CRITTOGRAFIA

Umberto Cerruti

Università di Torino

3 Ottobre 2007, INFN Frascati

L'ATBASH Biblico

א alef	ב bet	ג ghimel	ד dalet	ה he	ו vav	ז zain	ח che	ט tet	י yod	כ caf
ל lamed	מ mem	נ nun	ס samech	ע ain	פ pe	צ tzaddi	ק qof	ר resh	ש shin	ת tau

א ב ג ד ה ו ז ח ט י כ
ת ש ר ק צ פ ע ס נ מ ל

Sostituzione monoalfabetica

L'atbash è un codice di *sostituzione monoalfabetica*, che viene utilizzato alcune volte nella Bibbia, per esempio nel libro di Geremia (25:26) si trova *Sheshakh* invece di *Babel*:

Insomma, feci bere da quella coppa tutti i regni che sono sulla terra e lasciai per ultimo il re di Sheshakh

La Bibbia contiene un testo in codice?

Dal Libro di Daniele (5:25-30)

25 Ecco quel che c'è scritto: MENE, MENE, TEKEL, PERES.

30 In quella stessa notte, Baldassar, re di Babilonia, venne ucciso.

Steganografia

La steganografia è l'arte di nascondere un messaggio all'interno di un altro.

Si manda una lettera che racconta la nostra vacanza a Venezia, ma spazi, virgole e punti raccontano tutta un'altra storia.

I metodi sono infiniti, basta avere sufficiente fantasia.

Oggi è facile nascondere un testo dentro un file immagine (.jpg) o musicale (.mp3), a costo di una impercettibile differenza dall'originale.

Watermarking

Il watermarking invisibile è una tecnica che utilizza la steganografia per nascondere (tipicamente in immagini o file audio - video) informazioni che riguardano la provenienza dell'opera e i diritti di copyrigth.

Sostituzione Monoalfabetica

Chiamiamo Alfabeto l'insieme dei simboli che intendiamo utilizzare.
Qui sotto si trova la nostra versione dell'alfabeto, costituito da 73 simboli:

.:;?!0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyzòàùèé

Ad ogni simbolo si associa un numero, nel nostro caso da 0 a 72.
0 è il punto e 72 è lo spazio.
Uno dei primi codici a sostituzione monoalfabetica, detto di Giulio Cesare, spostava semplicemente di tre passi l'alfabeto, in modo ciclico.

Interpretazione Algebrica

Algebricamente questo corrisponde a sommare 3 ai numeri corrispondenti alle cifre, e ridurre modulo 73.

Esempio di Sostituzione Monoalfabetica

In questo modo l'incipit della Divina Commedia:

Nel mezzo del cammin di nostra vita mi ritrovai per una selva oscura, ché la diritta via era smarrita. Ahi quanto a dir qual era è cosa dura esta selva selvaggia e aspra e forte che nel pensier rinova la paura!

diventa:

Qho;phùùr;gho;fdpplq;gl;qrvwud;ylwd;pl;ulwurydl;shu;xqd;vhoyd;rvfxud;fk:;od;glulwwd;yld;hud;vpduulwd?;Dkl;txdqwr;d;glu;txdo;hud;.;frvd;gxud;hvwd;vhoyd;vhoydjjld;h;dvsud;h;iruwh;fkh;qho;shqvlhu;ulqryd;od;sdxud2

Se sommiamo 6 invece di 3 otteniamo:

Tkr0sk u0jkr0igssot0jo0tuyzxg0àozg0so0xozxuàgo0vkx0òtg0ykràg0uyiòxg0in!0rg0joxozzg0àog0kxg0ysgxxozg10Gno0wògtzu0g0jox0wògr0kxg0?0iuyg0jòxg0kyzg0ykràg0ykràgmmog0k0gyvxg0k0luxzk0ink0tkr0vktyokx0xotuàg0rg0vgòxg5

Ci sono poche chiavi!

Abbiamo a disposizione soltanto 72 chiavi, i numeri da 1 a 72

Lo spazio delle chiavi

Identifichiamo un messaggio M con una successione finita di numeri compresi, nel nostro esempio, tra 0 e 72.

$$M = x_1\, x_2\, \ldots x_n$$

Lo spazio K delle chiavi contiene 72 elementi, gli interi 1, ... 72.
Per ogni chiave k c'è una *funzione di cifratura* Φ_k così definita:

$$\Phi_k(M) = mod(M + k, 73)$$

Nel codice di Cesare si ha $k = 3$.
Per ogni chiave k c'è anche una *funzione che decifra* D_k che ovviamente è: $D_k(M) = mod(M - k, 73)$
Per esempio la parola 'Gatto' diventa il messaggio $M = 22\,42\,61\,61\,56$. Se scegliamo $k = 15$ si ha:

$$\Phi_{15}(M) = 37\,57\,3\,3\,71$$

Il metodo di esaustione

Quando le chiavi sono poche, per scoprire il messaggio è sufficiente provarle tutte una dopo l'altra!
In effetti finora abbiamo usato soltanto 72 permutazioni cicliche dei numeri 0 ... 72.
Nessuno ci impedisce di utilizzare una generica permutazione σ di questi numeri. In questo caso avremo

$$\Phi_\sigma(M) = \sigma(x_1)\,\sigma(x_2)\ldots\sigma(x_n)$$

Lo spazio delle chiavi è immenso

Ci sono 73! permutazioni su 73 simboli. L'ordine dello spazio delle chiavi è circa 4.47×10^{105}.
E' uno spazio inesauribile, non è più possibile violare il codice con la sola forza bruta.

L'indagine statistica

In ogni linguaggio le lettere dell'alfabeto appaiono con una certa frequenza caratteristica.

Il metodo di sostituzione che abbiamo appena visto non altera la *statistica* del messaggio.
Se in M la lettera e (in codice il numero 46) è quella che compare più volte, in $\Phi_\sigma(M)$ il simbolo più frequente sarà quello che è codificato dal numero $\sigma(46)$.

Se M non è troppo breve questo permette una facile decodifica da parte dell'avversario.

The Gold Bug, 1843

Un esempio molto bello di applicazione di questa tecnica si trova nel racconto di Edgar Allan Poe Lo scarabeo d'oro, che ottenne grande successo.

I codici di sostituzione polialfabetica

Si può dire che la crittografia moderna sia nata nel 1586, quando il diplomatico francese Blaise de Vigenère (1523-1596) pubblicò il cifrario che porta il suo nome.
L'idea di Vigenere è quella di utilizzare più di una permutazione dell'alfabeto. Se abbiamo k permutazioni $\sigma_1\,\sigma_2\ldots\sigma_k$:
$\Phi_\sigma(M) = \sigma_1(x_1)\,\sigma_2(x_2)\ldots\sigma_k(x_k)\,\sigma_1(x_{k+1})\,\sigma_2(x_{k+2})\ldots$

La parola chiave è mela

c	*i*	*v*	*e*	*d*	*i*	*a*	*m*	*o*	*d*	*o*	*m*	*a*	*n*	*i*
m	*e*	*l*	*a*	*m*	*e*	*l*	*a*	*m*	*e*	*l*	*a*	*m*	*e*	*l*
J	*H*	*b*	9	*K*	*H*	*G*	*H*	*V*	*C*	*U*	*H*	*H*	*M*	*O*

In realtà si sommano interi modulo 73

44	50	63	46	45	50	42	54	56	45	56	54	42	55	50
54	46	53	42	54	46	53	42	54	46	53	42	54	46	53
25	23	43	15	26	23	22	23	37	18	36	23	23	28	30

Come rompere i codici di sostituzione polialfabetica

L'idea di Vigenère è molto efficace. La statistica del messaggio viene completamente distrutta. Risulta inutile una indagine basata sulle frequenze relative delle lettere. Però nel 1863 Friedrich Kasiski comprese che, se è nota la lunghezza k della parola chiave, il problema di rompere un codice di sostituzione polialfabetica si può ridurre a quello di decifrare k codici monoalfabetici.
Infatti i simboli $x_1, x_{k+1}, x_{2k+1}, \ldots, x_{hk+1}$ saranno cifrati dalla permutazione σ_1, i simboli $x_2, x_{k+2}, x_{2k+2}, \ldots, x_{hk+2}$ saranno cifrati dalla permutazione σ_2, e così via.
Si tratta di una idea veramente *algebrica*: un codice complesso viene *spezzato* nella somma di k codici semplici.

Metodo di Kasiski

Se la parola chiave ha lunghezza k, si costruiscono - prendendo una lettera ogni k - k messaggi, ognuno dei quali è stato codificato con una singola permutazione σ_i.
Su ognuno dei k messaggi trovati si utilizza la indagine statistica.

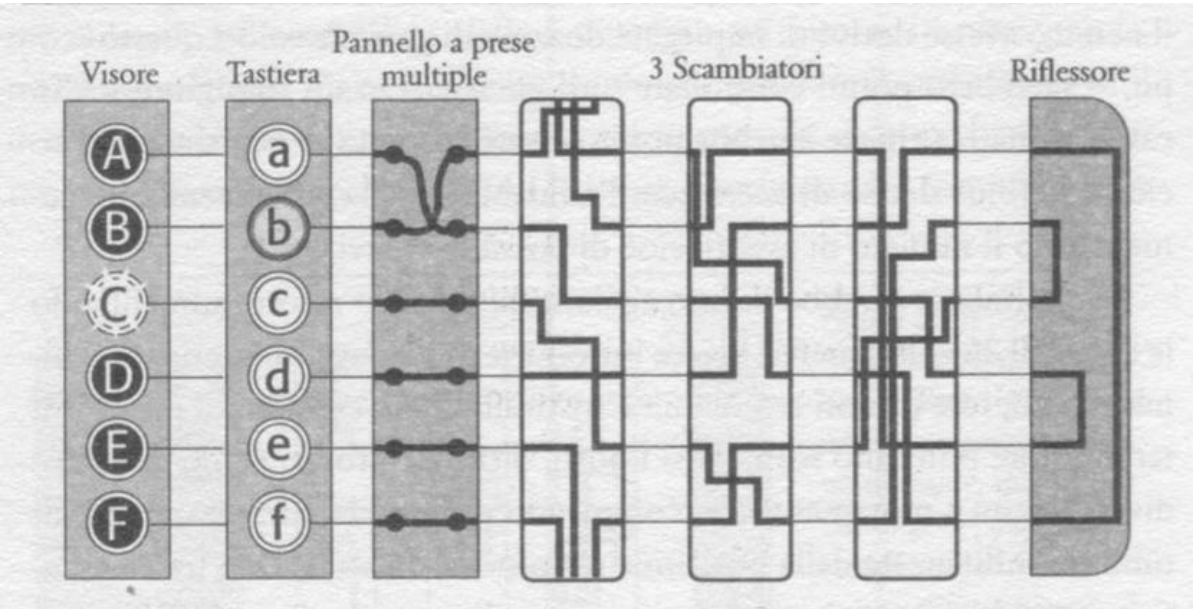
La chiave deve essere lunga

Kasiski stesso descrisse un metodo per trovare la lunghezza della parola chiave, basato sulla distanza tra segmenti uguali nel messaggio cifrato. Se, per esempio, *xyz* appare più volte a distanze 10 e 15, è probabile che la lunghezza sia 5.
Esistono metodi molto efficienti per trovare la lunghezza, come *l'indice di coincidenza I* di Friedman (1922). *I* è la frequenza delle coppie di simboli uguali.
Se ci sono q simboli e il testo è del tutto casuale, si ha $I = \frac{1}{q} = Q$.
Se il testo T è scritto in un dato linguaggio L, si ha
$I = \sum_1^q p_i^2 = P_L$, dove p_i è la frequenza caratteristica del simbolo i-esimo in L.
Se T vene messo in cifra con cifrario polialfabetico e chiave di lunghezza λ, si avrà $I \sim P_L$ se $\lambda = 1$ mentre I si avvicinerà sempre più a Q all'aumentare di λ. Da queste considerazioni si ricava una formula che dà I con ottima approssimazione, se il testo è abbastanza lungo.

La macchina Enigma

La macchina a rotori Enigma venne usata dai Tedeschi durante la Seconda Guerra Mondiale.

I rotori

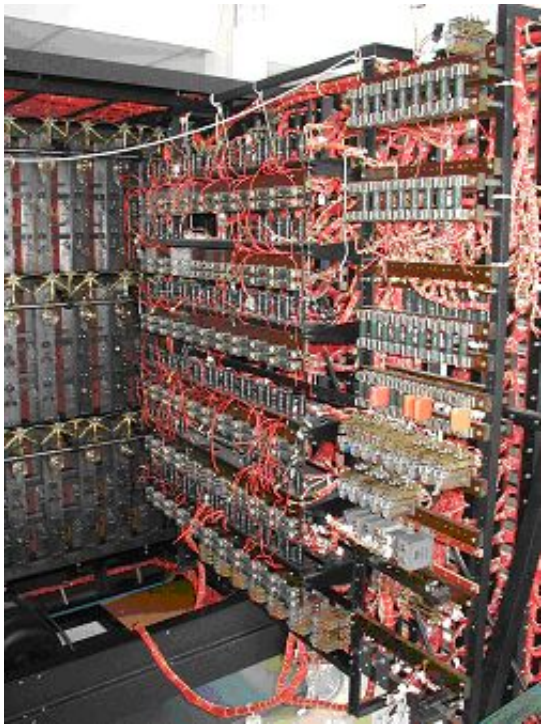


Premendo un tasto un rotore si spostava di uno scatto, e la permutazione cambiava. Le permutazioni si ripetevano, con tre rotori, soltanto dopo $26 \times 26 \times 26 = 15.756$ lettere. Dunque la lunghezza della parola chiave, ovvero il numero delle permutazioni utilizzate in sequenza era *uguale alla lunghezza del messaggio*, se il messaggio non era lunghissimo.

La chiave deve anche essere casuale

Nei casi di messaggi particolarmente importanti si aumentava il numero dei rotori.
La sequenza delle permutazioni, quella che abbiamo chiamato parola chiave, non veniva comunicata direttamente. Essa era automatica conseguenza della disposizione iniziale dei rotori, del loro ordinamento, e di altri dati, come le lettere abbinate dal cosiddetto pannello a prese multiple. Questo insieme di dati costituiva la chiave segreta che veniva condivisa dagli utenti. Lo spazio delle chiavi, con tre rotori, conteneva 10.586.916.764.424.000 elementi. Era dunque veramente grande, e le chiavi venivano cambiate ogni giorno.
Sembrava un codice impossibile da rompere.

La bomba di Turing e Welchman



La correlazione esistente tra le permutazioni successive e altri errori fatti dai Tedeschi permisero al team diretto dal grande matematico Alan Turing e da Gordon Welchman di rompere il codice Enigma. Furono usati i primi calcolatori elettrici, e particolari macchine, dette bombe, ognuna delle quali simulava il lavoro di dodici macchine Enigma.

Secondo gli storici la rottura del codice Enigma fu di capitale importanza per le sorti della guerra.

I codici perfetti

Gilbert Vernam introdusse e brevettò questi codici nel 1917, mentre lavorava alla ATT.
Ricordiamo che lo XOR tra due bit ,0 o 1, è la loro somma modulo 2 $(1 + 1 = 0)$.
Dato un messaggio M formato da n bit $M = b_1\, b_2\, \ldots b_n$, e data una chiave K consistente in una sequenza casuale di bit della stessa lunghezza del messaggio, il messaggio cifrato C è, per definizione, lo XOR bit per bit di M e di K. Denotiamo questa operazione, tra stringhe di bit, con $\oplus$.

Esempio

$$\begin{array}{r} M = 10011011100000111111 \\ K = 00101110010110000100 \\ \hline C = 10110101110110111011 \end{array}$$

Chi intercetta $C = M \oplus K$ non può ottenere alcuna informazione su M, senza conoscere parzialmente K.

La perfezione non è di questo mondo, però ...

Se chiave è casuale ed è lunga come il messaggio possiamo stare tranquilli. Del resto la chiave deve essere nota agli utenti che vogliono comunicare. Inoltre la chiave deve essere segreta. Ma se abbiamo un canale sicuro per inviare n bit di chiave, tanto vale inviare direttamente n bit di messaggio in chiaro!
Il brevetto di Vernam sembrerebbe totalmente inutile!
In realtà ci sono almeno due modi diversi modi per utilizzare i codici perfetti:

1. A e B si incontrano e creano due copie di un dvd (o altro supporto) che contiene una sequenza di gigabyte di bit casuali. Questi poi vengono utilizzati, con apposito software, da A e B dalle loro rispettive postazioni, nel corso di molteplici comunicazioni.
2. Rimanendo nelle loro posizioni, A e B generano a distanza la *medesima* successione di bit pseudocasuali.

Un modello per tutti: il generatore di Blum - Blum - Shub.

Blum - Blum - Shub

Siano dati due interi u_0 e n, coprimi tra loro.
Si definisca la successione ricorsiva:

$$u_k = mod(u_{k-1}^2, n)$$

u_0 e n costituiscono rispettivamente il *seme* e il *modulo* della sequenza.
Detto b_n il bit meno significativo di u_n, la successione dei b_n viene utilizzata come sequenza di bit pseudocasuali.

Per comunicare tra loro con un codice (pseudo) perfetto, gli utenti devono solo condividere il seme e il modulo.

Il crack del DES

Il 13 Luglio 1998, a San Francisco, venne rotto per la prima volta il Data Encryption Standard, con chiave di 64 bit (attacco a forza bruta).

Questo fatto venne riportato con grande clamore dalla stampa internazionale, anche se sembrava ancora remoto il rischio di incidenti del genere quando si fossero utilizzate, con il DES, chiavi a 128 bit.

Il NIST (National Institute of Standard and Technology) scelse il DES come standard di cifratura nel 1978 con validità quinquennale. Fu riaffermato nel 1983, nel 1988 ed infine nel 1993. Venne rotto pochi mesi prima della scadenza. Continuò comunque il suo lavoro, in versione rafforzata, fino al 2002, quando venne scelto il nuovo AES (Advanced Encryption Standard). L'AES è un codice algebrico, uttilizza polinomi con coefficienti nel campo finito GF(8). Le chiavi possono avere lunghezza 128, 192 o 256 bit (il sistema supporta chiavi di ogni lunghezza multipla di 4 con pochi aggiustamenti).

I sistemi del tipo DES, AES e molti altri sono detti *simmetrici*, perché chi possiede la chiave può sia cifrare che decifrare i messaggi. In altri termini la funzione di cifratura, se nota, è facilmente invertibile. Per questo motivo la chiave deve restare segreta.

Il problema dello scambio delle chiavi

Attualmente la Crittografia risulta uno strumento indispensabile nella trasmissione e nella archiviazione di dati riservati. Si pensi solo alle operazioni bancarie eseguite sulla rete, alle comunicazioni industriali, finanziarie e militari. Vi sono in questo campo enormi investimenti di capitali e di intelligenze. Specialmente negli USA molti tra i più brillanti giovani matematici, fisici e informatici vengono assunti per difendere i canali più riservati e per decifrare i messaggi segreti del nemico (anche al di là di guerre in corso, si pensi alle associazioni di stampo mafioso e al terrorismo internazionale).
Più modestamente la sicurezza delle nostre Bancomat e dei nostri risparmi, è affidata alla crittografia.
Nei sistemi ad alta sicurezza le chiavi devono essere sempre diverse.
Nel caso di sistemi simmetrici questo comporta un intenso *scambio di chiavi* tra gli utenti. Questo scambio deve rimanere segreto.
Per non entrare in un circolo vizioso si dovrebbe poter scambiare le chiavi senza bisogno di ulteriori chiavi ...

Il doppio lucchetto

Supponiamo che A e B vogliano scambiarsi una chiave K per poter poi comunicare dati riservati con il sistema AES.
A per ipotesi possiede un sistema privato di cifratura, noto solo a lui ed estremamente sicuro, formato dalla coppia Φ_A (la funzione che cifra) e D_A (la funzione che decifra). Allo stesso modo B ha Φ_B e D_B.
A sceglie una chiave K e la mette in cifra (chiude il suo lucchetto) ottenendo $S = \Phi_A(K)$. A manda S a B.
B non può leggere K. Chiude ora il suo lucchetto e manda $T = \Phi_B(S)$ ad A.
Ora A apre il suo lucchetto e invia a B $U = D_A(T)$.
Infine B calcola $D_B(U)$ e ritrova K.
Sulla rete, visibili a tutti, viaggiano $\Phi_A(K)$, $\Phi_B(\Phi_A(K))$ e $D_A(\Phi_B(\Phi_A(K)))$
Ognuno di questi tre messaggi è chiuso da almeno un lucchetto, che sappiamo inviolabile.

La duplice debolezza del doppio lucchetto

La tecnica del doppio lucchetto non richiede la condivisione di una chiave, e sembra risolvere il problema dello scambio delle chiavi. Però:

- Affinché si abbia $\Phi_B(K) = D_A(\Phi_B(\Phi_A(K)))$ occorre che $D_A(\Phi_B(\Phi_A(K))) = \Phi_B(D_A(\Phi_A(K)))$ oppure che $D_A(\Phi_B(\Phi_A(K))) = D_A(\Phi_A(\Phi_B(K)))$. Pertanto Φ_B deve commutare con D_A o con Φ_A. I due sistemi *personali* non possono essere indipendenti.
- Questo metodo è sensibile ad un attacco attivo dell'avversario C.
  C intecetta il primo messaggio $S = \Phi_A(K)$, lo chiude con il suo lucchetto e manda ad A $T^* = \Phi_C(S)$, fingendo di essere B. Ingenuamente A apre T^* e invia a B $U^* = D_A(T^*)$. Il malvagio C intecetta U^*. Ma a questo punto $U^* = \Phi_C(K)$. C apre con D_C e ottiene K.

Tre passaggi possono essere troppi

Supponiamo che esista un processo di autenticazione, per cui l'attacco attivo prima descritto non è possibile.
A e B sanno con certezza con chi stanno parlando.
Si potrebbero usare, a livello di codici personali, addirittura i codici perfetti!
Se A e B vogliono scambiarsi una chiave di, diciamo, 128 bit possono agire così:

Perfezione in atto

A sceglie K, stringa binaria di 128 bit. A ottiene, anche da un supporto fisico, una stringa *casuale* X di 128 bit, e invia a B $S = K \oplus X$. B mette 128 bit casuali in Y e invia ad A $T = S \oplus Y$. Infine A manda a B $U = T \oplus X$. Ora B possiede $U = T \oplus X = S \oplus Y \oplus X = K \oplus X \oplus Y \oplus X$.

Fine di un'illusione

Ricordiamo che lo $\oplus$ possiede tre ovvie proprietà:

- E' commutativo: $X \oplus Y = Y \oplus X$
- La stringa O formata da zeri è elemento neutro: $X \oplus O = X$
- Ogni stringa è opposta di se stessa: $X \oplus X = O$

Segue che B possiede $U = K \oplus X \oplus Y \oplus X = K \oplus Y$. B calcola $U \oplus Y$ e ottiene K.
Sulla rete sono passati S, T e U. Chi legga singolarmente questi messaggi non può avere nessuna informazione.
Però se C li possiede tutti C calcola:
$S \oplus T \oplus U = S \oplus T \oplus (T \oplus X) = S \oplus X = (K \oplus X) \oplus X = K$

Facile e difficile

Nel seguito consideriamo problemi che riguardano interi.
Diciamo che un certo problema, dipendente da un intero n, è facile (o trattabile) se conosciamo un algoritmo che lo risolve in un tempo T(n) tale che si ha $T(n) \leq C\, log(n)^k$, con C e k costanti e indipendenti da n.
Diciamo poi che un problema è difficile (o intrattabile) se non è facile.

Problemi facili

- Dato $n = a^b$, con $a > 1$ e $b > 1$ determinare a e b
- Dato n trovare il massimo comun divisore con intero qualsiasi m, $MCD(n, m)$
- Dati a, n coprimi (cioè con $MCD(a, n) = 1$), trovare l'inverso di a modulo n. Trovare cioè un intero b tale che il resto della disione di ab per n sia 1. Per esempio 4 è l'inverso di 3 modulo 11.
- Dati a, n, m calcolare $a^n \, (mod \, m)$
- (Problema Cinese dei Resti) Dato $n = m_1 m_2 \dots m_k$, dove gli m_i sono coprimi, e dati gli interi qualsiasi a_1, $a_2, \dots a_k$ trovare x che risolve $\forall i \, x \equiv a_i (mod \, m_i)$
- Dire se n è primo

Problemi difficili

1. (Logaritmo discreto) Noti a, n, $a^e \pmod n$ trovare l'esponente e
2. (Radice discreta) Noti e, n, $a^e \pmod n$ trovare la base a
3. (Quadraticità) Noti a, n dire se a è un quadrato modulo n
4. (Fattorizzazione) Fattorizzare n, ovvero scrivere n come prodotto di primi
5. (Problema di Diffie-Hellman) Noti n, $g^x \pmod n, g^y \pmod n$, trovare g^{xy}

Scambiarsi le chiavi secondo Diffie - Hellman

Questo metodo fu il primo pratico e efficiente sistema mai ideato (1976).

Gli utenti A e B vogliono condividere una chiave K.
Vengono scelti un primo n grande ed un intero $g < n$.

p e g sono resi *pubblici*.

A calcola un intero random x e invia a B $g^x \, (mod \, n)$
B calcola un intero random y e invia ad A $g^y \, (mod \, n)$
A calcola $U = (g^y)^x \, (mod \, n) = g^{yx} \, (mod \, n)$
B calcola $V = (g^x)^y \, (mod \, n) = g^{xy} \, (mod \, n)$

Sembra impossibile ma ...

Poiché $g^{yx} = g^{xy}$, $U = V$. I due utenti condividono ora uno stesso intero $K = U = V$ che nessun altro può conoscere.

La crittografia a chiave pubblica

Nel 1976, nell'articolo New Directions in Cryptography, Whit Diffie e Martin Hellman introdussero l'idea assolutamente innovativa di crittografia asimmetrica o crittografia a chiave publica

Difficile tornare indietro

Il modo migliore per evitare di lo scambio delle chiavi è quello di renderle pubbliche!
La funzione cifrante di A, Φ_A è nota a tutti!
Essa però è difficile da invertire, senza la conoscenza di una seconda chiave, nota soltanto ad A.
Soltanto A può calcolare facilmente la funzione inversa di Φ_A, cioè D_A.

Ma, esistono funzioni così!?

Autenticazione

Io son Beatrice, che ti faccio andare...

La crittografia asimmetrica permette la *autenticazione* del messaggio M.
Se A vuole inviare un messaggio autenticato M a B, prende la funzione pubblica di cifratura Φ_B di B, e manda a B $S = D_A(\Phi_B(M))$, dicendogli: io sono A.
B calcola $D_B(\Phi_A(S)) = D_B(\Phi_A(D_A(\Phi_B(M))))$ (si ricordi che anche Φ_A è pubblica).
Se il risultato è valido, B è sicuro che il mittente è A, perché solo A conosce D_A.

Per lo stesso motivo, la seconda cifratura con D_A ottiene tre altri risultati:

1. A non può negare di avere inviato il messaggio a B
2. B non può sostenere di avere ricevuto un messaggio diverso
3. Chi intercetta il messaggio non può alterarlo

Fermat e Eulero

La *funzione di Eulero* e così definita:

$$\phi(N) = |\{a < N : MCD(a, N) = 1\}|$$

Uno dei tanti di teoremi di Eulero ...

Eulero nel 1736 dimostrò che, se $MCD(a, N) = 1$:

$$a^{\phi(N)} \equiv 1 \, (mod \, N)$$

Circa un secolo prima, nel 1640, Fermat aveva affermato senza dimostrazione che, se p è un numero primo che non divide a:

Il Piccolo Teorema di Fermat

$$a^{p-1} \equiv 1 \, (mod \, N)$$

E' chiaro che il PTF è un caso particolarissimo del Teorema di Eulero.

Il metodo RSA (1977) è stato uno dei primi sistemi crittografici a *chiave pubblica*.

Rivest, Shamir e Adleman

L'utente A trova due primi grandi p e q, e calcola $n = p \times q$.
Prende e random coprimo con $\phi(n) = (p-1)(q-1)$, e calcola $d : ed \equiv 1 \, (mod \, \phi(n))$.
Questo significa che $ed = 1 + k\phi(n)$. A pubblica n ed e.
Se B vuole inviare un messaggio m ad A (m è rappresentato da un intero coprimo con n e $< n$), calcola $c = m^e \, (mod \, n)$ e lo manda ad A.
A riceve c e calcola $c^d \, (mod \, n)$, recuperando così m.
Il tutto funziona per il teorema di Eulero:
$c^d = (m^e)^d = m^{ed} = m^{1+k\phi(n)} = m \times (m^{\phi(n)})^k \equiv m \, (mod \, n)$.

Malgrado siano passati 30 anni, l'RSA continua ad essere considerato uno dei sistemi più sicuri.

Filtrare i numeri con il PTF

L'esistenza stessa del sistema RSA si basa sull'abisso temporale che apparentemente separa il problema di trovare numeri primi grandi (problema facile) e quello di fattorizzare un numero (problema difficile).
Ma come si trovano in fretta numeri primi grandi?
Il metodo più semplice si basa sul Piccolo Teorema di Fermat.
Ricordiamo che il piccolo teorema di Fermat (PTF) asserisce che se N è primo:

$$(\mathbf{1})\ N \nmid a \Rightarrow a^{N-1} \equiv 1\,(mod\ N)$$

Per ogni (base) a si prova che esistono infiniti interi composti N che verificano la **(1)**.
Però se N è grande la probalitità che N passi il *test* **(1)** e al tempo stesso non sia primo è piccolissima.
Se diciamo $P(x)$ la probabilità che un numero random composto $N \leq x$ passi **(1)** su random $a < N$, si prova che:
$P(10^{100}) < 2.77 \times 10^{-8}$
$P(x) < (\log x)^{-197}$ se $x > 10^{10^5}$

Fattorizzare con la radice quadrata

Chiunque troverà un modo facile per fattorizzare i numeri avrà fama imperitura e diventerà ricco!
Tutti i metodi odierni sono inefficaci con interi di centinaia di cifre.
Molti di essi si basano su una idea di Lagrange: conoscere le radici quadrate di a modulo N.
Se N è primo e a è un quadrato modulo N l'equazione $x^2 \equiv a\,(mod\ N)$ ha due soluzioni $\pm b$.
Se N non è primo e a è un quadrato modulo N l'equazione $x^2 \equiv a\,(mod\ N)$ possiede almeno due soluzioni b e c essenzialmente diverse, tali cioè che $b \not\equiv \pm c\,(mod\ N)$.

Fattorizzazione di Lagrange

Se b e c sono noti, dal fatto che $b^2 \equiv c^2\,(mod\ N)$ si ricava che N divide $(b-c)(b+c)$
Per ipotesi $N \not| (b-c)$ e $N \not| (b+c)$, quindi MCD(a-b, N) e MCD(a+b, N) sono fattori propri di N.

L'importanza di essere radici quadrate di uno

Supponiamo che, come avviene nell'RSA, si abbia $N = p \times q$, p e q primi.
Consideriamo l'equazione $x^2 \equiv 1 \, (mod \, N)$.
Essa ha banalmante le soluzioni ± 1.
E ne possiede altre due, $\pm z$.
Se z è noto, N si fattorizza!

Esempio

$N = 40463931305392307$.
Vengo a sapere che $z = 16423002386099336 7$.
Calcolo MCD(N, z-1) e MCD(N, z+1).
Trovo, rispettivamente, 707830303 e 571653869.
Deduco che $N = 707830303 \times 571653869$.

Dollari e fattori

Sul sito RSA si trova un elenco di interi da fattorizzare (sono tutti prodotto di due primi), con relativi premi.

Il più grande è un intero di 2048 bit (617 cifre), e vale 200.000 dollari:

RSA-2048

25195908475657893494027183240048398571429282126204
03202777713783604366202070759555626401852588078440
69182906412495150821892985591491761845028084891200
72844992687392807287776735971418347270261896375014
97182469116507761337985909570009733045974880842840
17974291006424586918171951187461215151726546322822
16869987549182422433637259085141865462043576798423
38718477444792073993423658482382428119816381501067
48104516603773060562016196762561338441436038339044
14952634432190114657544454178424020924616515723350
77870774981712577246796292638635637328991215483143
81678998850404453640235273819513786365643912120103
97122822120720357