. Come appare dal nome il DES è appunto uno standard, vale a dire un cifrario di uso universale: quello che varia è solo la chiave. L'adozione di un cifrario standard ha degli ovvi vantaggi di economicità e di compatibilità, ma implica altrettanto ovvi svantaggi: se esso dovesse venir forzato, il che non si può escludere, tutti coloro che ne facessero uso dovrebbero cambiare il sistema di cifra e questo comporterebbe spese gigantesche. Non solo: i crittanalisti sono incoraggiati a lavorare solo perché uno standard è un obiettivo particolarmente invitante dal punto di vista della criminalità informatica: con una sola fava, di piccioni se ne cattura un intero stormo!

Ci sono altre critiche che riguardano l'algoritmo in quanto tale. In linea di principio il DES potrebbe essere forzato ricorrendo alla tecnica crittanalitica più rozza, quella della ricerca esaurente: tuttavia le chiavi sono in tutto 256, che sembra davvero una cifra tale da scoraggiare anche i più audaci. Non è così: alcuni hanno affermato che ormai una macchina di calcolo "a scopo speciale" destinata a forzare il DES in tempo ragionevole

è alla nostra portata, o lo sarà fra pochi anni; le valutazioni del suo costo sono assai varie: c'è chi ha parlato di venti milioni di dollari e chi di duecento milioni. Sta di fatto che ormai nell'ambiente crittografico si è diffuso il timore che 2^{56} sia un numero ... troppo piccolo e che bisognerebbe proporre un nuovo DES con una chiave più lunga. Per superare l'impasse, si è suggerito di ricorrere alla cifratura tripla con tre chiavi indipendenti (riecco i cifrari composti!), o se si preferisce con una chiave complessiva di lunghezza $3 \times 56 = 168 : 2^{168}$ (grossomodo 10^{50}) chiavi mettono il DES al sicuro dagli attacchi oggi concepibili:

5 Un cifrario perfetto

1 Un risultato sorprendente

Abbiamo appena visto come neppure il DES, che tuttavia è un cifrario piuttosto ponderoso, si salvi dalle critiche. Sembra dunque che l'ambizione secolare dei crittografi, quella di scoprire un cifrario *perfetto*, tale che il crittogramma, in mancanza della chiave, non fornisca *nessuna* informazione sul messaggio in chiaro, sia destinata a restare insoddisfatta. Ebbene, uno dei risultati più sorprendenti della crittologia moderna è che un tale cifrario in realtà *esiste*, come si può *dimostrare* con rigorose argomentazioni matematiche. Ciò sembra implicare che la crittologia abbia esaurito il suo compito e che le ricerche crittologiche siano ormai ridotte a uno sterile passatempo accademico.

Per fortuna, o per sfortuna, ciò non è affatto vero: vedremo infatti che i cifrari perfetti sono così farruginosi che il loro impiego nella maggioranza delle situazioni pratiche resta precluso; si è allora costretti a ripiegare su cifrari meno perfetti ma più realistici, come il DES. Anche questo risultato negativo non è una mera congettura, bensì un rigoroso teorema di matematica. Ora i teoremi di matematica non si applicano a concetti la cui definizione sia più o meno vaga: il primo passo che dobbiamo fare è dunque quello di fornire un *modello* adeguato e ben preciso dei sistemi crittografici di cui ci stiamo occupando.

Il processo di matematizzazione o di formalizzazione della crittografia è legato in maniera essenziale al nome di Claude Shannon e al suo già citato articolo “Communication theory of

secrecy systems”, pubblicato nel 1949: si è spesso affermato che solo dopo le ricerche di Shannon la crittologia è davvero diventata “scienza” (ciò non toglie che la fantasia e certe abilità “artigianali” rimangono ancor’oggi doti essenziali di un abile crittografo o crittanalista).

2 Un modello matematico delle sorgenti d’informazione

Nel capitolo 2 abbiamo già usato in maniera decisiva la *struttura statistica* del messaggio, o dei messaggi, che costituiscono il testo in chiaro. Le *sorgenti d’informazione* reali, com’è facile immaginare, non hanno affatto un comportamento statistico semplice, che si lasci costringere entro la gabbia di modelli matematici troppo rigidi. Di solito una sorgente d’informazione viene descritta, o meglio *modellata*, mediante un opportuno *processo stocastico* (*casuale, aleatorio*). Per poter dimostrare risultati matematici interessanti si è spesso costretti a fare delle ipotesi restrittive sul processo stocastico che funge da modello, per esempio che il processo sia stazionario, che abbia memoria limitata o perfino che non abbia memoria; queste ipotesi possono essere più o meno artificiose in rapporto alla ricchezza e alla varietà delle sorgenti reali. In altre parole il modello matematico è un’*approssimazione* della realtà; in generale tanto più regolare è il modello tanto più numerosi sono i teoremi di matematica che si riesce a dimostrare, ma tanto più sospetta è la sua adeguatezza. Diciamo subito, per tranquillità del lettore, che i risultati che stiamo per enunciare a proposito dei cifrari perfetti, non presuppongono nessuna ipotesi particolare sul comportamento statistico della sorgente d’informazione che genera i messaggi in chiaro: essi hanno validità *universale*.

Visto che abbiamo cominciato a parlare di processi stocastici, faremo una breve digressione su questo concetto probabilistico così importante nella crittografia teorica. I processi stocastici più semplici da descrivere dal punto di vista matematico sono quelli *stazionari* e *senza memoria*, per esempio quello che si ottiene continuando a lanciare una moneta senza fermarsi

mai (naturalmente i processi reali non hanno una durata infinita se non "in potenza"). Se la moneta è equa (non tarata), le due uscite possibili, "testa" e "croce" (ma noi troveremo più comodo parlare di "zero" e di "uno"), sono equiprobabili (hanno ciascuna probabilità $1/2$; non abbiamo intenzione di lasciarvi trascinare, perché per i nostri fini non ne varrebbe la pena, in una diatriba che farebbe la gioia dei teorici della probabilità: come si fa a stabilire se una moneta è davvero equa?).

Le uscite di un processo stocastico verranno indicate con i simboli come M_1, M_2, \dots, M_n , dove M_n rappresenta il risultato aleatorio dell'ennesima prova (lancio, estrazione, esperimento). Una volta effettuata la prova, la *variabile aleatoria* M_n perde il suo carattere aleatorio e si "materializza" in un certo *valore osservato* che è appunto il risultato concreto della prova; nel nostro esempio M_3 , il risultato aleatorio del terzo lancio, potrebbe "materializzarsi" in "testa", ossia in "zero".

I processi stazionari e senza memoria sono caratterizzati dalla validità della comoda formula:

$$\begin{aligned} &\text{Prob}\{M_1=m_1, M_2=m_2, \dots, M_n=m_n\} = \\ &= \text{Prob}\{M_1=m_1\} \times \text{Prob}\{M_2=m_2\} \times \dots \times \text{Prob}\{M_n=m_n\} = \\ &= \text{Prob}\{M_1=m_1\} \times \text{Prob}\{M_1=m_2\} \times \dots \times \text{Prob}\{M_1=m_n\} \end{aligned}$$

Nella formula n è il numero delle prove e m_1, m_2, \dots, m_n è l'ennupla dei valori concretamente osservati. La prima uguaglianza è conseguenza della mancanza di memoria del processo (in termini tecnici: le variabili M_1, M_2, \dots, M_n sono *indipendenti*), mentre la seconda consegue dalla stazionarietà (eventi "traslati nel tempo" conservano le loro possibilità). Nel caso della moneta, poiché i due risultati possibili in ciascun lancio sono equiprobabili, la probabilità scritta sopra è uguale a $1/2^n$.

Ovviamente se dai lanci di una moneta passiamo a una sorgente d'informazione meno artificiosa, le cui uscite, tanto per fissare le idee, siano le lettere di un testo in buona lingua italiana, il modello stazionario e senza memoria diventa pressocché inservibile tant'è grossolano (ma non del tutto inservibile, come dimostra un familiare esempio tratto dalla teoria dei codi-

ci, l'alfabeto Morse: esso è basato soltanto sulla probabilità delle lettere singole, già le probabilità dei diagrammi vengono ignorate). Per avere un modello che abbia un minimo di ragionevolezza bisogna dotare il processo di memoria: nell'emettere una U il processo si "ricorda" di aver appena emesso una Q, per cui la probabilità del diagramma QU è quasi identica alla probabilità della lettera Q da sola ed è ben più elevata del prodotto delle probabilità "assolute" della Q e della U (*quasi* identica per colpa del solito SOQQUADRO guastafeste, oltre che dell'IRAQ, delle linee aeree australiana QANTAS ecc.). I più semplici processi con memoria sono i processi a memoria limitata, o *processi di Markov*. Il comportamento statistico di un processo markoviano di memoria 1 è completamente descritto dalle probabilità dei "diagrammi" (dei possibili risultati di due prove consecutive), un processo markoviano di memoria 2 è descritto dalle probabilità dei trigrammi ecc. (i processi di Markov a memoria 0 sono i processi senza memoria).

Quando in statistica si parla di *italiano del primo, del secondo o del terzo ordine* ci si riferisce a "lingue" prive di significato che approssimano, ma sono statisticamente, l'italiano reale. In generale le "frasi" dell'italiano di ordine n si ottengono innescando un processo markoviano di memoria n-1 che genera n-grammi di lettere dell'alfabeto italiano con le probabilità corrette (o meglio: accuratamente stimate; si potrebbero per esempio usare le tabelle del capitolo 2). In pratica ci si ferma al terzo ordine: non val la pena di proseguire perché gli "italiani" di ordine maggiore o uguale a 3 sono pressoché indistinguibili. Naturalmente tutte queste approssimazioni ignorano gli aspetti semantici che sono vitali in una lingua reale: non sono ragioni statistiche quelle che ci fanno prevedere che la prima lettera mancante in PROGRESSO TEC... è una N, anche se il diagramma CN ha una probabilità piuttosto bassa.

Può essere interessante vedere come appaia una frase scritta nell'italiano del primo o del secondo ordine. Per farlo ci si può servire del calcolatore; mettere a punto il relativo programma è piuttosto semplice se si ha a disposizione un'istruzione per la generazione di numeri pseudocasuali. Alcuni risultati ottenuti

per i primi due ordini sono riportati negli esempi 5.1 e 5.2. Si noti che l'italiano del secondo ordine ha un'aria più "familiare" di quello del primo.

AUSGRIONOPRENDTUBLAOEAEADORIAASMONORRNLERAGT
NROSERAOARTCRAOELRSNNTDOVEIELIILBORGEUTTZEON
EDTINNAIREITNIESDNIONODLEADMLAAOCDCFOEALD

Esempio 5.1 Una frase in italiano di primo ordine.

AZZALLUEGROLITOBBIEDILELONESTROLTGNSTREGNDI
EHEUNDANARIISOVINOIUNSUNSOSOCODICIDINIZZAVONF
ANRATOPOTATIENARSIAHETESTACCOCHIASISBICIGIRO
LISUEZIVANERBECCHERINTOLENCACRUNITTAUNODUERS
TENDOIORTINEROLIDIMECOUESCHOVILAPEFITAGLLACA

Esempio 5.2 Una frase in italiano di secondo ordine.

3 Il cifrario a chiave non riutilizzabile

In questa sezione i messaggi da cifrare verranno visti come una sequenza binaria di zero e di uno "potenzialmente infinita", costituita dalle realizzazioni (dai valori osservati) di un processo stocastico sul cui comportamento non facciamo nessuna ipotesi restrittiva.

Abbiamo bisogno anche di un secondo processo stocastico binario, senza memoria ed equo, proprio come quello che si ottiene continuando a lanciare una moneta non tarata. Questo processo serve a generare la chiave, che è essa stessa "potenzialmente infinita": il meccanismo che genera la chiave è aleatorio e indipendente dal "meccanismo" che genera il messaggio (i due processi stocastici operano "senza conoscersi"). Per ottenere il crittogramma la chiave viene sommata, cifra binaria dopo cifra binaria, al testo in chiaro (la somma di cui parliamo è, beninteso, quella binaria descritta nel paragrafo 4 del capitolo 4); per decifrare basta sottrarre la chiave del crittogramma: in realtà nell'aritmetica binaria somma e sottrazione coincidono, per cui le operazioni di cifratura e di decifrazione si eseguono allo stesso modo.

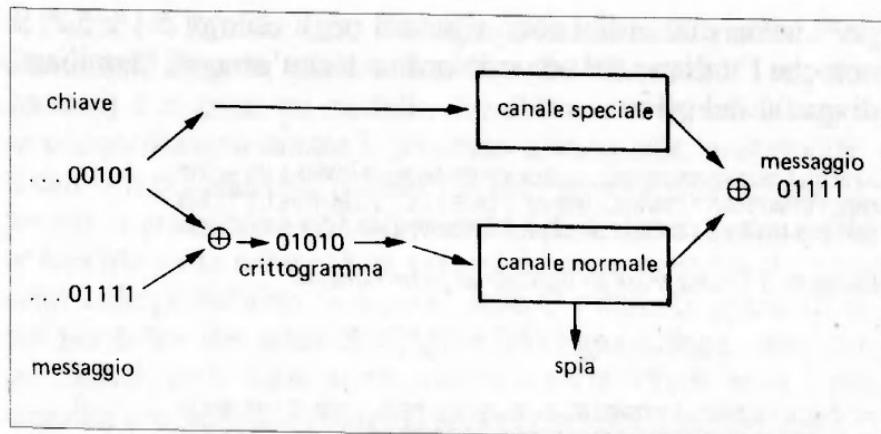


Fig. 5.1 Il cifrario a chiave non riutilizzabile.

Il lettore si sarà già accorto che siamo di fronte a un formidabile problema di attuazione: come generare la chiave? In linea di principio possiamo servirci di una moneta equa, ma rischiamo a lungo andare di mortificare troppo il nostro orgoglio tecnologico, anche ammesso che la mano regga allo sforzo.

Non importa, per ora fingeremo di non esserci accorti di questo intoppo e continueremo a lanciare monete. C'è dell'altro: la chiave del cifrario è troppo "voluminosa", visto che ha le stesse dimensioni del messaggio, o del crittogramma: ciò implica un uso troppo intenso del canale speciale, il che è una contraddizione in termini. Potrebbe perfino sorgere il dubbio che il nostro sistema di cifra non serva a niente, ma ciò è esagerato come mostra il nostro esempio a dire il vero di sapore piuttosto antiquato: trasmettitore e ricevitore si incontrano e generano assieme, per esempio lanciando una moneta, una lunga sequenza di zero e di uno registrata in due copie, una per il trasmettitore e una per il ricevitore. I due poi si separano raggiungendo le rispettive sedi: a questo punto essi sono in grado di scambiarsi messaggi segreti, perlomeno fin quando la chiave non sia esaurita (le cifre della chiave vanno usate una sola volta come dice il nome del cifrario).

Ci sono dunque dei casi in cui il cifrario a chiave non riutilizzabile, per quanto scomodo, può tornare utile; è chiaro però che tali casi sono piuttosto eccezionali. Dopo i difetti è ora di

nominare i pregi: lo faremo nel prossimo paragrafo. Non sono i pregi da poco se, a quanto sembra, è proprio a un cifrario di questo tipo che veniva affidata, in tempi di guerra fredda, la protezione della "linea calda" fra Washington e Mosca.

4 Prestazioni del cifrario a chiave non riutilizzabile

A causa del nostro pessimismo da crittografi valuteremo le prestazioni del cifrario a chiave non riutilizzabile supponendo che il crittanalista (l'intercettatore) ignori quale sia il messaggio trasmesso, ignori quale chiave sia stata usata per cifrarlo, ma per il resto sia al corrente di tutto: conosce il crittogramma, sa qual è il tipo di cifrario usato e sa perfino qual è il comportamento statistico della sorgente dei messaggi (comportamento che il crittografo non ha nessun bisogno di conoscere).

In base a queste ipotesi, di ogni possibile messaggio in chiaro costituito dalle cifre binarie m_1, m_2, \dots, m_n il crittanalista è in grado di calcolare la *probabilità a priori* che esso sia stato emesso:

$$\text{Prob}\{M_1=m_1, M_2=m_2, \dots, M_n=m_n\} \quad (5.1)$$

Una volta che il crittogramma $c_1c_2\dots c_n$ sia stato intercettato, tale probabilità a priori si trasforma in una *probabilità a posteriori*, e cioè nella probabilità che sia stato emesso il messaggio $m_1m_2\dots m_n$ *condizionatamente* al fatto che il crittogramma è proprio $c_1c_2\dots c_n$:

$$\text{Prob}\{M_1=m_1, M_2=m_2, \dots, M_n=m_n | C_1=c_1, C_2=c_2, \dots, C_n=c_n\} \quad (5.2)$$

(Il crittogramma intercettato viene esso stesso visto come la realizzazione di un processo stocastico $C_1, C_2, \dots, C_n, \dots$, che, se si vuole, costituisce la *sorgente dei crittogrammi*; la barra verticale è il simbolo del condizionamento probabilistico: la probabilità a posteriori è dunque una *probabilità condizionata*,

mentre quella a priori è una *probabilità assoluta*.)

Quando si adopera un cifrario “imperfetto” succede di regola che, purché il crittogramma sia abbastanza lungo, c’è un messaggio la cui probabilità a posteriori è vicina a 1. Ciò implica che tutti gli altri messaggi abbiano probabilità a posteriori praticamente nulla: il processo di decrittazione può allora ritenersi concluso; con alta probabilità il crittanalista ha scoperto il messaggio che è stato trasmesso in cifra dal crittografo. (Naturalmente il processo di decrittazione potrebbe essere possibile solo in linea di principio, per ragioni di tempo, di costo ecc. : in altre parole in crittografia c’è posto anche per i cifrari imperfetti, com’è per esempio il DES.)

Ebbene, se si adopera il cifrario a chiave non riutilizzabile succede questo fatto sorprendente: la probabilità a priori e la probabilità a posteriori del messaggio sono *identiche* anche se il crittogramma è molto lungo; in altre parole il crittogramma non fornisce *nessuna informazione* sul messaggio in chiaro. In termini probabilistici il messaggio e il crittogramma sono variabili aleatorie *indipendenti* (beninteso il crittogramma non fornisce informazioni sul messaggio *di per sé*, in assenza della chiave).

Dobbiamo dunque dimostrare che nel nostro caso le probabilità (5.1) e (5.2) coincidono, ed è ciò che ora faremo; il lettore impaziente può invece passare subito al paragrafo successivo.

Cominceremo col calcolare una generica probabilità congiunta del tipo

$$\text{Prob}\{M_1 M_2 \dots M_n = \mu_1 \mu_2 \dots \mu_n, C_1 C_2 \dots C_n = c_1 c_2 \dots c_n\} \quad (5.3)$$

L’unica chiave che sommata al messaggio $\mu_1 \mu_2 \dots \mu_n$ dia il crittogramma $c_1 c_2 \dots c_n$ è la chiave $k_1 k_2 \dots k_n$ in cui

$$k_1 = c_1 \oplus \mu_1, \quad k_2 = c_2 \oplus \mu_2, \quad \dots, \quad k_n = c_n \oplus \mu_n \quad (5.3)$$

Ne consegue che la probabilità (5.3) si può anche scrivere come:

$$\begin{aligned} \text{Prob}\{K_1 K_2 \dots K_n = k_1 k_2 \dots k_n, M_1 M_2 \dots M_n = \\ = \mu_1 \mu_2 \dots \mu_n, C_1 C_2 \dots C_n = c_1 c_2 \dots c_n\} \end{aligned}$$

ossia come

$$\begin{aligned} & \text{Prob}\{K_1 K_2 \dots K_n = k_1 k_2 \dots k_n\} \times \\ & \times \text{Prob}\{M_1 M_2 \dots M_n = \mu_1 \mu_2 \dots \mu_n | K_1 K_2 \dots K_n = k_1 k_2 \dots k_n\} \\ & \quad \times \text{Prob}\{C_1 C_2 \dots C_n = c_1 c_2 \dots c_n | K_1 K_2 \dots K_n = \\ & = k_1 k_2 \dots k_n, M_1 M_2 \dots M_n = \mu_1 \mu_2 \dots \mu_n\} \end{aligned}$$

La prima delle tre probabilità appena scritte vale $1/2^n$ in conseguenza del meccanismo aleatorio di generazione della chiave; nella seconda si può omettere l'evento condizionante poiché la chiave aleatoria e il messaggio aleatorio sono variabili aleatorie fra di loro indipendenti; la terza probabilità vale 1 perché $c_1 c_2 \dots c_n$ è l'*unico* crittogramma corrispondente al messaggio $\mu_1 \mu_2 \dots \mu_n$ e alla chiave $k_1 k_2 \dots k_n$. Dunque la probabilità (5.3) vale

$$1/2^n \times \text{Prob}\{M_1 M_2 \dots M_n = \mu_1 \mu_2 \dots \mu_n\}$$

Ciò è vero in particolare per $\mu_1 \mu_2 \dots \mu_n = m_1 m_2 \dots m_n$; si ha anche:

$$\begin{aligned} & \text{Prob}\{C_1 C_2 \dots C_n = c_1 c_2 \dots c_n\} = \\ & = \Sigma \text{Prob}\{M_1 M_2 \dots M_n = \mu_1 \mu_2 \dots \mu_n, C_1 C_2 \dots C_n = c_1 c_2 \dots c_n\} \\ & = \Sigma 1/2^n \text{Prob}\{M_1 M_2 \dots M_n = \mu_1 \mu_2 \dots \mu_n\} = 1/2^n \end{aligned}$$

(La somma Σ va estesa a *tutti* i 2^n messaggi $\mu_1 \mu_2 \dots \mu_n$ di lunghezza n, la cui probabilità complessiva è appunto 1.) Il rapporto fra la probabilità della coppia messaggio-crittogramma e la probabilità del crittogramma è allora uguale alla probabilità del messaggio (alla probabilità (5.1)): ma tale rapporto è appunto la probabilità condizionata (5.2), come volevasi dimostrare. Si osservi che la probabilità (assoluta) di qualunque crittogramma è $1/2^n$, come risulta dall'ultima catena di uguaglianze: anche la "sorgente dei crittogrammi", come quella delle cifre di chiave, è del tipo "lanci successivi di una moneta equa".

5 Una digressione sulle aritmetiche circolari

Nell'aritmetica binaria si lavora con le cifre binarie 0 e 1; più in generale si possono considerare le cifre m -arie 0, 1, 2, ..., $m-1$. Su questo insieme numerico si può costruire l'*aritmetica circolare modulo m* . Nella figura 5.2 è ad esempio rappresentata, per $m=6$, la somma $3+4=1$ modulo 6 (se non c'è pericolo di equivoci l'indicazione del modulo viene omessa e si scrive semplicemente $3 \oplus 4=1$, o persino $3+4=1$):

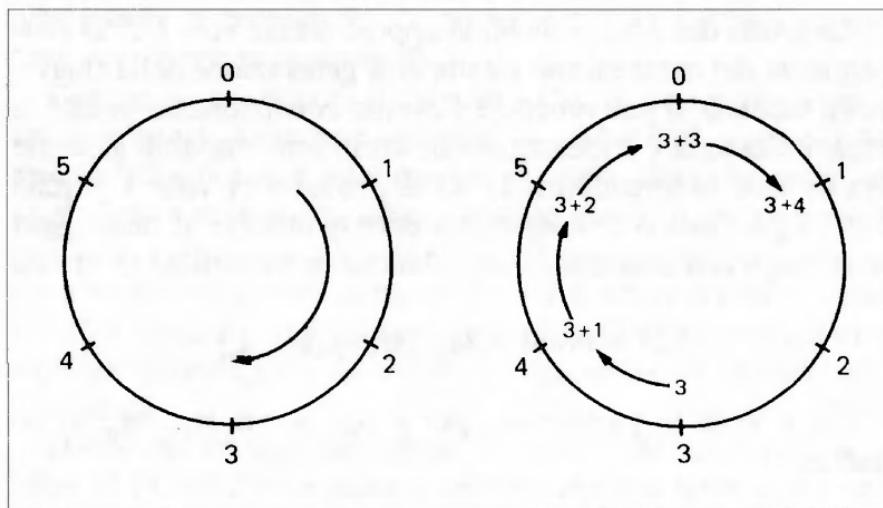


Fig. 5.2 La somma modulo 6.

Rispetto all'aritmetica ordinaria i multipli di m (i "giri completi") vengono scartati. La regola di calcolo è molto semplice: somma e prodotto vengono prima eseguite nell'aritmetica ordinaria, non circolare; il risultato viene diviso per m e si conserva solo il resto. Per esempio, per $m=8$: $6 \oplus 4=2$ perché $10=8+2$, $6 \oplus 7 \oplus 4=1$ perché $17=2 \times 8+1$, $2 \oplus 2=4$ perché $4=0 \times 8+4$, $6 \oplus 4=0$ perché 24 è un multiplo di 8. Nella vita d'ogni giorno le aritmetiche circolari sono comunissime; c'è quella dell'orologio, modulo 12 o modulo 24 a seconda del tipo d'orologio, quella dei contachilometri modulo 100.000, quella dei giorni della settimana modulo 7, e quella dei pari e dispari modulo 2 che già conosciamo bene.

Per quanto concerne le proprietà algebriche delle aritmetiche circolari la somma non dà problemi: valgono tutte le proprietà dell'ordinaria somma fra i numeri reali; in termini tecnici si ha a che fare con un *gruppo finito commutavo*.

Il prodotto invece può creare problemi, per esempio $6 \oplus 4 = 0$ modulo 8, per cui può cadere la regola di annullamento del prodotto, quella che recita: un prodotto è nullo se e solo se lo è almeno uno dei due fattori.

Se tuttavia il modulo m è un *numero primo*, e solo allora, gli inconvenienti scompaiono: in questo caso si è di fronte a una struttura algebrica molto ricca, che i termini tecnici si chiama o *campo di Galois*, in onore del celebre algebrista francese vissuto nel secolo scorso.

Per dirla alla buona, in un campo si possono applicare tutte le comode regole algebriche di calcolo che valgono per la somma e il prodotto ordinari. Possiamo ora tornare ai cifrari.

6 L'alto prezzo della perfezione

È facile generalizzare le argomentazioni dei paragrafi precedenti in modo da ottenere cifrari perfetti che non siano binari.

Consideriamo per esempio un cifrario 21-ario di alfabeto A, B, C, ..., Z o, che è lo stesso, 0, 1, 2, ..., 20 (abbiamo fatto ricorso all'ovvia codifica, o trascrizione, A→0, B→1, C→2, ..., Z→20, in modo da riottenere i simboli che si impiegano normalmente nell'aritmetica circolare modulo 21).

La "sorgente" delle chiavi è ancora un processo stocastico stazionario e senza memoria, ma stavolta le uscite possibili sono 21, tutte equiprobabili (di probabilità 1/21); in linea di principio ci si può servire di una "ruota della fortuna" (di una roulette) divisa in 21 settori uguali numerati da 0 a 20.

In cifratura messaggio e chiave vengono sommati cifra per cifra; per decifrare bisogna *sottrarre* la chiave del crittogramma (nell'aritmetica modulo 21 somma e sottrazione non coincidono più: per esempio $-4=17$ modulo 21 perché $4+17=0$ modulo 21).

CRITTOGRAFIA

Messaggio in lettere:	S	C	O	R	P	I	O	N	E
Messaggio numerico:	16	2	12	15	13	8	12	11	4
Chiave:	4	12	14	19	2	11	7	12	9
Crittogramma:	20	14	5	13	15	19	19	2	13
ossia:	Z	Q	F	P	R	V	V	C	P
Chiave inversa:	-4	-12	-14	-19	-2	-11	-7	-12	-9
ossia:	17	9	7	2	19	10	14	9	12
Messaggio ricostruito:	16	2	12	15	13	8	12	11	4

La codifica da lettere a cifre 21-arie mette in evidenza l'ovvia relazione che lega i cifrari a rotazione del capitolo 1 e le aritmetiche circolari: le rotazioni del disco della figura corrispondono a somme del modulo 21, tant'è che i cifrari a rotazione si chiamano anche cifrari additivi. Non solo, a questo punto siamo in grado di capire perché il cifrario di Vigenère sia un antenato del cifrario a chiave non riutilizzabile, il quale è appunto un cifrario di Vigenère con la parola-chiave casuale e (potenzialmente) infinita. Del resto i crittologi non avevano tardato ad accorgersi che le prestazioni del cifrario di Vigenère diventavano eccellenti al crescere della lunghezza della parola-chiave, e che viceversa le parole-chiavi con un significato ben preciso, facili da tenere a mente, indebolivano il cifrario.

I cifrari m-ari a chiave non riutilizzabile sono perfetti, ma hanno tutti gli inconvenienti di quelli binari. La ricerca di cifrari perfetti meno farraginosi è destinata a rimanere delusa: si può dimostrare che in un cifrario perfetto la "quantità di chiave" dev'essere *almeno* pari alla "quantità di messaggio" che si vuole cifrare; ciò implica che non esistano cifrari perfetti più maneggevoli di quelli illustrati in questo capitolo.

Supponiamo infatti di avere un cifrario perfetto. Ciò significa che per qualunque coppia costituita da un messaggio in chiaro m e da un crittogramma c si ha:

$$\text{Prob}\{M=m, C=c\} = \text{Prob}\{M=m\} \text{ Prob}\{C=c\}$$

(Il messaggio in chiaro e il crittogramma sono stocasticamente indipendenti).

Poiché le probabilità $\text{Prob}\{M=m\}$ e $\text{Prob}\{C=c\}$ sono

entrambe diverse da zero (altrimenti quel messaggio, o quel crittogramma, di fatto non esisterebbero e potrebbero venir cassati senza danno), anche la probabilità congiunta $\text{Prob}\{M=m, C=c\}$ è sempre un numero strettamente maggiore di zero (m e c sono fra di loro compatibili), e dunque deve esistere una chiave che, applicata al crittogramma c , lo decifri a quel messaggio. Ma allora il numero delle chiavi distinte (che operano in maniera diversa quando sono applicate a c) dev'essere almeno pari al numero dei messaggi in chiaro, che è proprio quanto volevamo dimostrare:

Teorema. In un cifrario perfetto il numero delle chiavi è maggiore o uguale al numero dei messaggi in chiaro.

Nel caso del cifrario a chiave non riutilizzabile, il numero delle chiavi di lunghezza n è *esattamente uguale* al numero dei messaggi della stessa lunghezza, ciò che, a norma del teorema, è il meglio che si possa fare. Dunque non esistono cifrari perfetti meno farruginosi di quelli che già conosciamo. Un bel guaio!

7 Cifre casuali e pseudocasuali

Un'idea per ottenere chiavi molto lunghe e “ragionevolmente” casuali è quella di ricorrere a elenchi statistici, raccolte di dati demografici ecc.

Una chiave di tipo decimale si può ottenere dall'elenco telefonico di una certa città e di un certo anno considerando, diciamo, l'ultima cifra dei numeri telefonici degli abbonati (o le ultime due, o le ultime tre; certo non la *prima* cifra se si vuole che la chiave abbia buone proprietà di casualità). Va da sé che una sequenza casuale decimale può essere facilmente trasformata in una sequenza casuale binaria, per esempio sostituendo zero a ogni cifra pari e uno a ogni cifra dispari. Ciò fa intravedere la possibilità di attuare un cifrario a chiave non utilizzabile basato sugli elenchi telefonici, il che, se è un progresso sul lancio della moneta, non è ancora una soluzione molto comoda.

Un'obiezione è spontanea: visto che i calcolatori hanno di

solito degli ottimi programmi per generare numeri casuali, perché non servirsene ed evitare la noia di lanciare monete o di sfogliare elenchi telefonici? L'idea è buona, eppure le cose non sono così semplici: i numeri casuali del calcolatore in realtà sono solo *pseudocasuali* (sono generati in maniera perfettamente deterministica, ma possono venire usati *come se* fossero davvero casuali); nella maggioranza delle applicazioni ciò non ha nessuna importanza, ma il contesto crittografico è così delicato che i normali crittogrammi si rivelano del tutto inadeguati.

Nel prossimo capitolo esamineremo uno fra i più importanti metodi che vengono usati per la generazione di cifre binarie pseudocasuali e vedremo di renderci conto dei suoi difetti sul piano crittografico. Purtroppo, come se si possa ovviare a questi difetti è un problema estremamente complesso, anzi è uno dei problemi fondamentali della ricerca crittografica attuale.

6 Alla ricerca della perfezione perduta

1 Registri a scorrimento con retroazione lineare

In questo paragrafo descriveremo dei meccanismi di generazione di cifre binarie pseudocasuali che, almeno a prima vista, sembrano rispondere molto bene alle esigenze della crittografia. Un *registro a scorrimento lineare* si compone di n elementi di memoria binari, o *stadi*, il cui contenuto può essere la cifra 0 o la cifra 1; n è la *lunghezza* del registro; l'ennupla binaria contenuta nel registro in un dato momento è lo *stato* del registro in quel momento (dunque gli stati possibili di un registro di lunghezza n sono 2^n). A intervalli di tempo regolari il contenuto di ciascuno stadio viene trasferito (“scorre”) nello stadio di sinistra e il contenuto di S_0 “esce” dal registro (si veda la figura 6.1). Anche il contenuto dello stadio più a destra, S_{n-1} , va aggiornato: ciò viene effettuato mediante la *funzione di retroazione* (o di *feedback*) del registro. Tale funzione consiste in un *addizionatore* che è collegato a certi stadi del registro: l’addizionatore somma il contenuto di tali stadi (*prima* che il registro venga aggiornato) e il risultato viene appunto trasferito in S_{n-1} ; la somma di cui parliamo è beninteso quella binaria modulo 2 in cui $1 \oplus 1 = 0$. Le uscite del registro sono costituite dalle cifre binarie che via via occupano lo stadio S_0 . In figura 6.1 è rappresentato due volte lo stesso registro di lunghezza 5; la seconda rappresentazione serve a mostrare che la funzione di retroazione può venire alterata semplicemente aprendo o chiudendo certi contatti.

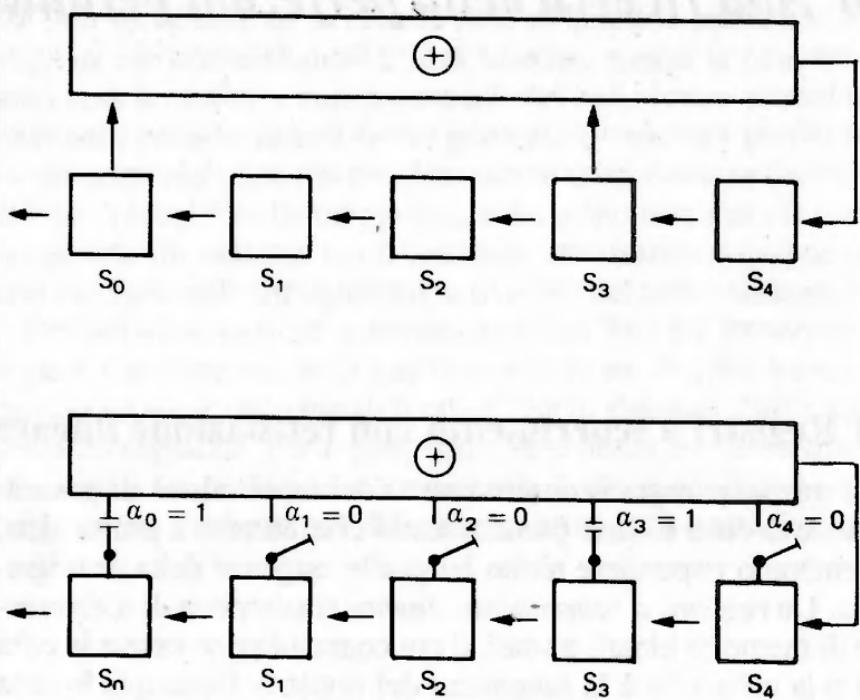


Fig. 6.1 Un registro a scorrimento lineare.

Il contenuto all'istante $t+1$ dello stadio S_{n-1} si può scrivere, in funzione del contenuto degli stadi all'istante t , come:

$$S_{n-1}(t+1) = a_0 S_0(t) + a_1 S_1(t) + \dots + a_{n-2} S_{n-2}(t) + a_{n-1} S_{n-1}(t)$$

dove i coefficienti a_0, a_1, \dots, a_{n-1} sono 1 o 0 a seconda che i rispettivi stadi siano collegati o meno all'addizionatore (nell'esempio della figura $S_4(t+1) = S_0(t) + S_3(t)$). La relazione che abbiamo scritto è appunto, in termini matematici, una relazione di tipo *lineare*; funzioni di retroazione più complesse, in cui per esempio si permettesse anche di *moltiplicare* il contenuto di certi stadi, darebbero luogo a registri a scorrimento *non lineari*.

Beninteso prima di far partire il registro l'utente deve innescarlo inserendo l'ennupla binaria di avvio (lo *stato iniziale* del registro); fatto questo, il registro può continuare a funzionare all'infinito. C'è uno spazio iniziale che è poco saggio usare, quello di solo zero (lo *stato nullo*), perché il registro continuerrebbe a emettere zeri e soltanto zeri, mentre il nostro scopo è quello di ottenere sequenze binarie in cui il numero degli uno e quello degli zero tendano a bilanciarsi.

Tuttavia, anche se si esclude lo stato iniziale nullo, non è affatto intuitivo che esistano registri a scorimento atti a generare sequenze binarie che abbiano accettabili proprietà di casualità (o di pseudocausalità).

Si potrebbe perfino obiettare che un registro a scorimento *non* può produrre un'autentica sequenza pseudocasuale: infatti una buona sequenza pseudocasuale dev'essere "imprevedibile", e dunque *non* può essere *periodica*, ossia non può ripetersi tale quale a intervalli regolari. D'altra parte, tenendo presente che i possibili stati distinti di un registro di lunghezza n sono in numero finito (sono 2^n), si vede che la sequenza che esce dal registro, che è una macchina deterministica, non può che essere periodica: il suo *periodo*, per esser più precisi, è al massimo $2^n - 1$ (il periodo di una sequenza periodica $b_1 b_2 \dots b_n \dots$ è il più piccolo intero P tale che $b_i = b_{i+p}$ per ogni intero i ; abbiamo sottratto 1 a 2^n perché lo stato nullo sicuramente non appare nel corso della generazione di una sequenza a periodo elevato).

Naturalmente $2^n - 1$ è solo una *limitazione superiore*, mentre il periodo della sequenza generata potrebbe essere di fatto molto più corto; nel caso fortunato in cui esso sia proprio $2^n - 1$ il registro si chiama *a periodo massimo*: tutti gli stati possibili, salvo quello nullo, compaiono uno dopo l'altro nel registro. D'altra parte all'aumentare della lunghezza n la crescita di $2^n - 1$ è esponenziale e dunque "esplosiva": un registro a periodo massimo, anche per valori di n relativamente bassi, genera sequenze *praticamente aperiodiche* e ciò permette di superare l'obiezione appena avanzata.

Ma anche così il nostro obiettivo sembra ancora lontano: l'aperiodicità, pratica o teorica che sia, non basta da sola a

garantire la pseudocasualità; oltretutto finora non sappiamo neanche se i registri a periodo massimo davvero esistono.

2 Registri a periodo massimo

L'identificazione dei registri lineari a periodo masimo è un problema algebrico affascinante, che però qui sarebbe troppo lungo trattare: ci limiteremo a elencare i risultati che più ci interessano.

Al registro a scorrimento di lunghezza n con coefficienti di retroazione a_0, a_1, \dots, a_{n-1} (nella figura 6.1 $a_0=a_3=1, a_1=a_2=a_4=0$) si usa associare il suo *polinomio caratteristico* di grado n :

$$p(x) = a_0 + a_1x + a_2x^2 + \dots + a_{n-2}x^{n-2} + a_{n-1}x^{n-1} + x^n$$

(x è una variabile che può assumere i valori 0 e 1; le operazioni vanno eseguite nell'aritmetica modulo 2). La traduzione del nostro problema in termini di algebra dei polinomi non è un mero passo formale, visto che consente di adoperare strumenti matematici studiati da secoli e dunque noti in profondità. Un polinomio che sia quello caratteristico di un registro a periodo massimo si chiama appunto *polinomio a periodo massimo*, o *polinomio primitivo*. Gli algebristi hanno scoperto delle caratterizzazioni dei polinomi primitivi assai pregnanti. A noi basterà sapere che i polinomi primitivi sono "a portata di mano": ce n'è per ogni lunghezza n e si conoscono algoritmi efficienti per generarli; lunghe liste di polinomi primitivi sono pubblicate nella letteratura specializzata. Da parte nostra riportiamo l'elenco di *tutti* i polinomi primitivi di grado al più 8; la presenza del polinomio $1+x^3+x^5$ garantisce che il registro della figura 6.1 abbia proprio periodo massimo, $P=2^5-1=31$.

Il lettore noterà che i polinomi primitivi sono particolarmente numerosi per $n=7$ (i polinomi di ottavo grado sono in totale il doppio di quelli di settimo grado, ma ci sono meno polinomi primitivi di ottavo e di settimo grado).

Non è una coincidenza: la teoria mostra che i polinomi pri-

ALLA RICERCA DELLA PERFEZIONE PERDUTA

mitivi sono percentualmente molto numerosi se n individua un *primo di Mersenne*, vale a dire un numero primo della forma $2^n - 1$: il $7 = 2^3 - 1$ e il $127 = 2^7 - 1$

 TABELLA 6.1 *Polinomi primitivi*

grado	polinomi
2	$1 + x + x^2$
3	$1 + x + x^3$ $1 + x^2 + x^3$
4	$1 + x + x^4$ $1 + x^3 + x^4$
5	$1 + x^2 + x^5$ $1 + x^3 + x^5$ $1 + x^2 + x^3 + x^4 + x^5$ $1 + x + x^2 + x^3 + x^5$ $1 + x + x^2 + x^4 + x^5$ $1 + x + x^3 + x^4 + x^5$
6	$1 + x + x^6$ $1 + x^5 + x^6$ $1 + x + x^2 + x^5 + x^6$ $1 + x + x^4 + x^5 + x^6$ $1 + x^2 + x^3 + x^5 + x^6$ $1 + x + x^3 + x^4 + x^6$
7	$1 + x^3 + x^7$ $1 + x^4 + x^7$ $1 + x + x^2 + x^3 + x^7$ $1 + x^4 + x^5 + x^6 + x^7$ $1 + x^2 + x^3 + x^4 + x^7$ $1 + x^3 + x^4 + x^5 + x^7$ $1 + x + x^2 + x^4 + x^5 + x^6 + x^7$ $1 + x + x^2 + x^3 + x^5 + x^6 + x^7$ $1 + x + x^2 + x^3 + x^4 + x^5 + x^7$ $1 + x^2 + x^3 + x^4 + x^5 + x^6 + x^7$ $1 + x^2 + x^4 + x^6 + x^7$

TABELLA 6.1 *Polinomi primitivi*

grado	polinomi
	$1 + x + x^3 + x^5 + x^7$
	$1 + x + x^7$
	$1 + x^6 + x^7$
	$1 + x + x^3 + x^6 + x^7$
	$1 + x + x^4 + x^6 + x^7$
	$1 + x^2 + x^5 + x^6 + x^7$
	$1 + x + x^2 + x^5 + x^7$
8	$1 + x^2 + x^3 + x^4 + x^8$
	$1 + x^4 + x^5 + x^6 + x^8$
	$1 + x^3 + x^5 + x^6 + x^8$
	$1 + x^2 + x^3 + x^5 + x^8$
	$1 + x + x^2 + x^5 + x^6 + x^7 + x^8$
	$1 + x + x^2 + x^3 + x^6 + x^7 + x^8$
	$1 + x + x^3 + x^5 + x^8$
	$1 + x^3 + x^5 + x^7 + x^8$
	$1 + x^2 + x^5 + x^6 + x^8$
	$1 + x^2 + x^3 + x^6 + x^8$
	$1 + x + x^5 + x^6 + x^8$
	$1 + x^2 + x^3 + x^7 + x^8$
	$1 + x + x^2 + x^3 + x^4 + x^6 + x^8$
	$1 + x^2 + x^4 + x^5 + x^6 + x^7 + x^8$
	$1 + x + x^6 + x^7 + x^8$
	$1 + x + x^2 + x^7 + x^8$

sono appunto primi di Mersenne.

Nella tabella 6.2 sono riportati i *trinomi* primitivi il cui grado dà un primo di Mersenne, fino a n=607.

Si noti che per n=607 il periodo $2^n - 1$ ha l'ordine di grandezza di 10^{183} .

Per avere un'idea di che razza di numero sia 10^{183} , basti dire che 10^{80} è la stima corrente (a livello di ordine di grandezza, va da sé) del numero degli atomi che compongono l'universo.

Se per ipotesi fantasiosa, ognuno di questi atomi fosse esso stesso un sub-universo di 10^{80} sub-atomi, il numero totale dei sub-atomi dell'universo sarebbe $10^{80} \times 10^{80}$, ossia 10^{160} .

TABELLA 6.2 *Trinomi primitivi*

$x^{17} + x^k + 1$	$k = 3, 5, 6, 11, 12, 14$
$x^{31} + x^k + 1$	$k = 3, 6, 7, 13, 18, 24, 25, 28$
$x^{89} + x^k + 1$	$k = 38, 51$
$x^{127} + x^k + 1$	$k = 1, 7, 15, 30, 63, 64, 97, 112, 120, 126$
$x^{521} + x^k + 1$	$k = 32, 48, 158, 168, 353, 363, 473, 489$
$x^{607} + x^k + 1$	$k = 105, 147, 273, 334, 460, 502$

3 Proprietà di casualità

Abbiamo risolto il problema di costruire registri a periodo “praticamente infinito”: basterà usare la tabella 6.2 oppure procurarsi una lista di polinomi primitivi più esauriente della nostra.

Non siamo ancora giunti in porto: il fatto che una sequenza binaria abbia periodo massimo non basta a dichiararla pseudocasuale se non si riesce a dimostrare che essa gode di proprietà di casualità adeguate. Di tali proprietà noi illustreremo solo la più semplice.

A priori *qualsiasi* sequenza di zero e di uno può venire generata lanciando una moneta non tarata: tuttavia la *legge dei grandi numeri* assicura che, *quasi certamente*, la sequenza avrà una struttura “imprevedibile”: il numero degli zero tenderà a bilanciarsi col numero degli uno e più in generale la frequenza relativa con cui nella sequenza comparirà una certa “configurazione” di k cifre binarie tenderà a stabilizzarsi attorno al valore $1/2^k$, che è lo stesso per tutte le 2^k “configurazioni” di lunghezza k (si ricordi che $1/2^k$ è la probabilità di una qualsiasi di queste configurazioni).

Allora una prima e immediata proprietà di pseudocasualità cui dovrebbero soddisfare le nostre sequenze pseudocasuali è che gli zero e gli uno che compaiono in un periodo completo di $2^n - 1$ cifre siano in numero “circa” uguale. Ciò in effetti è vero, come ora proveremo.

Se il periodo è massimo *tutte* le configurazioni di n cifre binarie compaiono nel registro (costituiscono a turno lo stato

del registro), fatta eccezione per quella nulla di soli zero.

Se pensiamo allo stato del registro come a un numero intero scritto in base 2 (si riveda il paragrafo 4 del capitolo 4) possiamo anche dire che tutti i numeri interi da $1=(00\dots01)_2$ a $2^n-1=(11\dots11)_2$ compaiono nel registro, anche se non nel loro ordine naturale (la lunghezza delle rappresentazioni binarie è stata uniformata a n aggiungendo, là dove occorreva, degli zero in testa). 1 e 2^n-1 sono due numeri dispari e il numero dei dispari fra 1 e 2^n-1 è superiore di un'unità a quello dei pari: ma un numero è pari o dispari a seconda che nella sua rappresentazione binaria l'ultima cifra sia 0 oppure 1 .

Ciò significa che in un periodo completo il numero degli zero che transitano nello stadio S_{n-1} è di un'unità inferiore a quello degli uno; d'altronde le cifre che transitano per S_{n-1} sono quelle stesse che, con un ritardo di $n-1$ istanti di tempo, transitano per S_0 , ossia quelle che costituiscono le uscite del registro.

Si potrebbe dimostrare che proprietà analoghe valgono per le coppie di zero e di uno, per le terne ecc.

I risultati sono perfino "troppo esatti": il fatto che su un periodo che potrebbe essere lunghissimo lo scarto fra il numero degli zero e quello degli uno sia così contenuto potrebbe indurci a dire che le nostre sequenze pseudocasuali sono troppo casuali... per esserlo davvero!

Senza voler fare i sofistici, possiamo ormai affermare di essere in grado di costruire un programma per generare cifre binarie pseudocasuali: non è un programma a scatola chiusa, siamo in grado di capire con chiarezza quello che sta accadendo, fatto indispensabile in crittografia dove non si può dare nulla per scontato.

Nel programma 6.1 si simula un registro di lunghezza n ; A è l'ennupla dei coefficienti a_i , S è l'ennupla che specifica lo stato iniziale.

Aiutandoci con le tabelle 6.1 e 6.2 possiamo far sì che il registro abbia periodo massimo.

Programma 6.1 Algoritmo *reglin*(n, A, S, k)

inizial "simula un registro di n stadi per k istanti"

*reglin*₁:=s₀ "prima uscita"

per j:=2 a k esegui

inizial

x:=0

per i:=0 a n-2 esegui

inizial

x:=x+a_i*s_i mod 2; s_i:=s_{i+1}

fine

s_{n-1}:=x+a_{n-1}*s_{n-1} mod 2

*reglin*_j:=s₀ "uscita all'istante j"

fine

fine

4 Uso crittografico dei registri lineari

A prima vista i registri lineari sembrano assai adatti ai nostri scopi: gli utenti legittimi possono generare la stessa sequenza binaria pseudocasuale purché abbiano una copia ciascuno dello stesso registro a scorrimento e lo inneschino con la stessa ennupla iniziale; questa ennupla ovviamente deve restare segreta poiché è la *vera chiave* del cifrario (un confronto superficiale con i cifrari a chiave non riutilizzabile potrebbe far pensare che la chiave sia costituita dalla sequenza pseudocasuale di periodo $2^n - 1$, ma questa, per così dire, è solo la *chiave apparente*). I gravi inconvenienti di cui soffrivano i cifrari perfetti, le dimensioni eccessive della chiave e la difficoltà di generarla, sono completamente scomparsi.

Naturalmente la pseudocasualità è solo un'approssimazione della vera casualità: il nostro è ora un cifrario *pseudoperfetto*, le cui prestazioni si spera approssimino quelle dei cifrari perfetti. Purtroppo non è così: i cifrari pseudoperfetti basati sui registri a scorrimento con retroazione lineare sono *del tutto inadeguati* alle esigenze della crittografia odierna, come ora mostreremo.

5 Attacchi crittanalitici con testo in chiaro

Finora nel valutare i rischi che un cifrario venisse forzato abbiamo supposto che il crittanalista potesse avere accesso al canale poco sicuro lungo cui è trasmesso il crittogramma, ma non al congegno di cifratura usato dal crittografo. Supponiamo invece che egli possa osservare una porzione di messaggio in chiaro assieme al crittogramma corrispondente: la coppia messaggio-crittogramma potrebbe essergli sufficiente per capire qual è la chiave che è stata usata e dunque per decifrare tutti i crittogrammi futuri che sarà in grado di intercettare, perlomeno finché la chiave non verrà cambiata. Si può pensare, per esempio, che la spia abbia accesso, sia pure per breve tempo, ai depositivi di cifra e ne approfitti per inserire un suo messaggio e prender nota del crittogramma in uscita. Oggi le operazioni di cifratura si svolgono spesso in ambienti, come i centri di calcolo, che non possono essere sorvegliati con metodi polizieschi, per cui il rischio che si verifichi un simile incidente non è affatto remoto. Per valutare la bontà di un cifrario bisogna dunque studiare anche la sua capacità di resistenza agli *attacchi con testo in chiaro*, che sono basati appunto su coppie messaggio-crittogramma.

Purtroppo da questo punto di vista i cifrari pseudoperfetti basati su registri a scorrimento lineari sono totalmente indifesi: bastano $2n$ cifre di messaggio e le $2n$ cifre del crittogramma corrispondente perché il crittanalista sia in grado di forzare il cifrario in maniera completa.

Vediamo di capire con un esempio come ciò sia possibile. Torniamo alla relazione lineare del paragrafo 1 e ripensiamo al meccanismo di scorrimento a sinistra del registro: $S_1(t)$, il contenuto dello stadio S_1 all'istante t , è anche $S_o(t+1)$, che il contenuto dello stadio S_o all'istante successivo; $S_2(t)$ coincide con $S_1(t+1)$ e con $S_o(t+2)$, e così via. La relazione lineare si può allora riscrivere tutta in termini di S_o :

$$S_o(t+n) = a_o S_o(t) + a_1 S_o(t+1) + a_2 S_o(t+2) + \dots + a_{n-2} S_o(t+n-2) + a_{n-1} S_o(t+n-1) \quad (6.1)$$

Dunque il contenuto di S_o in un dato istante (la cifra binaria usata in cifratura in quell'istante) è una *combinazione lineare* dei contenuti di S_o negli n istanti precedenti.

Supponiamo, per essere più concreti, che il registro abbia lunghezza 5 e che il crittanalista abbia intercettato il messaggio $m_{12}m_{13}m_{14}\dots m_{20}m_{21}$ e il crittogramma $c_{12}c_{13}c_{14}\dots c_{20}c_{21}$ (gli istanti di tempo sono t=12, 13, 14, ..., 20, 21). Poiché il crittogramma si calcola sommando la chiave al messaggio cifra per cifra, si ha:

$$c_{12} = m_{12} + S_o(12), c_{13} = m_{13} + S_o(13), \dots, c_{21} = m_{21} + S_o(21)$$

Ciò consente al crittanalista di calcolare $S_o(12)$, $S_o(13)$, ..., $S_o(21)$. Per t=12, 13, ..., 16 la relazione (6.1) si particolarizza in:

$$\begin{aligned} S_o(17) &= a_0S_o(12) + a_1S_o(13) + a_2S_o(14) + a_3S_o(15) + a_4S_o(16) \\ S_o(18) &= a_0S_o(13) + a_1S_o(14) + a_2S_o(15) + a_3S_o(16) + a_4S_o(17) \\ S_o(19) &= a_0S_o(14) + \dots + a_4S_o(18) \\ S_o(20) &= a_0S_o(15) + \dots + a_4S_o(19) \\ S_o(21) &= a_0S_o(16) + \dots + a_4S_o(20) \end{aligned}$$

Queste cinque equazioni lineari consentono di trovare i valori di a_0 , a_1 , a_2 , a_3 e a_4 (nel nostro caso si può fare a mano, ma comunque esistono algoritmi assai efficienti per risolvere sistemi lineari): è come se il crittanalista avesse forzato il registro e avesse potuto controllare quali sono i collegamenti operanti. Grazie alla relazione (6.1) egli è ormai in grado di calcolare $S_o(22)$, $S_o(23)$, e così via all'infinito: il cifrario è stato forzato, la spia può ricostruire il messaggio a partire dal solo crittogramma.

Prima di chiudere questo paragrafo vogliamo fare qualche osservazione sulle chiavi dei cifrari basati su registri di scorrimento. Si è già detto che la vera chiave è l'ennupla con cui il cifrario viene innescato: in realtà stavolta conviene parlare di *due livelli* di chiavi. Ricordiamoci che i collegamenti di un registro di lunghezza n possono essere facilmente alterati (figura 6.1): di fatto, con un unico oggetto, abbiamo a disposizione tanti cifrari pseudoperfetti quanti sono i polinomi primitivi di grado n. Possiamo allora immaginare che ci sia anche una

“chiave” (un’informazione segreta) di “livello alto” che viene alterata di rado e che specifica quali collegamenti sono aperti e quali sono chiusi (specificando il polinomio primitivo che è stato scelto come polinomio caratteristico del registro). Questa chiave, per esempio, potrebbe venire attivata nel momento in cui il cliente acquista il registro, ed esser dunque una *chiave personale* di quel cliente. La chiave di “livello basso”, che viene alterata di frequente, specifica invece lo stato iniziale che serve a innescare il registro: essa è una *chiave di lavoro*. Purtroppo l’attacco con testo in chiaro che abbiamo descritto porta a identificare entrambe le chiavi, cosa che ribadisce l’inadeguatezza crittografica dei registri lineari.

6 Uno sguardo nel buio: i registri non lineari

Forse il lettore si chiederà perché abbiamo parlato tanto di registri lineari, che in crittografia sembra non servano a niente.

Questi registri sono troppo regolari, troppo ben strutturati, abbiamo invece bisogno di qualcosa di non lineare, di caotico... Non è così semplice; l’idea di generare cifre pseudocausalì con meccanismi caotici e complicatissimi è allettante, ma va vista con sospetto.

L’esperienza dimostra che in crittografia bisogna procedere coi piedi di piombo: il crittografo deve capire in profondità i meccanismi che mette in moto per poter garantire la loro validità, altrimenti si rischia che ciò che a lui appare confuso sia invece cristallino agli occhi del crittanalista. Bisogna dunque mirare a meccanismi di generazione che non siano lineari, ma contemporaneamente abbiano una struttura che può essere compresa a fondo.

Diciamo subito che la posta in gioco è alta, tanto alta che presumibilmente molti risultati sui registri non lineari sono segreti e gelosamente custoditi.

Vediamo comunque di capire quale potrebbe essere una via di uscita.

Poiché comprendiamo bene i registri lineari ma abbiamo bisogno di un sistema di generazione non lineare, un’idea può

essere quella di combinare registri lineari in maniera non lineare (è proprio per questo motivo che i registri lineari, nonostante tutto, *sono* importanti in crittografia). Si consideri per esempio lo schema di figura 6.2.

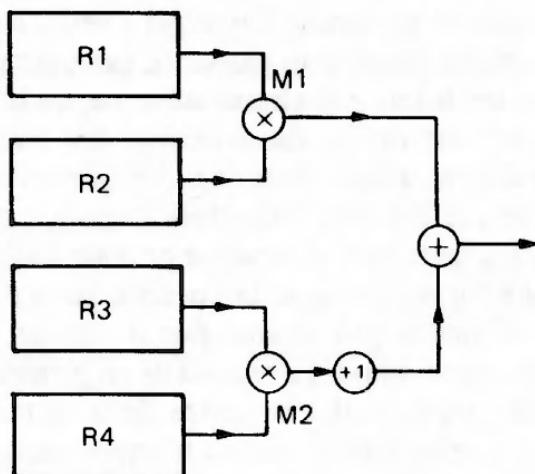


Fig. 6.2

R_1, R_2, R_3 e R_4 sono quattro registri lineari sincroni; le uscite di R_1 e R_2 vengono moltiplicate dal moltiplicatore M_1 e quelle di R_3 e R_4 dal moltiplicatore M_2 .

Visto che 1 si ottiene come prodotto di $1 \oplus 1$, mentre 0 si ottiene come prodotto di $0 \oplus 0$, $0 \oplus 1$ e $1 \oplus 0$, le uscite dei moltiplicatori sono squilibrate a favore dello 0. Per riaggiustare le cose a ciascuna uscita di M_2 viene sommata la costante 1 in modo da invertirla ($0 \oplus 1 = 1$, $1 \oplus 1 = 0$); il risultato, che ora è squilibrato a favore dell'1, viene sommato all'uscita di M_1 .

Abbiamo davvero trovato un generatore di cifre pseudocasuali adeguato ai fini crittografici? In realtà che questo sistema funzioni è ancora tutto da dimostrare. Tanto per dare un esempio banale, se i quattro registri fossero uguali (e finora non lo abbiamo escluso), e se fossero stati innescati con la stessa sequenza d'avvio, il sistema non funzionerebbe affatto, perché,

come il lettore potrà verificare, le uscite sarebbero tutte uguali a 1. In qualche modo, ma solo confusamente, si intuisce che i registri devono essere “casuali l’uno rispetto all’altro”, e dunque non devono affatto rassomigliarsi. Si potrebbe costruirli di lunghezza diversa; presumibilmente è vantaggioso scegliere come lunghezze quattro numeri primi fra loro. Quanto casuali saranno le sequenze generate? Tanto per cominciare possiamo ricorrere a controlli statistici di casualità, per esempio confrontare il numero degli zeri e degli uni su sequenze lunghe, oppure, e meglio, servirci di uno dei numerosi test escogitati dagli statistici (test del chi quadro, test di Kolmogorov-Smirnov, *run test* delle tratte ecc.). Anche i controlli statistici, tuttavia, non sono affatto conclusivi. Se si riesce a provare che un sistema *non* ha certi difetti, per esempio la linearità, resta pur sempre il dubbio che un’analisi più attenta possa rilevare defezienze diverse ma altrettanto gravi. La ricerca di un generatore di cifre pseudocasuali adeguato alle esigenze della crittografia è un problema la cui soluzione avrebbe un’importanza cruciale: la strada da percorrere, tuttavia, è quanto mai insidiosa.

7 Cifrari a blocco e cifrari a flusso

Il lettore ha ormai conosciuto sia esempi di *cifrari a blocco*, come il DES, sia esempi di *cifrari a flusso*, come quelli di questo capitolo. In un cifrario a blocco i dati binari vengono suddivisi in *blocchi* di lunghezza fissa (64 cifre nel caso del DES) e a ciascun blocco viene applicata la trasformazione determinata dalla chiave. Nel caso di un cifrario a flusso la chiave viene usata per generare appunto un “flusso” ininterrotto di cifre binarie che vengono sommate, cifra per cifra, a quelle emesse dalla sorgente dei messaggi.

Esistono anche modi più fantasiosi di usare cifrari a blocco e a flusso; per esempio nella figura 6.3 è illustrato un sistema in cui un cifrario a blocco viene impiegato per costruire un cifrario a flusso. La chiave è doppia; la prima parte serve a innescare un registro a scorrimento a n stadi: le n cifre bianarie che compongono lo stato del registro vengono cifrate mediante un

cifrario a blocco (n , dunque, è anche la lunghezza di blocco).

La seconda parte della chiave globale è appunto la chiave del cifrario a blocco. L'uscita di quest'ultimo viene sommata cifra per cifra al messaggio binario e produce il crittogramma.

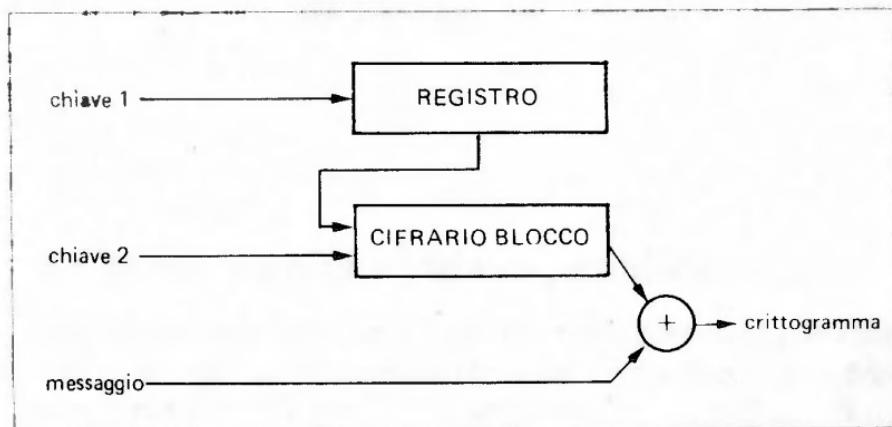


Fig. 6.3

Si possono immaginare configurazioni ancora più ricche, in cui agiscano complessi meccanismi di retroazione: lo scopo, va da sé, è quello di migliorare le prestazioni delle configurazioni più semplici. Purtroppo, lo ribadiamo ancora una volta, maggiore complicazione non implica necessariamente maggiore sicurezza.

7 Crittografia a chiave pubblica

1 Chiave segreta e chiave pubblica

Già nel capitolo dedicato al Data Encryption Standard abbiamo visto che oggi la crittografia deve far fronte a esigenze completamente nuove legate all'impiego sempre più diffuso del calcolatore. Può destare meraviglia che le risposte che la crittografia ha saputo trovare, perlomeno quelle che abbiamo illustrato fino a questo punto, siano tutto sommato di tipo tradizionale. Ciò è vero per il DES, ma in una certa misura è anche vero per il cifrario a chiave non riutilizzabile e per le sue approssimazioni pseudoperfette: i cifrari polialfabetici basati su una parola-chiave, di cui i cifrari perfetti e pseudoperfetti sono i legittimi discendenti, erano già stati studiati dall'Alberti in pieno Rinascimento. (L'età veneranda delle soluzioni non implica affatto che oggi esse non siano più valide: i cifrari pseudoperfetti e i problemi connessi di generazione di cifre binarie pseudocasuali rappresentano probabilmente il punto nodale della crittografia contemporanea.)

Tutti i sistemi di cifra fin qui esposti presupponevano che la chiave di cifratura fosse tenuta rigorosamente segreta. In questo capitolo illustreremo dei sistemi che costituiscono un'innovazione radicale rispetto al passato: in essi la chiave di cifratura è un'informazione divulgata pubblicamente, ad esempio stampandola su una specie di elenco telefonico dove accanto al nome di ciascun utente è riportata la chiave che deve usare chi voglia trasmettergli un messaggio riservato.

In tal modo si supera un problema gravissimo della crittografia tradizionale, quello della *distribuzione* delle chiavi.

Quanto abbiamo detto sembra paradossale, ma non lo è: si possono escogitare sistemi crittografici in cui la chiave di cifratura e quella di decifrazione sono “essenzialmente” diverse, per cui basta tener segreta quest’ultima.

In tutti i sistemi crittografici classici, a chiave segreta, la chiave di cifratura e quella di decifrazione sono “essenzialmente” uguali, nel senso che, anche se non coincidono, la seconda può venire immediatamente ricavata dalla prima: per esempio in un cifrario a sostituzione le due chiavi si avvalgono della stessa permutazione, letta in un senso in cifratura e nell’altro in decifrazione.

Nei sistemi a chiave pubblica l’operazione di ricavare la chiave di decifrazione da quella di cifratura è *trop*po complessa per venire eseguita in pratica.

Ma stiamo anticipando: i principi generali saranno più chiari quando avremo esposto qualche esempio specifico.

Per non suscitare aspettative eccessive nel lettore diciamo subito che la crittografia a chiave pubblica non è esente da critiche: le speranze che in essa si ripongono sembrano tuttavia ben giustificate.

2 Le funzioni unidirezionali e il logaritmo finito

Nella crittografia a chiave pubblica hanno grande importanza le cosiddette *funzioni unidirezionali*: si tratta di funzioni invertibili tali che il calcolo della funzione diretta sia “facile”, mentre quello della funzione inversa sia “difficile”.

Anche qui conviene passare subito a un esempio concreto. Abbiamo già menzionato nel paragrafo 5 del capitolo 5 i campi di Galois; quelli più semplici si ottengono considerando le aritmetiche circolari modulo un numero primo.

Nella tabella 7.1 abbiamo riportato la somma e il prodotto di \mathbb{Z}_7 , ossia del campo di Galois degli interi modulo 7 (delle cifre 7-arie 0,1,2,3,4,5,6).

TABELLA 7.1 *La somma e il prodotto di Z₇*,

\oplus	0	1	2	3	4	5	6	\oplus	1	2	3	4	5	6
0	0	1	2	3	4	5	6	1	1	2	3	4	5	6
1	1	2	3	4	5	6	0	2	2	4	6	1	3	5
2	2	3	4	5	6	0	1	3	3	6	2	5	1	4
3	3	4	5	6	0	1	2	4	4	1	5	2	6	3
4	4	5	6	0	1	2	3	5	5	3	1	6	4	2
5	5	6	0	1	2	3	4	6	6	5	4	3	2	1
6	6	0	1	2	3	4	5							

Nella tabella del prodotto la riga e la colonna dello 0 sono state omesse perché fatte di soli zero. In un campo di Galois è ovviamente possibile parlare di potenze; per esempio, in Z_7 , $3^0=1$, $3^1=3$, $3^2=3 \times 3=2$, $3^3=3^2 \times 3=2 \times 3=6$, $3^4=3^3 \times 3=4$, $3^5=3^4 \times 3=5$, $3^6=3^5 \times 3=1=3^0$, $3^7=3=3^1$, $3^8=2=3^2$, $3^9=3=3^3$ e così via circolarmente. Come si vede il 3 con tutte le sue sei potenze distinte produce *tutti* gli elementi non nulli del campo: per questa ragione il 3 viene detto un *elemento primitivo* del campo di Galois Z_7 . Se invece del 3 prendiamo il 4 abbiamo $4^0=1$, $4^1=4$, $4^2=2$, $4^3=1=4^0$, $4^4=4=4^1$, $4^5=2=4^2$, e così via circolarmente: poiché le sue potenze distinte sono in numero insufficiente il 4 *non* è un elemento primitivo di Z_7 . Il lettore potrà verificare che oltre al 3 c'è un altro elemento primitivo in Z_7 , il 5. In un campo di Galois, dopo aver fissato come *base* un elemento primitivo B , possiamo considerare la *funzione esponenziale* $y=B^x$ (l'algebra assicura che *tutti* i campi di Galois possiedono elementi primitivi). Nel caso di Z_7 , scegliendo la base $B=3$, la funzione $y=3^x$ trasforma gli esponenti 0,1,2,3,4,5 nelle cifre 7-arie 1,3,2,6,4,5, in quest'ordine (si legga la tabella 7.2 da sinistra verso destra). Se la tabella 7.2 viene letta da destra verso sinistra, si ottiene una trasformazione che *inverte* la funzione esponenziale $y=3^x$: alle cifre 7-arie 1, 2, 3, 4, 5 e 6 essa associa gli esponenti 0, 2, 1, 4, 5 e 3.

Il *logaritmo* in base B , $x=\log_B y$, è appunto la funzione inversa della funzione esponenziale $y=B^x$.

TABELLA 7.2 *L'esponenziale e il logaritmo in base 3 su Z*

$y=3^x$	$0 \leftrightarrow 1$ $1 \leftrightarrow 3$ $2 \leftrightarrow 2$ $3 \leftrightarrow 6$ $4 \leftrightarrow 4$ $5 \leftrightarrow 5$	$x=\log_3 y$ \leftarrow
---------	----------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------

Nel nostro caso, data la piccolezza del modulo, non ci sono problemi né per calcolare la funzione diretta né per calcolare quella inversa: basta usare la tabella 7.2.

Diversa sarebbe la situazione se il modulo fosse un numero primo enorme, dell'ordine di grandezza, diciamo, di 10^{50} : non solo la costruzione della tabella, ma perfino la sua consultazione diverrebbe un'operazione di complessità proibitiva.

Tranne che per i valori più bassi del modulo abbiamo dunque bisogno di *algoritmi* ingegnosi che ci permettano di risparmiare tempo e fatica.

Da questo punto di vista la situazione è estremamente asimmetrica: mentre esistono algoritmi assai efficienti per il calcolo dell'esponenziale (delle potenze), quelli per il calcolo del logaritmo non ci consentono scorciatoie altrettanto decisive rispetto al controllo esauriente implicito in una tabella come la 7.2, tabella che di per sé costituisce un algoritmo di calcolo, sia pure assai poco ingegnoso, dell'esponenziale e del logaritmo.

Per essere un po' più precisi, l'algoritmo più veloce che si conosca per il calcolo del logaritmo finito richiede un numero di "passi" (operazioni elementari) circa pari a $e^{\sqrt{\ln p \ln(\ln p)}}$ (l'esponenziale e il logaritmo \ln sono in base naturale $e=2,718\dots$; p è il modulo del campo di Galois).

Se p è un numero primo di una cinquantina di cifre decimali, la formula restituisce un valore di dieci cifre decimali.

Se si ipotizza di usare un calcolatore che effettui un passo ogni nanosecondo (un miliardo di passi al secondo), il computo del logaritmo richiederebbe un tempo dell'ordine del secon-

do; ma se p di cifre decimali ne avesse duecento il numero dei passi salirebbe a oltre 10^{23} , e ciò terrebbe occupato il nostro calcolatore per milioni di anni!

Per calcolare l'esponenziale esiste invece un algoritmo la cui complessità è grosso modo pari a $\log_2 p = \ln p / \ln 2 = \log_{10} p / \log_{10} 2$; per $p=10^{200}$ si trova $\log_2 p = 200 / \log_{10} 2 = 166$, che corrisponde a un tempo di calcolo di neppure un secondo.

Nel prossimo paragrafo vedremo come il logaritmo finito possa venire usato in crittografia.

Non parleremo ancora di cifrari a chiave pubblica: il problema che tratteremo, quello della distribuzione delle chiavi, è comunque di importanza cruciale; esso sorge, beninteso, quando si adopera un cifrario tradizionale a chiave segreta, per esempio il DES.

A conclusione del paragrafo presentiamo un algoritmo rapido per il calcolo delle potenze modulo un intero n , che non occorre neppure sia primo (la primalità del modulo interveniva solo quando si parlava di elementi primitivi e di logaritmi).

L'idea di base è semplice: quando si deve calcolare un prodotto nell'aritmetica circolare si possono ridurre modulo n i due fattori, moltiplicarli così ridotti, e poi ridurre modulo n il risultato (un'osservazione simile vale anche per la somma); ciò consente di evitare nei calcoli valori intermedi ingombranti. Per esempio

$$\begin{aligned} 7^5 \text{ modulo } 11 &= [(7^2 \text{ modulo } 11) \times (72 \text{ modulo } 11) \times 7] \text{ modulo } 11 \\ &= [5 \times 5 \times 7] \text{ modulo } 11 = [(25 \text{ modulo } 11) \times 7] \text{ modulo } 11 \\ &= [3 \times 7] \text{ modulo } 11 = 10. \end{aligned}$$

(i resti delle divisioni $49:11$, $25:11$ e $21:11$ sono rispettivamente 5 , 3 e 10).

In realtà 7^5 non è poi così ingombrante e possiamo eseguire per verifica anche il calcolo diretto: $7^5 = 16807 = 1527 \times 11 + 10 = 10$ modulo 11 .

Nel programma 7.1 `div` indica la divisione intera con troncamento; per esempio $6 \text{ div } 3 = 7 \text{ div } 3 = 8 \text{ div } 3 = 2$.

Programma 7.1 Un algoritmo per il calcolo delle potenze nell'aritmetica modulo n. Algoritmo *espon*(b,x,n)

```

inizial "calcola y=bx mod n"
    b1:=b; x1:=x; y:=1
    finché x1≠0 esegui
        inizial
            finché x1 mod 2 = 0 esegui
                inizial "fa il quadrato di b1 finché x1 è pari"
                    x1:=x1 div 2; b1:=(b1*b1) mod n
                fine
                x1:=x1-1; y:=(y*b1) mod n "moltiplica"
            fine
        espon:=y
    fine

```

3 Il logaritmo finito e la distribuzione delle chiavi

Supponiamo che gli utenti di una certe "rete" abbiano la necessità di scambiarsi informazioni riservate; supponiamo anche che essi usino un cifrario a chiave segreta. Se gli utenti sono N, le coppie di utenti sono $N(N-1)/2$: ciò significa che occorrono circa $N^2/2$ canali speciali per la trasmissione delle chiavi. Ciascun canale va protetto non solo dagli estranei, ma anche dagli $N-2$ utenti che non sono direttamente coinvolti nello scambio di informazioni.

Il problema della distribuzione delle chiavi è dunque particolarmente importante nella crittografia destinata a proteggere le reti con molti utenti, che è tipicamente la crittografia commerciale: ha poco senso far uso di cifrari raffinati se il sistema di distribuzione delle chiavi non è fidato.

Si può allora procedere come segue: si fissa un campo di Galois Z_p e si sceglie in esso un elemento primitivo B ; fin qui non c'è nulla di segreto.

Gli utenti della rete scelgono ciascuno un esponente e calcolano B elevato a quell'esponente: il valore dell'esponente viene

tenuto segreto, mentre il risultato dell'elevamento a potenza viene reso noto in un apposito elenco accessibile a tutti. Supponiamo che gli utenti X e Y vogliano comunicare fra di loro: per farlo devono prima accordarsi su una chiave. Se i due esponenti segreti X e Y sono rispettivamente x e y , X e Y possono usare come chiave il valore B^{xy} : questo valore è calcolabile sia da X, nella forma $(B^y)^x$, sia da Y, stavolta nella forma $(B^x)^y$ (si ricordi che B^x e B^y stanno sull'elenco; l'esponenziale finito gode delle stesse proprietà algebriche di quello ordinario, per cui $(B^y)^x = (B^x)^y = B^{xy}$). Fin qui sono state calcolate solo delle potenze, mai dei logaritmi. In linea si principio chiunque può calcolare B^{xy} : basta ricavare la x (o la y) dall'informazione pubblica B^x (o B^y); tuttavia per farlo bisogna calcolare un logaritmo, e ciò è inattuabile se il modulo p è abbastanza grande.

Il sistema di distribuzione delle chiavi sembra funzionare assai bene: è vero che gli esponenti vanno tenuti segreti, ma non c'è nessuna necessità di farli circolare, e dunque *non c'è nessun bisogno di canali speciali*. Il fatto che la chiave del DES sia una "parola" binaria mentre B^{xy} è l'elemento di un campo di Galois non crea difficoltà perché possiamo sempre trasformare la chiave in binario, aggiungendo eventualmente degli zero in testa. Va anche detto che l'insieme delle 2^n parole binarie di lunghezza n (n fissato ma arbitrario) può venire direttamente strutturato in modo da formare un campo di Galois: in esso la somma è quella binaria, cifra per cifra, mentre il prodotto è un'operazione piuttosto complessa che qui non intendiamo descrivere; ci limiteremo a dire che le nozioni del paragrafo 2 conservano la loro validità.

Per evitare fraintendimenti, sottolineiamo che l'aritmetica circolare modulo 2^n *non* fornisce un campo di Galois se n è maggiore di 1.

Purtroppo (c'era da aspettarselo!) il sistema di distribuzione delle chiavi basato sul logaritmo finito presenta qualche inconveniente; piuttosto che imbarcarci subito in una discussione critica, preferiamo esporre prima in dettaglio il funzionamento di due dei più noti sistemi di cifra a chiave pubblica: i tre sistemi, di distribuzione e di cifra, hanno infatti difetti comuni.

I paragrafi 4 e 5 sono “stellati”, ossia possono venir omessi in prima lettura. Il cifrario del fusto, cui essi sono dedicati, è molto istruttivo, ma i suoi meriti sono ormai più che altro storici e didattici: grazie a tecniche riposte, Adi Shamir ed Ernest Brickell sono riusciti a scardinarlo (l’opera di scardinamento è durata due anni, dal 1982 all’84), e hanno così conseguito uno dei più brillanti successi della ricerca crittanalitica contemporanea.

4 Il problema del fusto

In questo paragrafo, come nel 2, parleremo non di crittografia ma di matematica finita e di logaritmi.

Supponiamo che siano stati assegnati n numeri interi positivi, a_1, a_2, \dots, a_n , e un intero c che sia la somma di alcuni degli n numeri. Per esempio potrebbe essere $a_1=3, a_2=7, a_3=5, a_4=9, a_5=8, a_6=11$, e $c=a_2 + a_4 + a_6=27$; per ragioni che appariranno chiare fra poco è comodo scrivere c nella forma

$$\begin{aligned} c &= a_1 \times 0 + a_2 \times 1 + a_3 \times 0 + a_4 \times 1 + a_5 \times 0 + a_6 \times 1 \\ &= (a_1, a_2, a_3, a_4, a_5, a_6) (0, 1, 0, 1, 0, 1) \end{aligned}$$

Nell’ultima riga si è implicitamente definito il *prodotto scalare* di due ennuple $A=(a_1, a_2, a_3, \dots, a_6)$ e $M=(m_1, m_2, m_3, \dots, m_6)$: nel caso specifico $n=6$, $A=(3, 7, 5, 9, 8, 11)$ e $M=(0, 1, 0, 1, 0, 1)$; si può anche scrivere, in maniera compatta, $c=AM$; si noti che nel prodotto scalare i moltiplicandi sono due ennuple numeriche, e il risultato è appunto uno *scalare*, ossia un numero singolo.

Il problema che ci poniamo è il seguente: dati c e A , ricostruire M , ossia, identificare gli addendi provenienti da A la cui somma è c . Per capire dove interviene il fusto, si pensi a questa comoda immagine: è dato un fusto cilindrico molto alto, e un gruppo di n oggetti di altezza $a_1, a_2, a_3, \dots, a_n$ che possono venire infilati nel fusto uno sopra l’altro. Alcuni di questi oggetti sono stati infilati nel fusto: non si possono più estrarre, ma si può controllare il livello a cui arrivano (figura 7.1).

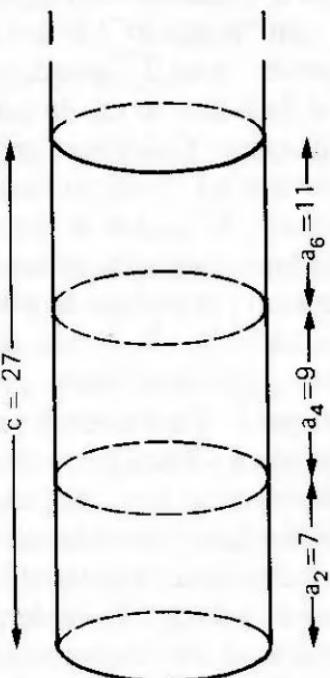


Fig. 7.1

Naturalmente il problema che ci siamo posti potrebbe avere più soluzioni: in questo caso basta trovarne una qualunque. Un algoritmo molto banale per risolvere il problema del fusto è ancora quello della ricerca esauriente: si verificano tutte le ennuple binarie M finché non ci si imbatte in una per la quale sia proprio $c=AM$; *per male che vada*, bisogna fare al più 2^n "passi" (2^n controlli), tanto quante sono le ennuple binarie (sulle parole in corsivo dovremo tornare più avanti). Esistono algoritmi più veloci, ma, almeno finora, non si è riusciti a trovare scorciatoie davvero efficaci: l'algoritmo migliore ricerca circa $2^{n/2}$ passi, quantità che comunque cresce esponenzialmente ("esplosa") al crescere di n . Ciò significa che il problema del fusto è *praticamente irrisolvibile*, a meno che n non sia molto piccolo. Bisogna però fare qualche precisazione. Dicendo che l'algoritmo banale ha *complessità* 2^n e che quello intelligente

ha *complessità* circa 2^{n^2} intendiamo semplicemente dire che esistono dei casi (i casi "peggiori") in cui occorre effettuare 2^n passi o, rispettivamente, circa 2^{n^2} passi: non è affatto escluso che esistano dei casi fortunati in cui la soluzione si possa trovare molto più rapidamente. Ciò è vero anche per il calcolo del logaritmo finito: per fare un esempio limite il logaritmo di 1, che è 0, si calcola immediatamente. E' tipico della *teoria della complessità* valutare la complessità di un algoritmo in base al numero di passi necessari per trovare la soluzione *nel caso peggiore*: il lettore già intuirà che ciò è fonte di guai in crittografia; ritorneremo su questo punto nell'ultimo paragrafo.

L'esistenza di casi facili, d'altra parte, può anche essere volta dal crittografo a proprio vantaggio, come vedremo nel paragrafo seguente. Descriveremo ora tutta una classe di problemi del fusto semplici da risolvere: sono quelli in cui la ennupla $a_1, a_2, a_3, \dots, a_n$ è *supercrecente*, ciò significa che ciascun a_i è superiore alla somma di tutti quelli che lo precedono.

Per esempio una 6-upla supercrecente è $a_1=3, a_2=5(>3), a_3=11(>3+5), a_4=28(>3+5+11), a_5=50(>3+5+11+28), a_6=100$; supponiamo che sia $c=136$: il fatto che c sia maggiore di 100 implica che a_6 debba essere uno degli addendi sommati (i primi cinque a_i , tutti assieme, sommano a meno di 100 perché la 6-upla è supercrecente); dunque m_6 , l'ultima componente di $M=(m_1, m_2, m_3, m_4, m_5, m_6)$, è uguale a 1. Consideriamo allora il valore $c - a_6=36$, che è la somma di alcuni dei numeri a_1, a_2, a_3, a_4, a_5 ; abbiamo dunque ridotto le "dimensioni" del problema del fusto che stiamo risolvendo : siamo passati da 6 possibili addendi a 5 possibili addendi.

Poiché $c - a_6 < a_5$, si vede subito che a_5 non può comparire nella somma, per cui $m_5=0$, $c - a_6$ è la somma dei numeri a_1, a_2, a_3, a_4 : siamo giunti a quattro possibili addendi ecc. Si trova $c=3+5+28+100$, ossia $M=(1,1,0,1,0,1)$.

Il numero di controlli che occorre effettuare è pari a n : la complessità dell'algoritmo che serve a risolvere i *fusti supercrecenti* è dunque soltanto *lineare* (si veda il programma 7.2).

Programma 7.2 Algoritmo *facile*(c,A)

```

per i:=n scendendo a 1 esegui
  inizia
    se c≥ai allora mi:=1 altrimenti mi:=0
    c:=c-ai*mi
  fine
se c=0 allora facile:=M altrimenti "non ci sono soluzioni"

```

Invitiamo il lettore a una breve divagazione: l'ennupla $a_1=2^0$, $a_2=2^1=2$, $a_3=2^2$, $a_4=2^3$, ..., $a_n=2^{n-1}$ è appunto supercrescente; se $c=AM$, allora $(m_n, m_{n-1}, \dots, m_2, m_1)$ ossia M scritto alla rovescia, è proprio la rappresentazione binaria del numero c (si richiede che sia $c < 2^n$; la rappresentazione ha comunque n cifre binarie per cui potrebbero esserci degli zero in testa; a essere pignoli, per avere la rappresentazione binaria nella scrittura abituale bisogna ancora togliere le due parentesi e le virgole): dunque il programma può essere adoperato per convertire i numeri interi dalla scrittura decimale a quella binaria.

5 Il cifrario del fusto

Il sistema di cifra che ora illustreremo è dovuto a R.C. Merkle e M.E. Hellman; esso è basato sulla nozione di *funzione unidirezionale a molla segreta*: si tratta di una funzione unidirezionale che tuttavia si può invertire facilmente se si conosce un'informazione segreta (la "molla segreta"), che nei cifrari serve da chiave di decifrazione. Nel nostro caso quest'informazione consentirà di trasformare un fusto difficile in uno facile, supercrescente.

Se si usa il cifrario del fusto, il messaggio in chiaro è di tipo binario e viene cifrato a blocchi di uguale lunghezza; il crittogramma che corrisponde a ciascun blocco è un numero intero (che può venire trascritto in binario, anche se noi non lo faremo).

Descriveremo per prima la procedura con cui viene costruita

la chiave segreta di decifrazione. Si sceglie un intero n , una ennupla supercrescente a_1, a_2, \dots, a_n e un intero $u > 2a_n$ (il numero n , che è la lunghezza dei blocchi in cui viene spezzato il messaggio è il solo di questi valori che viene reso pubblico). Si sceglie ancora un intero v minore di u e primo con u (v e u non devono avere divisori comuni tranne l'1). Si calcola infine l'“inverso” di v nell'aritmetica circolare modulo u : tale inverso, per definizione, è l'unico numero intero w compreso fra 1 e $u - 1$ per il quale si abbia $vw=1$ modulo u ; la sua esistenza e la sua unicità sono assicurate dal fatto che u e v siano primi fra loro, a norma di un teorema di teoria dei numeri. Diciamo subito che il calcolo di w non è complesso, ma per non perdere il filo del discorso rimandiamo i dettagli al paragrafo successivo.

La chiave di decifrazione (l'informazione che costituisce la “molla segreta” del cifrario) è data dai valori $a_1, a_2, \dots, a_n, u, w$. La sequenza supercrescente $A = (a_1, a_2, \dots, a_n)$ viene infine trasformata nella sequenza “difficile” $D = (d_1, d_2, \dots, d_n)$ in base alla formula

$$d_i = v a_i \text{ modulo } u \quad (i = 1, 2, \dots, n)$$

che più compattamente si può scrivere

$$D = vA \text{ modulo } u. \quad (7.1)$$

Veniamo ora alla descrizione dell'operazione di cifratura; sia $M = (m_1, m_2, \dots, m_n)$ la porzione di messaggio in chiaro che si vuole cifrare; M è un'ennupla binaria. La *chiave pubblica di cifratura* è costituita dall'ennupla numerica d_1, d_2, \dots, d_n ; il crittogramma è il numero c definito dal prodotto scalare

$$c = DM \quad (7.2)$$

Per calcolare M a partire da c , il crittogramma intercettato, e da D , che è la chiave pubblica, una spia dovrebbe risolvere il problema del fusto, e ciò non è possibile a meno che la lunghezza di blocco n non sia molto bassa (in realtà questo è un

punto molto delicato: se v non è scelto con cautela anche il fusto D potrebbe essere facile; si pensi al caso limite in cui v , e dunque w , fossero entrambi uguali a 1: D coinciderebbe con A e il cifrario non servirebbe a niente).

Grazie alla conoscenza della "molla segreta", il ricevitore legittimo riesce a trasformare il fusto difficile $c=DM$ in un fusto supercrescente. Il procedimento è spiegato nella seguente catena di uguaglianze modulo u ; il numero c^* da cui parte è definito nella prima di queste relazioni:

$$\begin{aligned} c^* &= wc \text{ modulo } u \\ &= w(DM) \text{ modulo } u \\ &= w[(vA)M] \text{ modulo } u \\ &= (wv)(AM) \text{ modulo } u \\ &= Am \text{ modulo } u \end{aligned}$$

Abbiamo usato le due formule (7.1) e (7.2), l'associatività dei prodotti modulo u e il fatto che $wv=1$ modulo u (w è l'inverso di v modulo u). Dunque $c^*=AM$ modulo u ; d'altra parte u è maggiore di $2a_n$ e quindi, per la supercrescenza dell'ennupla A , è anche maggiore di $a_1 + a_2 + \dots + a_n$, e a maggior ragione di AM che è la somma soltanto di alcuni degli a_i : poiché AM è minore di u l'indicazione modulo u è superflua e si può scrivere direttamente

$$c^* = AM$$

Il decifratore legittimo conosce A (che fa parte della chiave di decifrazione), può calcolare $c^* = wc$ modulo u (anche w e u fanno parte della chiave di decifrazione; c è il crittogramma trasmessogli); per la supercrescenza di A altrettanto facilmente calcolabile è M , che è il blocco del messaggio in chiaro (si veda la figura 7.2).

Diamo un esempio: siamo stati costretti a usare valori bassi, per cui anche il fusto cosiddetto "difficile" in realtà sarebbe risolvibile. Sia $A = (1, 3, 5, 10)$, $u = 20$ e $v = 7$.

Poiché $7 \times 3 = 1$ modulo 20, $w=3$; la chiave di decifrazione è

COSTRUZIONE DELLE CHIAVI:

Scegli l'ennupla supercrescente A;
 scegli $u > 2a_n$; scegli v primo con u;
 calcola w in base alla formula $wv = 1 \text{ mod } u$;
 calcola l'ennupla D in base alla formula $D = vA \text{ mod } u$

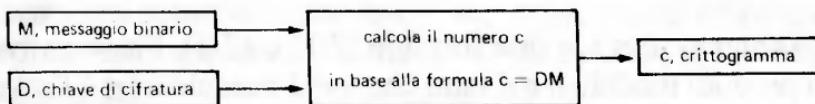
CHIAVE PUBBLICA DI CIFRATURA:

D

CHIAVE SEGRETA DI DECIFRAZIONE:

A, u, w

CIFRATURA:



DECIFRAZIONE:

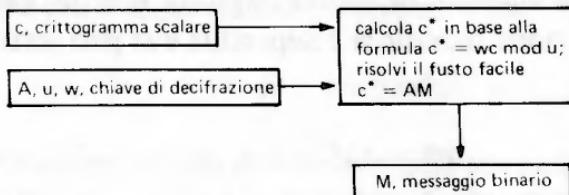


Fig. 7.2 Il cifrario del fusto

$A=(1,3,5,10)$, $u=20$, $w=3$. La chiave pubblica di cifratura è $D=(7,1,15,10)$: infatti $7=7 \times 1$ modulo 20, $1=7 \times 3$ modulo 20, $15=7 \times 5$ modulo 20 e $10=7 \times 10$ modulo 20. Se il blocco da cifrare è $M=(1,1,0,1)$, il crittogramma risultante è $c = DM = 7 \times 1 + 1 \times 1 + 15 \times 0 + 1 \times 10 = 18$. Passiamo alla decifrazione: $c^* = 3 \times c$ modulo 20, ossia $c^* = 14$. Bisogna risolvere il fusto

$14=AM=(1,3,5,10)$ (m_1, m_2, m_3, m_4); l'algoritmo del fusto supercrescente dà subito $M=(1,1,0,1)$.

6 Il teorema di Fermat-Eulerio

A saldo del debito contratto nel paragrafo precedente (ma queste nozioni ci serviranno anche in seguito) mostriamo come si risolve l'equazione $wv=1$ modulo u nell'incognita w , ossia calcoliamo l'inverso di v nell'aritmetica circolare modulo u (u e v sono primi fra loro).

Cominceremo con l'introdurre *la funzione di Eulero* $\phi(n)$, ($n=1,2,3,\dots$), che al numero intero n associa il numero dei numeri interi fra 1 e n che sono primi con n : ad esempio $\phi(3)=2$ perché 1 e 2 sono i primi con 3, $\phi(4)=2$ perché 1 e 3 sono primi con 4, $\phi(5)=4$, $\phi(6)=2$ perché solo 1 e 5 sono primi con 6 ecc. (si conviene che 1 sia primo con ogni numero incluso se stesso per cui $\phi(1)=1$). Se p è un numero primo, *tutti* i numeri che lo precedono sono primi con p , e dunque $\phi(p)=p-1$; più in generale $\phi(n)$ si può calcolare in base alla formula

$$\phi(n) = n \left(1 - \frac{1}{n_1}\right) \left(1 - \frac{1}{n_2}\right) \cdots \left(1 - \frac{1}{n_m}\right)$$

dove n_1, n_2, \dots, n_m sono i fattori primi *distinti* che compaiono nella decomposizione di n . Ad esempio

$$\phi(6) = 6 \left(1 - \frac{1}{2}\right) \left(1 - \frac{1}{3}\right) = 2 \text{ perché } 6=2 \times 3,$$

$$\phi(63) = 63 \left(1 - \frac{1}{3}\right) \left(1 - \frac{1}{7}\right) = 36 \text{ perché } 63=3^2 \times 7,$$

$$\phi(99991) = 99990 \text{ perché } 99991 \text{ è primo ecc.}$$

Ebbene, uno dei più importanti teoremi di teoria dei numeri, il teorema di Fermat-Eulero, afferma che, se $\text{MCD}(v,u)=1$

$$v^{\phi(u)} = 1 \text{ modulo } u$$

ma allora

$$w = v^{\phi(u)-1} \text{ modulo } u$$

(Fermat risolse il caso in cui u è un numero primo, la forma generale è dovuta a Eulero). Ne consegue che il calcolo di un inverso, una volta che si sia riusciti a calcolare $\phi(u)$, si riduce a effettuare un elevamento a potenza nell'aritmetica circolare modulo u (programma 7.1). Sia ad esempio $u=63$ e $v=11$ (u e v sono sicuramente primi fra loro perché 11 è un numero primo) allora $w = 11^{\phi(u)-1}$ modulo u , ossia $w = 11^{35}$ modulo 63; si trova, senza dover passare attraverso il calcolo esplicito di 11^{35} , $w = 23$. Verifichiamo il risultato: $vw = 23 \times 11 = 253 = 4 \times 63 + 1 = 1$ modulo 63. In quanto si è detto c'è un intoppo: il calcolo di $\phi(u)$ non è affatto agevole; ritorneremo su questo punto nel paragrafo successivo. Fortunatamente esiste un algoritmo alternativo che consente il calcolo degli inversi *senza* dover passare attraverso il calcolo della funzione di Eulero. Esso è una "variazione" di un algoritmo assai famoso, l'algoritmo di Euclide per il calcolo del massimo comun divisore di due numeri interi. Li presentiamo entrambi; per evitare troppe divagazioni omettiamo di giustificarne la correttezza e rimandiamo il lettore ai testi indicati nella Bibliografia.

Programma 7.3 L'algoritmo di Euclide. Algoritmo *mcd(a,n)*

inizializza

$g_0 := n$; $g_1 := a$; $i := 1$

finalché $g_i \neq 0$ **esegui**

inizializza

$g_{i+1} := g_{i-1} \bmod g_i$; $i := i + 1$

fine

$mcd := g_{i-1}$

fine

Programma 7.4 L'algoritmo di Euclide adattato al calcolo degli inversi. Algoritmo *inv(a,n)*

inizia

g₀:=n; g₁:=a; u₀:=1; v₀:=0; u₁:=0; v₁:=1; i:=1

finché g_i≠0 **esegui** "g_i=u_in + v_ia"

inizia

y:=g_{i-1} div g_i; g_{i+1}:=g_{i-1} - y*g_i;

u_{i+1}:=u_{i-1} - y*u_i; v_{i+1}:=v_{i-1} - y*v_i;

i:=i+1

fine

x:=v_{i-1}

se x≥0 **allora** inv:=x **altrimenti** inv:=x+n

fine

Visto che abbiamo introdotto la funzione di Eulero, osserveremo di passaggio che il numero dei polinomi binari primitivi di grado n è $\phi(2^n - 1)/n$: il lettore potrà confrontare questo risultato con quanto si è detto sui primi di Mersenne nel paragrafo 2 del capitolo 6.

7 Generazione di numeri primi elevati

Il cifrario a chiave pubblica nel quale al momento presente si ripongono le maggiori speranze è il cifrario RSA, così chiamato dal nome dei suoi inventori, R. Rivest, A. Shamir e L. Adleman; la funzione unidirezionale che sta alla base viene costruita sfruttando il fatto che è facile calcolare il prodotto di due numeri primi "molto grandi", ma dato solo il prodotto è oltremodo difficile risalire ai due fattori (più in generale è assai difficile passare da un numero composto alla sua decomposizione in fattori primi).

Altre due operazioni facili, o relativamente facili, che compaiono nell'algoritmo del cifrario RSA sono le seguenti: dati due numeri interi calcolare il loro massimo comun divisore, o MCD (in particolare controllare se i due numeri sono primi fra

loro, ossia hanno l'MCD pari a 1), generare numeri primi elevati. Per il calcolo dell'MCD non ci sono problemi di sorta: si ricorre all'algoritmo di Euclide.

Il secondo punto è invece molto più delicato. L'algoritmo di generazione che illustreremo è dovuto a Solovay e Strassen e risale al 1975: giustificare la correttezza è al di là dei limiti di questo libro, per cui dobbiamo pregare il lettore di crederci sulla parola (a dire il vero non abbiamo dato spiegazioni neppure nel caso dell'algoritmo di Euclide e della sua variazione, ma ciò era dovuto solo a ragioni di spazio).

Cominciamo a definire il *simbolo di Jacobi*, $J(a,b)$, dove a e b sono due interi primi fra loro, a è minore di b , b è dispari. La definizione che proponiamo, alquanto insolita ma comoda nel nostro contesto, è di tipo “ricorsivo” (algoritmico):

$$1) J(1,B) = 1$$

$$2) \text{ se } a \text{ è pari } J(a,b) = J\left(\frac{a}{2}, b\right) \times (-1)^{(b^2-1)/8}$$

$$3) \text{ se } a \text{ è dispari } J(a,b) = J(b \text{ modulo } a, a) \times (-1)^{(a-1)(b-1)/4}$$

La 1), la 2) e la 3) consentono di calcolare il simbolo di Jacobi, che vale +1 o -1; inutile aggiungere che la definizione che abbiamo dato appare arbitraria e irragionevole solo perché l'abbiamo data “a freddo”, togliendola dal suo contesto naturale che è la teoria dei numeri: ad esso accenneremo in appendice al paragrafo. Calcoliamo ad esempio $J(6,13)$. Si ha, per la 2)

$$J(6,13) = J(3,13) \times (-1)^{21} = -J(3,13)$$

e per la 3):

$$-J(3,13) = -J(13 \text{ modulo } 3, 3) \times (-1)^6 = -J(1,3) = -1$$

nel penultimo passaggio 1 è il resto della divisione 13:3; nell'ultimo passaggio si è usata la 1). Si può dimostrare che se b è un numero primo le uguaglianze

$$\text{MCD}(a, b) = 1 \text{ e } J(a, b) = a^{(b-1)/2} \text{ modulo } b$$

valgono per *tutti* i numeri a compresi fra 1 e $b-1$; se b non è primo si dimostra invece che esse cadono per *almeno la metà* degli a compresi fra 1 e $b-1$ (dicendo che le uguaglianze cadono intendiamo dire che o è già falsa la prima, e allora ovviamente b non è primo, o comunque è falsa la seconda).

Ciò implica che se noi scegliamo a *caso* un numero a compreso fra 1 e $b-1$ la probabilità che quelle uguaglianze cadano è rigorosamente nulla se b è primo ed è almeno $1/2$ se b non lo è. Invece di scegliere un unico numero a compreso fra 1 e $b-1$ ne possiamo scegliere, diciamo, un centinaio (non si dimentichi che ciò che stiamo facendo ha senso solo se b è un numero “enorme”). Se le uguaglianze tornano per tutte le cento scelte del numero a la probabilità che b sia primo è nell’ordine di $(1-2^{-100})$ (o più grande), il che significa che è “praticamente certo” che b sia primo; se cadono anche una sola volta b è sicuramente composto (va da sé che non si è in grado di dire *quale* sia la decomposizione di b). Grazie a questo algoritmo un calcolatore ad alta velocità può verificare in pochi secondi l’eventuale primalità di un numero b preassegnato dell’ordine di grandezza di un centinaio di cifre decimali; se la verifica ha esito negativo si può ricorrere a controlli successivi finché non si trova un numero primo, cosa che avviene in breve tempo. Infatti, e fortunatamente, i numeri primi sono abbastanza frequenti. Un celebre teorema della teoria dei numeri dovuto al russo Pafnutii Ljvovic Cebisev (1821-1894) asserisce che $\pi(n)$, ossia il numero dei numeri primi che precedono n , tende a stabilizzarsi sul valore $\ln n/n$ (il simbolo \ln indica stavolta il logaritmo naturale, o neperiano). Più precisamente, si dimostra che il rapporto fra $n\pi(n)$ e $\ln(n)$ tende a 1 al crescere di n . I primi valori di $\pi(n)$ si possono calcolare a mano: ad esempio $\pi(10)=4$, perché i numeri primi che precedono il 10, e cioè 2,3,5,7, sono in tutto quattro. Per n uguale a un miliardo (e siamo ben al di sotto dei valori che interessano la crittografia, visto che 1.000.000.000 si scrive con appena dieci cifre), il valore esatto di $\pi(n)$ è stato calcolato ed è 50.847.478, mentre la stima $n/\ln n$

restituisce 48.254.942: un errore di appena il 6%. Valutazioni probabilistiche effettuate usando la stima di $\pi(n)$ mostrano che, in media (statisticamente), per stanare un primo di un centinaio di cifre decimali bisogna verificare la primalità di poco più di un centinaio di candidati dispari consecutivi.

L'algoritmo che abbiamo illustrato, come il lettore avrà notato, è di tipo probabilistico: non è infatti garantito che il suo verdetto sia veridico. Questa tuttavia, non preoccupa perché è in nostro potere rendere la probabilità d'errore del tutto irrilevante. Del resto, come vedremo nell'ultimo paragrafo, gli algoritmi probabilistici in crittografia sono molto importanti.

Appendice

A vantaggio dei lettori più curiosi torniamo ora al simbolo di Jacobi e alla sua genesi storica (Karl Gustav Jacobi è vissuto nella prima metà del secolo scorso).

Nelle aritmetiche modulari è importante saper risolvere le equazioni di secondo grado del tipo

$$bx^2 + cx + d = a \text{ modulo } m$$

dove a, b, c, d, m sono assegnati e x è l'incognita (attenzione, siamo in una struttura finita e, in linea di principio, possiamo sempre risolvere quell'equazione "per tentativi", provando per la x tutti i valori possibili da 0 a $m-1$: tuttavia sarebbe mortificante accontentarsi di così poco, e rinunciare a cercare comode formule risolutive!). Orbene, si è dimostrato che il problema generale della soluzione di equazioni di secondo grado nelle aritmetiche modulari si può ricondurre alla soluzione di equazioni di un tipo molto speciale, e cioè:

$$x^2 = a \text{ modulo } p$$

dove il modulo p è un numero primo dispari (ossia diverso da 2, che è l'unico primo pari), e il coefficiente a non è congruo a zero modulo p (non è un multiplo di p ; naturalmente si può sempre ridurre a modulo p , e quindi supporre che a sia compreso fra 1 e $p-1$). È nell'equazione di tipo speciale che stanno tutte le difficoltà serie: difficoltà non da poco, visto che al loro

superamento è dedicato uno dei più celebri teoremi della storia della matematica, il teorema *aureum* del “principe delle matematiche” Carl Friedrich Gauss (1777-1855). Se l’equazione $x^2 = a$ modulo p ammette soluzioni si dice che a è un *residuo quadratico* modulo p . Si è dimostrato che esattamente la metà dei resti che vanno da 1 a $p-1$ sono dei residui quadratici (sono dei quadrati nell’aritmetica modulo p). Al nome di Adrien-Marie Legendre (1752-1834), che portò importanti contributi alla formulazione della “legge della reciprocità quadratica” (al *theoremum aureum*, appunto), è legato il simbolo di Legendre $L(a,p)$ che per definizione, vale 1 se a è un residuo quadratico modulo p , vale -1 altrimenti. Saper calcolare il valore del simbolo di Legendre equivale dunque a scoprire se a sia o non sia un residuo quadratico modulo p : è proprio in questo calcolo che interviene il simbolo di Jacobi, definito nei manuali di teoria dei numeri al modo seguente:

$$J(a,m) = L(a,p_1) L(a,p_2) \dots L(p_s)$$

dove m è un numero intero dispari eventualmente composto, a non ha fattori primi a comune con m (il massimo comun divisore fra a e m è 1), e p_1, p_2, \dots, p_s sono i fattori primi, ripetuti con la loro molteplicità, che compaiono nella fattorizzazione di m ; ad esempio $J(7,45)=J(7,3\times 3\times 5)=L(7,3) L(7,3) L(7,5)=L(1,3) L(1,3) L(2,5)=(+1) (+1) (-1)=-1$ (il 7 è congruo ad 1 modulo 3 e a 2 modulo 5; ovviamente l’1 è un residuo quadratico qualunque sia il modulo; si vede per tentativi che i due residui quadratici modulo 5 sono 1 e 4: $1^2=1$, $2^2=4$, $3^2=4$, $4^2=1$ modulo 5). Invece $J(6,15)$ non è definito perché MCD(6,15)=3 e dunque 6 e 15 non sono primi fra loro (del resto la definizione darebbe $J(6,15)=L(6,3) L(6,5)=L(0,3) L(1,5)$ e farebbe intervenire il “simbolo di Legendre” $L(0,3)$ che non è ammesso). Si noti che se m è un numero primo il simbolo di Jacobi e il simbolo di Legendre sono la stessa e identica cosa.

Ma allora la definizione di simbolo di Jacobi che abbiamo dato nel corpo del paragrafo si rivela per quel che essa davvero è: un comodo algoritmo per il calcolo dell’importante sim-

bolo di Legendre! Ad esempio l'equazione

$$x^2=21 \text{ modulo } 23$$

non ammette soluzioni perché $L(21,23) = -1$. Infatti $L(21,23) = J(21,23) = J(23 \text{ modulo } 21, 21) (-1)^{110} = J(2,21) = -J(1,21) \times (-1)^{55} = J(1,21) = -1$. (Si noti che, pur partendo da un simbolo di Legendre, 21 non essendo primo: ben fece dunque Jacobi ad estendere la definizione dei simboli di Legendre; sottolineiamo tuttavia che il simbolo di Jacobi non ha una interpretazione diretta in termini della risolubilità di equazioni: in altre parole, $J(a,m) = 1$ con m composto non implica affatto che l'equazione $x^2 = a \text{ modulo } m$ abbia soluzioni).

8 Il cifrario RSA

La chiave segreta di decifrazione del cifrario RSA consiste essenzialmente, anche se non letteralmente, di due numeri primi molto elevati, p e q ; il loro prodotto $n=pq$ viene calcolato e reso pubblico: esso costituisce parte della chiave di cifratura. Entriamo in maggiori dettagli. Abbiamo già definito nel paragrafo 6 la funzione di Eulerio $\phi(n)$; si vede facilmente, anche senza ricorrere alla formula, che nel nostro caso $\phi(n) = (p-1)(q-1)$. L'aritmetica modulo n e quella modulo $\phi(n)$ intervengono entrambe in quanto stiamo per dire. La prima è usata nelle operazioni di cifratura e di decifrazione, la seconda nella costruzione delle chiavi: oltre a p e a q il decifratore deve scegliere un intero x primo con $\phi(n)$ e calcolarne l'inverso y modulo $\phi(n)$ (ci si può servire degli algoritmi descritti nel paragrafo 6; l'algoritmo di Euclide può essere usato per controllare che x sia primo con $\phi(n)$). Anche y , assieme a n , reso pubblico; x invece è tenuto segreto.

Il messaggio da cifrare è costituito da una sequenza di numeri interi ciascuno compreso fra 0 e $n-1$. Se l'alfabeto dei messaggi fosse diverso si potrebbe sempre ricorrere a una codifica preventiva: per ragioni di formato potrebbe essere conveniente scrivere quegli interi (le "lettere" dell'alfabeto dei messaggi)

come blocchi binari o decimali di lunghezza fissa, aggiungendo dove occorre degli zero in testa.

Per esempio, se $n=pq$ fosse uguale a $391=17 \times 23$, gli interi da 0 a 390 potrebbero venire scritti in decimale come 000, 001, 002, ..., 389, 390. Tutto ciò, lo ribadiamo, è un problema di codifica e non di cifratura: non c'è nulla di segreto.

Fissiamo l'attenzione su uno dei numeri che costituiscono il messaggio, diciamo il primo, m (m è compreso fra 0 e $n-1$).

Il crittogramma corrispondente a m è allora $c=m^y$ modulo n ; esso può venir calcolato a partire da m e dalla chiave pubblica dicifratura (n, y) . (Si noti che l'alfabeto dei messaggi e quello dei crittogrammi coincidono: testo in chiaro e testo in cifra hanno lo stesso "formato".)

La chiave segreta di decifrazione è costituita dall'intero x : il messaggio infatti, come si può dimostrare, è calcolabile nella forma $m=c^x$ modulo n ; questo calcolo si può eseguire se si conoscono il crittogramma e la chiave di decifrazione x (bisogna conoscere anche n , ma ciò è scontato perché si tratta di un'informazione pubblica). Tutto il procedimento è riassunto e schematizzato in figura 7.3.

Sono necessari alcuni commenti. Innanzi tutto non è affatto ovvio che sia $c^x=m$ modulo n : vediamo di dimostrarlo nel caso che né p né q dividano m e dunque che m sia primo con n , lasciando al lettore matematico la cura di trattare gli altri casi possibili, che d'altronde sono simili.

Grazie al teorema di Fermat-Eulero si ha $m^{\varphi(n)}=1$ modulo n . D'altra parte, per come è stato costruito y , $xy=1$ modulo $\varphi(n)$, ossia $xy=Q\varphi(n)+1$, dove Q è il quoziente nella divisione di xy per $\varphi(n)$ (1 è il resto). Dunque

$$m^{xy} = m^{Q\varphi(n)+1} = [m^{\varphi(n)}]^Q m = [1]^Q m \text{ modulo } n,$$

ossia $m^{xy}=m$ modulo n , o ancora, tenendo presente che il crittogramma è $c=m^y$:

$$m^{xy} = (m^y)^x = c^x = m \text{ modulo } n,$$

CRITTOGRAFIA

COSTRUZIONE DELLE CHIAVI:

Scegli l'ennupla supercrescente A;
 scegli $u > 2a_n$; scegli v primo con u;
 calcola w in base alla formula $wv = 1 \bmod u$;
 calcola l'ennupla D in base alla formula $D = va \bmod u$

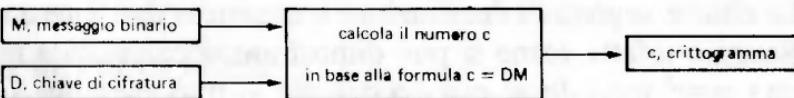
CHIAVE PUBBLICA DI CIFRATURA:

D

CHIAVE SEGRETA DI DECIFRAZIONE:

A, u, w

CIFRATURA:



DECIFRAZIONE:

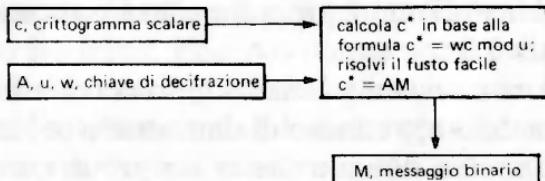


Fig. 7.3 Il cifrario RSA.

che è quanto si voleva dimostrare.

E se m dividesse uno dei due fattori primi, diciamo q ? In questo caso la dimostrazione precedente mostra che si ha $m^{\varphi} = m$ modulo p (basterà applicare il teorema di Eulero con p invece di n e ricordare che $xy = Q\varphi(n) + 1 = Q(p-1)(q-1) + 1 = Q\varphi(p)\varphi(q) + 1$). D'altronde, se m divide q , si dimostra subito che è $m^{\varphi} = m$ modulo q , e non c'è neppure bisogno di scomporre Eulero: m^{φ} e m sono entrambi multipli di q , e dunque sono entrambi congrui a 0 modulo q . Ricordando la definizione di congruenza modulo p e modulo q , abbiamo allora scoper-

to che la differenza $m^y - m$ è un multiplo sia di p sia di q : nella decomposizione in fattori primi di $m^y - m$, decomposizione che è unica, intervengono dunque sia il fattore primo p sia il fattore primo q ; ma allora $m^y - m$ è un multiplo di $n = pq$, ossia $m^y = m$ modulo n , in tutta generalità.

Vediamo di capire perché il cifrario RSA sia considerato sicuro. Se si disponesse di un algoritmo efficiente per fattorizzare n e risalire a p e a q , la spia potrebbe calcolare $\varphi(n) = (p-1)(q-1)$ e quindi x in base alla formula $xq \equiv 1 \pmod{\varphi(n)}$ (si ricordi che y è noto): il cifrario sarebbe forzato. D'altronde, al momento, non sono noti algoritmi efficienti per fattorizzare un numero composto, anzi, si è “pressoché” sicuri che tali algoritmi *non esistano affatto* (sul “pressoché” torneremo nell'ultimo paragrafo). Il lettore potrebbe tuttavia accampare altre obiezioni: sorge per esempio il dubbio che ci sia il modo di calcolare $\varphi(n)$ senza passare attraverso p e q . In realtà il problema di fattorizzare $n = pq$ o di calcolare $\varphi(n)$ a partire da n hanno grosso modo lo stesso grado di complessità (sono “essenzialmente” lo stesso problema), come traspare dalle seguenti uguaglianze che consentono di calcolare p e q a partire da n e da $\varphi(n)$ (ciò prova che qualunque algoritmo efficiente per calcolare $\varphi(n)$ può venir convertito in un algoritmo efficiente per fattorizzare n):

$$\begin{aligned}\varphi(n) &= (p-1)(q-1) = n - (p+q) + 1 \\ (p-q)^2 &= (p+q)^2 - 4n\end{aligned}$$

(dalla prima si calcola la somma $(p+q)$, dalla seconda la differenza $(p-q)$; somma e differenza danno subito p e q).

Può comunque sussistere il dubbio che sia possibile forzare il cifrario RSA senza disporre di un algoritmo efficiente per fattorizzare n ; il dubbio è fondato: tutto ciò che si può dire è che nessuno ci è finora riuscito.

Concludiamo con un esempio numerico “in miniatura”. Sia $p=5$, $q=11$; dunque $n=55$, $\varphi(n)=40$. Le “lettere” in cui va preliminarmente trascritto il messaggio in chiaro sotto i numeri $0, 1, 2, \dots, 53, 54$. Il decifratore deve scegliere un intero x primo con 40 , per esempio 23 che è esso stesso primo. L'inverso di $x=23$

modulo 40 è $y=7(7 \times 23=161=4 \times 40 + 1=1$ modulo 40). La chiave pubblica di cifratura è allora la coppia (55,7), la chiave segreta di decifrazione è $x=23$. Supponiamo che il messaggio, o meglio la sua prima "lettera", sia $m=4$. Allora $c=4^7$ modulo 55=49; il lettore potrà controllare che $c^x=49^{23}=4$ modulo 55 ricorrendo in entrambi i casi all'algoritmo che consente di effettuare l'elevamento a potenza nelle aritmetiche circolari (paragrafo 2).

9 Autenticazione e firme numeriche

I sistemi di trasferimento elettronico di denaro sono ormai un elemento essenziale della vita moderna e sono destinati ad acquistare sempre più spazio nel futuro: purtroppo, al di là dei loro ovvi vantaggi che non sta a noi sottolineare, tali sistemi rischiano di offrire occasioni d'oro alla criminalità informatica. Stavolta non è la segretezza dei messaggi che va protetta, per quanto anche questo aspetto possa essere presente: il rischio è piuttosto che un intruso, facendo le veci dell'utente legittimo ("interpretandolo"), si impossessi del suo denaro, per esempio lo storni sul proprio conto corrente. Come si vede, nella società telematica, oltre che dagli intercettatori *passivi*, quelli che si limitano a "origliare", la crittografia deve proteggerci anche dagli intercettatori *attivi*, che potrebbero modificare o distruggere un messaggio legittimo, oppure inserire fraudolentemente nel sistema di comunicazione un messaggio non autorizzato che torni a loro vantaggio.

Date queste premesse, si capisce quanto sia importante che due utenti remoti l'uno dall'altro, che non possono comunicare a viva voce anche perché uno di loro talvolta è il calcolatore centrale di una complessa rete telematica, possano riconoscersi tramite procedure automatiche in modo da avere piena fiducia l'uno nell'altro. I crittografi devono dunque misurarsi con il problema dell'*autenticazione* degli utenti di un sistema di comunicazione.

La crittografia a chiave pubblica sembra offrire soluzioni estremamente promettenti anche per questo tipo di problemi. In

particolare è possibile escogitare un sostituto valido di quell'antichissimo sistema di autenticazione che è la firma scritta. Lo scopo di una firma è quello di evitare che il ricevente possa attribuire al mittente un messaggio autentico; in caso di contestazioni un giudice imparziale è in grado di comporre la lite. Vediamo ora come una normale firma scritta possa essere sostituita da una *firma numerica*, o *firma digitale*, opportunamente cifrata.

Supponiamo di avere a disposizione un sistema crittografico a chiave pubblica, basato su una funzione unidirezionale: ciascun utente X genera due chiavi, la C_x che viene resa pubblica, e la sua inversa C_x^{-1} che viene custodita segreta. Se l'obiettivo fosse la segretezza e non l'autenticazione, la C_x servirebbe per la cifratura dei messaggi riservati e la C_x^{-1} per la decifrazione dei crittogrammi. Supponiamo invece che X voglia spedire a Y un messaggio non riservato; X e Y possono servirsi della procedura descritta nello schemino che segue (in esso M è il messaggio, F è la firma numerica; la lettera sopra le frecce spiega qual è l'utente che compie l'operazione indicata; gli asterischi segnalano la trasmissione lungo il canale):

$$MF \xrightarrow{X} M C_x^{-1}(F) \xrightarrow{* * *} M C_x^{-1}(F) \xrightarrow{Y} M C_x[C_x^{-1}(F)] = MF$$

Ciò consente a Y , e a qualunque altro utente, di autenticare X , il solo che può aver calcolato $C_x^{-1}(F)$. Un'ovvia riserva è che la firma in cifra $C_x^{-1}(F)$ potrebbe essere usata da un intercettatore malintenzionato in un secondo tempo: è per questa ragione che la firma numerica, a differenza della firma scritta, *non deve essere riutilizzabile*: essa dovrà dunque contenere degli elementi che dipendono non solo dall'identità del mittente X ma anche dal contenuto del messaggio M e/o dall'istante di tempo in cui il messaggio viene trasmesso.

Si noti che il problema che stiamo discutendo ora è per così dire l'immagine speculare di quello della segretezza: chiunque può ricostruire la firma in chiaro F a partire dalla firma in cifra $C_x^{-1}(F)$, ma solo X è in grado di produrre il "crittogramma" $C_x^{-1}(F)$. Non meraviglia dunque che le chiavi vengano adopera-

te in ordine inverso rispetto a quanto succede nel caso dei cifrari: la chiave segreta serve per cifrare e quella pubblica per decifrare. La procedura descritta permette a Y di autenticare X : se si vuole che solo Y sia in grado di farlo bisogna adoperare, stavolta nel loro ordine solito (la chiave pubblica per prima e la chiave segreta per seconda) le due chiavi di Y , la C_Y e la C_Y^{-1} :

$$MF \xrightarrow{X} M C_X^{-1}(F) \xrightarrow{X} C_Y[M C_X^{-1}(F)]***$$

$$C_Y[M C_X^{-1}(F)] \xrightarrow{Y} C_Y^{-1} \{C_Y[M C_X^{-1}(F)]\} =$$

$$= M C_X^{-1}(F) \xrightarrow{Y} M C_X[C_X^{-1}(F)] = MF$$

Nello schema anche il messaggio è stato cifrato: se esso non fosse riservato la C_Y verrebbe applicata solo alla firma in cifra $C_X^{-1}(F)$ e non all'intero messaggio firmato $M C_X^{-1}(F)$

Aggiungeremo un paio di osservazioni. Sistemi simili a quello che abbiamo descritto possono essere realizzati anche nell'ambito della crittografia a chiave segreta; tuttavia in questo caso il sistema di autenticazione è attendibile solo se si dispone di un metodo assolutamente fidato per la distribuzione delle chiavi: è il punto debole tipico della crittografia tradizionale, e basta pensare all'esempio del trasferimento elettronico di denaro per capire che non si tratta di un difetto da poco. Seconda osservazione: accanto alla firma in cifra $C_X^{-1}(F)$ di norma compare anche la firma in chiaro X , priva di "valore legale", che corrisponde, per così dire, alla firma dattiloscritta che nelle lettere figura al di sotto della firma scritta a mano. È grazie a questa firma in chiaro che l'utente Y sa quale chiave pubblica C_X deve adoperare per dare inizio ai suoi controlli.

10 Osservazioni critiche

Il logaritmo finito, il problema del fusto e la fattorizzazione di un numero composto ci offrono tre tempi di funzioni unidirezionali: alle operazioni dirette che sono semplici dal punto di vista algoritmico (l'elevamento a potenza, il prodotto scalare, il

prodotto ordinario) si contrappongono quelle inverse che diventano rapidamente "intrattabili" (il calcolo del logaritmo finito, la determinazione di un'ennupla binaria incognita che moltiplicata scalarmente per un'ennupla nota dia uno scalare noto, la fattorizzazione di un numero composto). Non bisogna affatto illudersi che ogni funzione unidirezionale sia utile in crittografia: sono stati proposti numerosi sistemi che in apparenza sono perfettamente analoghi a quello del fusto o all'RSA, e che invece si sono rivelati del tutto insicuri.

Oltre a certe difficoltà di attuazione pratica, sensibili nel caso del cifrario RSA, i sistemi a chiave pubblica hanno vari punti deboli che ne rendono problematico l'uso. La situazione si può riassumere in una battuta: la crittografia a chiave pubblica si è largamente basata sulla teoria della complessità, ma la teoria della complessità di cui essa avrebbe davvero bisogno *non* è quella che i teorici hanno via via costruito in questi decenni (tuttavia, grazie a un fruttuoso meccanismo di retroazione, la crittografia sta ormai sollecitando tutta una serie di ricerche sulla complessità degli algoritmi che seguono direzioni poco ortodosse, o comunque relativamente poco coltivate nel passato). Il primo inconveniente è già noto al lettore: nella teoria della complessità, o almeno nella teoria della complessità "standard", la difficoltà di un problema viene valutata in base al caso più difficile; siamo dunque di fronte a un criterio di *complessità massima* e non, come sarebbe più utile per i nostri fini, a un criterio di *complessità media*. Quando si dice che il calcolo del logaritmo finito è troppo complicato si intende semplicemente dire che *esistono dei casi* in cui tale calcolo è troppo complicato: resta pur sempre il timore che esistano altri casi, che potrebbero anche essere molto numerosi, in cui la complessità cali in maniera determinante. Del resto il cifrario del fusto sfrutta proprio l'esistenza di tutta una classe di "fusti facili". Purtroppo di fusti facili e di logaritmi facili ne sono stati scoperti anche là dove non ce ne sarebbero dovuti essere: oggi siamo a conoscenza di intere classi di casi che il crittografo deve guardarsi dall'usare. In realtà il cifrario del fusto, deve ormai ritenersi forzato: per essere più precisi è il meccanismo di trasformazio-

ne del fusto segreto facile (supercrescente) in un fusto difficile che si è rivelato inadeguato. Le speranze di rabberciare i guasti sono state abbandonate: al cifrario del fusto va tuttavia reso l'onore delle armi, perché, a differenza di altri sistemi forzati nel breve lasso del congresso in cui erano stati presentati, la sua crittanalisi è stata lunga, impegnativa, e ha richiesto l'ideazione di tecniche nuove e profonde.

Non basta: quando si dice che un certo problema è difficile, si intende dire che *attualmente* non si conoscono algoritmi efficienti per risolverlo: tuttavia nessuno è riuscito a *provare* che tali algoritmi non esistano. Considerando gli sforzi enormi in cui molti dei più brillanti matematici si sono impegnati nella vana ricerca di algoritmi rapidi, il rischio che qualcuno un giorno finisca con lo scoprirli sembra abbastanza remoto. Poco importa: la coscienza professionale dei crittografi è comunque rosa dal tarlo del dubbio. Non basta ancora: nella teoria della complessità "standard" non si tiene conto della possibilità di algoritmi di tipo *probabilistico*, ossia di algoritmi che risolvono un certo problema nella maggioranza dei casi ("con grande probabilità"), anche se talvolta (ma ciò avviene appunto "con piccola probabilità") essi continuano a girare a vuoto senza produrre nessun risultato, oppure danno un risultato sbagliato. Vittima illustre di un algoritmo probabilistico è rimasto il logaritmo finito, sul campo di Galois di 2^{54} elementi che oggi non può più venire adoperato nella distribuzione delle chiavi DES: con probabilità molto vicina a uno (nella stragrande maggioranza dei casi), l'algoritmo riesce a calcolare quel logaritmo finito in tempi assai brevi e senza occupare uno spazio di memoria eccessivo. È questo uno dei motivi per cui oggi si ritiene che la chiave del DES sia troppo corta; raddoppiarne la lunghezza, cosa che porterebbe a un aumento vertiginoso del numero delle chiavi disponibili, sarebbe sufficiente per metterci al sicuro dalle insidie dell'algoritmo probabilistico cui abbiamo accennato, ma tuttavia chi può garantirci che un giorno o l'altro non venga inventato qualche nuovo algoritmo molto più efficiente?

Nonostante tutte queste riserve, la crittografia a chiave pub-

blica è verosimilmente destinata a diventare la crittografia del futuro, perlomeno per quanto riguarda le applicazioni commerciali: in essa gli elementi di segretezza sono ridotti all'osso, e la segretezza, nella società dei media di massa e della telematica, è un "bene" di costo estremamente elevato. Al momento nella gara fra RSA e DES, l'RSA, ormai entrato nell'uso effettivo dopo le perplessità iniziali, ha il vantaggio della non-segretezza ma il DES è molto più veloce: ai tassi di trasmissione del DES, che sono di milioni di bit al secondo, l'RSA può opporre tassi di qualche migliaio di bit al secondo, un centinaio di migliaia per grossi calcolatori VAX e una ventina di migliaia per piccoli calcolatori personali.

11 Prospettive future

Fra i futuribili vogliamo ricordare rapidamente il *cifrario della luna* dello svizzero Ueli Maurer, presentato nel 1990, e la *crittografia quantistica*, proposta da Charles Bennet e Gilles Brassard nel 1982 (ma preceduta negli anni '60 da un articolo da Stephen Wiesner troppo in anticipo sui tempi per venir apprezzato quando apparve). La crittografia quantistica è entrata nella fase sperimentale con la realizzazione di un prototipo nel 1989. In entrambi i casi, lo scopo, assai ambizioso, è quello di costruire un sistema di cifra da un lato maneggevole (a differenza del *one-time pad*, ossia del cfrario a chiave non riutilizzabile), dall'altro dimostrabilmente sicuro, come il *one-time pad*, e non soltanto *congetturalmente* sicuro, come il DES o l'RSA.

Il cfrario della luna è un cfrario "condizionatamente perfetto": in termini più esplicativi, il cfrario è inattaccabile (lo si può rigorosamente dimostrare) a condizione che non si verifichi un certo evento aleatorio, la cui probabilità, tuttavia, può essere calcolata, ed è così piccola che la catastrofe si può ritenere impossibile a tutti gli effetti pratici. La chiave segreta del cfrario è una sequenza binaria casuale breve e dunque maneggevole. Abbiamo risolto tutti i problemi della crittografia a chiave segreta, allora? L'inconveniente, al solito, c'è: il cfrario della

luna richiede infatti la costruzione preliminare di una tabella di bit casuali (e non pseudocasuali) che può sì venir riutilizzata quante volte si vuole, ma che è sterminata (sufficientemente ampia da impedire una lettura sostanzialmente esauriente da parte dell'avversario): se avete animo poetico, potete pensare di depositare i bit della tabella in modo da tappezzare l'intera superficie della luna, come suggerisce il buffo nome del cifrario: poi, osservando la luna col telescopio, estrarrete opportuni spezzoni di bit e li combinerete fra loro in base alle indicazioni della breve chiave casuale, in modo da formare la lunga sequenza binaria che sommata bit dopo bit, al messaggio in chiaro dà luogo al crittogramma. È poco credibile che ci si debba mai preoccupare dell'inquinamento del nostro satellite provocato dai bit dei crittografi; vanno trovate altre soluzioni tecnologiche, meno suggestive ma più realistiche: si è suggerito ad esempio di sfruttare le sorgenti "naturali" di rumore casuale dello spazio remoto.

Una buona dose di fiducia nel progresso tecnologico è necessaria anche quando si parla della crittografia quantistica (d'altronde l'esperienza che abbiamo sembra contraddirsi i pessimisti come quelli che a suo tempo dichiararono l'RSA un'elucubrazione matematica del tutto inutilizzabile sul piano pratico). Stavolta entra in gioco nientemeno che la fisica quantistica e più esattamente il *principio di indeterminazione* di Heisenberg, del 1927: tale principio afferma che certe proprietà fisiche sono "incompatibili" a coppie nel senso che qualunque misurazione del valore assunto dalla prima rende "indeterminato" (del tutto casuale) il valore assunto dalla seconda. Ad esempio, il fatto di misurare in un fotone la polarizzazione lineare, che può essere verticale o orizzontale, "casualizza" la sua polarizzazione circolare, che può essere oraria o antioraria; succede il viceversa se si misura la polarizzazione circolare. Un bel guaio per l'osservatore, dunque, ma un vantaggio per il crittografo quando al posto dell'osservatore ci sia proprio la spia, che non potrebbe più intercettare ("osservare") il crittogramma senza con ciò stesso deteriorarlo in maniera irrimediabile. Il primo esperimento convincente di trasmissione quantistica ha avuto

luogo nell'ottobre dell'89, e consisteva nello scambio di deboli lampi di luce polarizzata su un percorso di qualche decina di centimetri: siamo ancora lontani, dunque, dall'applicabilità effettiva e su larga scala! Non importa: l'avventura della crittografia non è finita. Il suo destino sembra davvero legato all'esplorazione di *quali siano i limiti, fisici, logici o algoritmici, che separano ciò che possiamo fare da ciò che non possiamo fare*: un interrogativo che, per il suo peso scientifico e filosofico, travalica di molto gli interessi meramente crittografici.

Bibliografia

La letteratura crittografica è ormai vastissima: in queste righe ci limiteremo a qualche indicazione di massima. Com'è facile intuire, il lettore che desideri approfondire le sue conoscenze crittologiche deve sapere l'inglese, o meglio l'americano.

A dire il vero uno dei migliori manuali introduttivi alla crittografia attualmente in commercio è scritto in inglese britannico: *Cipher Systems* di BEKER, H e PIPER, F. Londra: Northwood Books, 1982. Nell'inglese d'oltre atlantico sono invece scritti *Cryptography and Data Security* di DENNING, D.E., Reading: Addison Wesley, 1982 e *Cryptography, a Primer* di KONHEIM, A.G., New York: J. Wiley and Sons, 1981, entrambi ottimi. Il livello matematico di questi volumi è più elevato di quello da noi presupposto, anche se nessuno dei tre è gratuitamente astruso. Al lettore che conosca il tedesco possiamo suggerire *Kriptographie* di BETH, TH., HESS, P. e WIRKL, K., Stoccarda: B.G. Teubner, 1983. chi ha apprezzato il libro di Beker e Piper vorrà leggere anche *Secure Speech Communications*, New York: Academic Press 1985, che tratta di un tema trascurato nel nostro libro, quello della protezione della voce (si pensi alle comunicazioni via radio, per telefono ecc.). Più di recente è uscito *Contemporary Cryptology*, a cura di SIMMONS G.: IEEE Press, 1992 (Simmons è uno dei creatori della teoria dell'autenticazione).

I contributi "di punta" (e stavolta i prerequisiti tecnici sono di norma decisamente elevati) sono disseminati su numerose riviste di matematica, di informatica e di ingegneria delle comunicazioni. Citeremo solo qualche esempio illustre. Su *IEEE Transactions on Information Theory* (IEEE sta per Institute of Electrical and Electronics Engineers) è apparso nel novembre 1976 un articolo di DIFFIE, W. e HELLMAN, M.M. in cui viene introdotta la crittografia a chiave pubblica: "New Direc-

tions in Cryptography ”. Sempre su *Transactions on Information Theory* è stato presentato il cifrario del fusto (MERKLE, R.C. e HELLMAN, M.E. “Hiding Information and Signatures in Trapdoor Knapsacks”, settembre 1978), mentre la descrizione del cifrario RSA è apparsa su *Communications of ACM* (RIVEST, R.L., SHAMIR, A. e ADLEMAN, L. “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”, febbraio 1978; ACM sta per Association for Computing Machinery). Un’altra rivista su cui sono apparsi importanti lavori è il *Bell System Technical Journal*: ne citeremo uno “storico”, ma ancor’oggi leggibilissimo, “Communication Theory of Secrecy Systems” di Claude Shannon (ottobre 1949). Esiste d’altra parte una rivista americana, *Cryptologia*, interamente dedicata alla crittografia: i contributi sono di tipo molto vario e vanno dalla crittografia ricreativa alla storia della crittografia, fino a lavori di tono decisamente tecnico. A chi è interessato alla storia della crittografia è inevitabile suggerire un volume ormai “classico” di dimensioni ponderose: *The Codebreakers* di KAHN, D. New York: Macmillan, 1967; un’edizione ridotta è stata pubblicata dalla Signet nel 1973. Di lettura più leggera, e si spera divertente, è *Codici segreti*: Oscar Mondadori, 1989, scritto dall’autore del libro che state leggendo. Per chi è interessato agli algoritmi è inevitabile il riferimento a due opere ponderose: *The Art of Computer Programming* di KNUTH D.: Addison-Wesley, 1973 (vol. 1 e 3) e 1981 (vol. 2), e *Introduction to Algorithms* di CORMEN TH. H., LEISERSON CH.E. e RIVEST R.L. (proprio quello dell’RSA): MIT Press, 1990; in quest’ultimo viene anche discussa la teoria della complessità. Senz’altro più accessibile è *Algoritmi elementari* di PINTACUDA N., Muzzio, 1986.

Esiste anche una associazione per la ricerca crittologica, la IACR (International Association for Cryptologic Research) che organizza convegni sia negli Stati Uniti (Crypto) sia in Europa (Eurocrypt); gli atti dei lavori vengono poi pubblicati. Osserviamo che un dei primi Eurocrypt (a dire la verità questo nome è entrato ufficialmente in uso più tardi) ha avuto luogo a Udine nel 1983; i partecipanti italiani si contavano purtroppo sulle

dita di una mano. Non sarà certo così all'Eurocrypt del 1994, che si svolgerà a Perugia: la ripresa di interesse per la crittografia è oggi evidente anche nel nostro paese, di tradizioni crittografiche illustri, che però si erano alquanto appannate col tempo. Del resto piaccia o non piaccia, in una società "telematica" delle tecniche crittografiche non si può fare a meno.

Glossario crittografico inglese-italiano

La crittografia è una disciplina “informatica” estremamente antica: ne risulta che la gran parte dei termini crittografici inglesi sono di origine latina o greca, e di conseguenza del tutto analoghi a quelli italiani. Ecco un breve e ovvio elenco: *cryptography*, *cryptanalysis*, *cryptology*, *cipher* (cifrario), *to encipher* (cifrare), *to decipher*, *to cipher* (= *to cipher* + *to decipher*), *key* (chiave), *cryptogram*, *to encrypt* (sinonimo di *to encipher*), *to decrypt*, *plaintext* (testo in chiaro), *ciphertext* (testo in cifra), *substitution cipher*, *transposition cipher*, *grill cipher* (cifrario a griglia). Un cifrario può essere *additive*, *monoalphabetic*, *polyalphabetic*, *polygraphic*, *homophonic*, *perfect*. Un messaggio, o *message*, può essere *confidential* (riservato), o perfino *secret*. Ecco qualche termine più recente: *exhaustive search* (ricerca esauriente), *signal scrambling* (rimescolamento del segnale), *shift register* (registro a scorrimento), *linear feedback*, *random digits*, *pseudorandom digits* (cifre casuali o pseudocasuali), *one-time pad* (cifrario a chiave non riutilizzabile; il nome inglese si riferisce alla seguente immagine: le cifre casuali vengono scritte su tabelle che compongono un blocco di fogli, un *pad*, appunto, da usarsi una volta sola, *one time*), *chosen plaintext attack* (attacco con testo in chiaro), *black cipher* (cifrario a blocco) e *stream cipher* (cifrario a flusso).

I due casi più difficili sono quelli del *knapsack cipher* (cifrario del fusto) e delle *trapdoor one-way functions* (funzioni unidirezionali a molla segreta). Non abbiamo tradotto *knapsack* con zaino o *trapdoor* con botola, come consiglia il dizionario, per le ragioni seguenti: a differenza del *knapsack* americano uno zaino non evoca in noi italiani l’immagine di un oggetto tubolare, come dovrebbe; la *trapdoor* fa venire in mente, oltre alle botole, una di quelle porticine nascoste dietro i libri di una

biblioteca, che passano inosservate a chi non sia a parte del segreto e che si aprono premendo appunto un'opportuna molla segreta (un finto volume, una piastrella mobile ecc). Alle traduzioni letterali abbiamo preferito traduzioni infedeli che però evocassero immagini opportune, visto che questo è non altro è lo scopo per cui i termini inglesi, o meglio americani, sono stati scelti.

Ormai un numero enorme di messaggi viaggia su ogni sorta di canali, dalla posta tradizionale al telefono, alla radio, al telefax e alle linee di trasmissione dei dati ad alta velocità: una quantità di dati in rapidissima crescita viene immagazzinata nelle memorie dei calcolatori e nelle banche di dati.

Spesso le informazioni trasmesse o memorizzate sono preziose e riservate (basti pensare alle cartelle cliniche e al trasferimento elettronico di denaro): è difficile, se non impossibile, proteggerle da orecchi e occhi indiscreti senza ricorrere alle tecniche che la crittografia ci mette a disposizione, oggi grazie anche al progresso dell'informatica e della microelettronica.

Andrea Sgarro in queste pagine prende le mosse dalla crittografia "storica", dove i concetti fondamentali sono messi più facilmente a nudo per affrontare progressivamente la crittografia contemporanea, sia quella a chiave segreta, sia quella a chiave pubblica, sempre mirando a evidenziare le idee di fondo, più che le singole soluzioni tecniche.

Lire 18.000

ISBN 88-7021-640-3



9 788870 216400