Ottavio Serra

Sicurezza informatica, sequenze casuali e numeri primi

Premessa.

Il problema della riservatezza delle comunicazioni e della segretezza dei dati nei campi più disparati: commercio, industria, finanza, servizi segreti civili e militari, si è posto fin dai tempi antichi. Si sa che Giulio Cesare riceveva informazioni e trasmetteva ordini ai suoi luogotenenti mediante un sistema di codifica rudimentale consistente nella traslazione (traslitterazione) delle lettere del messaggio, noto come *Codice Cesare*. In questo codice ogni lettera viene sostituita da quella che la segue dopo n posti nell'alfabeto, immaginato come una sequenza circolar, per cui dopo Z ritorna A. Per esempio, se poniamo n=3, A viene sostituita da D, B da E, eccetera, Z da C. La decodifica viene eseguita usando il codice in senso inverso. Per semplicità non venivano usati segni di interpunzione, né spazi tra le parole e solo lettere maiuscole. Ciò aumentava un po' la difficoltà di decodifica.

Un esempio con l'alfabeto inglese (*ma i nemici non lo sanno!*) è il seguente:

ODELHQRDWWDFFDLJDOOLDOODOED

Pare che in tempi recenti anche Bernardo Provenzano codificasse i suoi *pizzini* con un codice Cesare modificato, sostituendo, dopo la traslitterazione alfabetica, ogni lettera con il suo numero d'ordine: A→1, B→2, …, Z→21.

Chiaramente, un codice di tal fatta è facilmente violabile, né vale usare più traslitterazioni di seguito, perché si ottiene ancora una traslitterazione; *le traslitterazioni, infatti, formano un gruppo isomorfo al gruppo ciclico additivo modulo 26 (modulo 21, per la lingua italiana*).

Si potrebbe migliorare il codice, dotando mittente e destinatario di una ***tabella cartacea*** (da usare come chiave crittografica) contenente una delle possibili permutazione dell'alfabeto, una delle 26! permutazioni per l'alfabeto inglese.

Per una versione elettronica della ***tabella cartacea*** vedi il mio programma in DELPHI *Crea file di N (almeno 256) interi random* da usare con il programma principale *Codice Cesare Random*[1] (Non è pensabile, al giorno d'oggi, che certe cose si facciano a mano!). Il programma genera una chiave di 256 (o più) interi random compresi tra 32 (codice ASCII del carattere "spazio") e 251, in modo che si possano utilizzare tutti i caratteri tipografici. Una tale chiave random è una delle possibili disposizioni con ripetizione dei 220 numeri da 32 a 251 di classe 256.

Siccome 220^{256} è un numero di 530 cifre, per violare un tale codice, anche ad avere un computer fantascientifico che facesse 10^{30} tentativi al secondo, si impiegherebbe un tempo enormemente maggiore dell'età dell'universo.

Simili codici, tuttavia, non offrono una garanzia sufficiente per gli odierni standard di sicurezza: posta elettronica, carte di credito, transazioni bancarie, archiviazione di dati sensibili richiedono sistemi di codifica e decodifica che siano allo stesso tempo veloci nella trasmissione di informazioni e praticamente inviolabili da parte di estranei. Infatti la chiave (la tabella) dovrebbe essere posseduta da due persone (mittente e destinatario), ma come dice un proverbio, *un segreto condiviso da due persone non è più un segreto*.

Anche i più elaborati cifrari basati su corrispondenze prefissate tra le lettere dell'alfabeto, se pure periodicamente variate, si pensi alla macchina "*Enigma"* usata da tedeschi durante la seconda guerra mondiale, sono vulnerabili per il fatto che, fin quando si usa la stessa chiave, lettere uguali sono

[1] Vedi *digilander.libero.it/ottavioserra0 (programmi eseguibili, cartella Delphi numeri 30 e 31).*

codificate con lettere uguali e pertanto si possono "*indovinare*" le lettere più frequentemente usate in una data lingua (per esempio, la lettera "E" seguita dalla "T" nella lingua inglese, le vocali esclusa la "U" in italiano). Sfruttando la frequenza statistica delle lettere alfabetiche della lingua tedesca, il gruppo di scienziati inglesi (matematici, linguisti e crittografi) guidati da Alan Turing[2] forzò il cifrario di Enigma (aiutò il fatto che il servizio di spionaggio inglese riuscì a impadronirsi di un esemplare della macchina).
Il mio "Codice Cesare" è però praticamente inviolabile, perché ogni carattere del testo originale è traslato di un numero casuale variabile da carattere a carattere; se, inoltre, mittente e destinatario si dotano ciascuno di una chiave random che l'altro non conosce, i pro e i contro sono identici a quelli della codifica "Xor" a chiave casuale che presenterò nel prossimo paragrafo.
Illustrerò ora due metodi di cifratura: l'uno, che chiamerò a "***Chiave casuale***", molto semplice dal punto di vista dell'implementazione informatica e particolarmente **comodo** (e **sicuro**) per proteggere dati personali (*Pin di bancomat e carte di credito, password per accesso al conto bancario e ad altri servizi on line*), utilizzata da me per cifrare i miei dati; meno comodo per trasmissioni riservate tra un mittente e un destinatario, perché richiede, come vedremo tra breve, tre trasmissioni; l'altro, detto a "***Chiave pubblica***", universalmente adoperata negli interscambi di informazioni nei vari campi delle attività civili e militari.

1. Codifica (cifratura) e decodifica a Chiave Casuale.

La funzione "*Xorl*", presente in molti linguaggi di programmazione, realizza il connettivo espresso dal latino "*aut*" (il così detto "*o*" *esclusivo*: *una e una sola di due alternative è vera*). In altre parole, se A e B sono due asserzioni, A *Xor* B è vera se una delle due asserzioni è vera e l'altra è falsa; è falsa se le due asserzioni sono entrambe vere o entrambe false. Nel caso che A e B siano due bit, 0 e 1, (0=False, 1=True) A *Xor* B realizza l'addizione binaria (cioè in base due), senza riporto: 1 + 0 = 0 + 1 = 1, 0 + 0 = 1 + 1 = 0.
Se *Xor* non è disponibile, si può facilmente realizzare con i connettivi usuali *Or*, *And* e *Not*.
Per la cifratura di un testo T si procede come segue, dopo aver imposto una limitazione ragionevole alla lunghezza (numero dei caratteri) di T: per esempio 256 caratteri, che è la lunghezza standard del tipo "String" del Pascal o di Delphi, coerentemente col fatto che ogni carattere è codificato in ASCII (American Standar Code Information Interchange) con 8 bit e $2^8 = 256$.
Per prima cosa si costruisce e si salva su disco una sequenza "***random***" (pseudo ***casuale***) S di almeno 256 numeri naturali (*La Chiave casuale*). Poi con un altro programma si legge da tastiera o da disco il testo T e, per k variabile da 1 alla lunghezza di T, si fa lo *Xor* tra il codice ASCII del k^{mo} carattere di T e del k^{mo} numero della chiave Random S. Si salva su disco questa sequenza numerica, che chiameremo TS ed è il testo cifrato. (Si noti che sarebbe stato più logico usare la notazione additiva T+S, come suggerisce la tavola di verità di *Xor* riportata più sopra, ma TS è più breve; e poi basta intenderci).
Per ottenere il testo in chiaro, cioè per decifrare TS, basta applicare a TS il precedente processo di cifratura: TSS=T, perché SS è l'identità, lo zero dell'addizione; infatti, se b è un bit, qualunque sia il suo valore, b + b = 0.
Naturalmente, T non è ancora il testo originale in caratteri alfabetici, ma la sequenza dei codici ASCII dei suoi caratteri; l'ultimo passo è perciò tradurre i codici ASCII in caratteri.

[2] Alan Turing, 1912-1954, grande matematico inglese, fondatore dell'informatica teorica.

Io ho creato i seguenti programmi: *NatRnd* (Naturali Random) per costruire la chiave (la sequenza) S e *Cripto* per cifrare e decifrare i testi. Posso affidare a chiunque sia i due programmi, sia il testo cifrato TS; l'unica possibilità di decifrare TS è di possedere la "CHIAVE DI CIFRATURA" S, che perciò va gelosamente custodita. Se mi venisse il sospetto che un estraneo si sia impossessato di S, dovrei al più presto generare con NatRnd una nuova chiave S', decifrare con la chiave S i testi cifrati e cifrarli daccapo con la nuova chiave S', che sostituirà S.
Vediamo ora come si può usare il metodo della Chiave casuale per intrattenere una corrispondenza segreta tra due persone, un mittente M e un destinatario D. (Si noti che M e D sono interscambiabili).
Le due persone si dotano entrambe dei programmi NatRnd e Cripto (che io ho pubblicato nel mio sito). Quando M deve inviare un messaggio *T* a D, per prima cosa crea una chiave M (la sua sequenza Random S), poi cifra T in TM e spedisce TM a D per posta elettronica o con altro mezzo (non importa quanto affidabile).
Il destinatario D crea con la sua copia di NatRnd (o con altro programma) la sua sequenza Random D, cifra TM con la sua copia di Cripto e ottiene TMD, che rispedisce al mittente M. Questi cifra con M e ottiene TMDM, che rispedisce a D. Siccome *Xor* è associativa e commutativa, TMDM=TDMM=TD. Infine D decifra TD con la sua chiave D e ottiene TDD=T, che è il messaggio in chiaro. A questo punto M e D possono distruggere le loro chiavi, per crearne di nuove al prossimo scambio di messaggi.
Lo schema è il seguente: M. (Mittente) → D. (Destinatario) → M. → D.
M CIFRA (Chiave M)→D CIFRA (Chiave D)→M CIFRA (Chiave M)→D DECIFRA (Chiave D).
Il metodo, a mio avviso, è assolutamente sicuro, ma richiede tre trasmissioni: da M a D, da D a M e da M a D. Ciò fa sì che, mentre va benissimo per conservare con assoluta tranquillità i propri dati sensibili (tutto sta a tenere ben nascosta la chiave S), non va bene per le transazioni commerciali e finanziarie e per le informative dei servizi segreti e faccende analoghe.
Nota. Si può usare anche il codice Cesare "*random*" per lo scambio di informazioni riservate, se mittente M e destinatario D si dotano ciascuno di una chiave "RND26"; lo schema è il seguente:
M cifra →D cifra →M decifra →D decifra.
Per evitare la macchinosità delle tre trasmissioni negli interscambi di informazioni, si ricorre al metodo cosiddetto di ***cifratura a chiave pubblica***, messo a punto negli anni '80 del novecento e continuamente aggiornato man mano che i calcolatori diventavano più potenti e veloci.
Tale metodo richiede una sola trasmissione, invece di tre.

2. Cifratura a chiave pubblica.

Questo metodo si basa su una nota proprietà aritmetica: ***mentre è facile moltiplicare o dividere due numeri naturali, la fattorizzazione[3] di un numero molto grande può essere praticamente impossibile, anche per un supercomputer.***
Si procede così: si cercano due numeri primi p e q di un centinaio di cifre ciascuno (io uso due numeri primi di circa 25 cifre ciascuno (in un'altra versione uso numeri di 45 cifre), trovati *giocando* col mio programma "*TestFerm*", Test di primalità basato sul piccolo teorema di Fermat, ma si può

[3] Ritengo veramente incredibile che nelle scuole si insegni a trovare il massimo comun divisore mediante scomposizione in fattori primi, algoritmo di complessità esponenziale, quando esiste il bellissimo e rapidissimo algoritmo Euclideo delle divisioni successive.

smanettare un po' anche su "Mathematica"; il loro prodotto n=pq sarà la *base "modulare"* delle chiavi: pubblica e privata. Si calcola poi l'indicatore di Eulero – Gauss, cioè il numero m=φ(n)=(p-1)(q-1), si cerca un numero c abbastanza grande che sia primo con m e si calcola il numero x tale che cx≡1 (Modulo m). Io uso il mio programma *Cprivata* (Chiave privata). Se si utilizza x come chiave privata (segreta), c sarà la chiave pubblica (o viceversa).

La coppia di numeri (n,c) è la chiave pubblica completa che uno rende nota, pubblicandola sul proprio sito internet, sui giornali o facendola recapitare ai propri corrispondenti. Una persona che vuole mandarmi un messaggio T in modo riservato cifra T eseguendo le seguenti operazioni: 1° sostituisce T con la stringa dei codici ASCII dei suoi caratteri; 2° trasforma la stringa numerica in un numero naturale T_{nat} (che ha un numero di cifre pari al doppio dei caratteri di T[4]); 3° calcola la potenza $P=(T_{nat})^c$ (Modulo n). P è il messaggio cifrato che tutti possono vedere, ma non decifrare. Per decifrare P occorre la chiave privata (segreta) x: $P^x=(T_{nat})^{cx}$ (Modulo n), che è uguale a T_{nat}, perché xc=1 (modulo m). Naturalmente T_{nat} è un numero naturale che va trasformato nella stringa di caratteri del testo originale in chiaro T, raggruppando le cifre a due a due, ogni coppia essendo il codice ASCII di un carattere. (Per semplicità io utilizzo i codici di due cifre, che mi consentono di utilizzare le lettere maiuscole, le cifre, i segni delle operazioni aritmetiche e i caratteri di punteggiatura usuali; per questo motivo il testo decifrato appare scritto in caratteri maiuscoli, anche se nell'originale erano presenti lettere minuscole. Il mio programma che realizza tutto questo è "*CifraKey*". Siccome il modulo n che utilizzo è di 50 cifre, il testo T non può essere più lungo di 25 caratteri; è poco, ma il mio è un demo, per illustrare il principio della crittografia a chiave pubblica. Ho esteso *Cifrakey* in *CifrakeZ*, che utilizza un modulo n di 90 cifre; ciò consente di trattare testi fino a 45 caratteri, ma naturalmente su un normale pc la durata dell'elaborazione è sensibilmente maggiore.

(Per inciso, la cifratura col mio ***Cripto*** a chiave casuale è velocissima e tratta testi fino 255 caratteri, senza restrizioni sui codici ASCII).

Per una crittografia "***realistica***" a chiave pubblica occorrono due numeri primi p e q di, poniamo, 200 cifre ciascuno, il modulo n =pq sarà di 400 cifre e per decomporlo nei suoi due fattori (operazione necessaria per scoprire la chiave privata) una rete in parallelo di tutti i supercomputer esistenti di ultima generazione impiegherebbe più dell'età dell'universo.

Esempio di cifratura a chiave pubblica.

(è un demo con numeri estremamente piccoli; solo per capire il funzionamento del metodo).

[4] Se si utilizzano caratteri che abbiano codici ASCII di due cifre, per esempio fino a "Z", che ha codice 90.

```
p := 7  Scelgo p

q := 11 Scelgo q

n := p * q

n   Base modulare

77

m := (p - 1) * (q - 1)

m   Indic di Gauss

60

c := 13  Chiave privata

x := 37 Chiave pubblic

Mod[c * x, m]

1   Perchè cx=1(Mod m)

T := 49  Testo originale
Tx := Mod[T^x, n]

Tx

14  Tx cifrato

Txc := Mod[Tx^c, n]

Txc

49   Txc Decifrato
```

(L'esempio è stato eseguito con "Mathematica").
m=φ(n) indicatore di Eulero – Gauss;
c (nell'esempio c=13) chiave privata; allora la chiave pubblica x deve essere tale che
cx≡1 (Mod. m).
Se T é il testo in chiaro, o meglio, la sua trasformata in numero come sequenza dei codici ASCII dei suoi caratteri, allora Tx = T^x (Mod. n) è il testo cifrato e Txc=$(T^x)^c$=T^{xc} (Mod. n) restituisce il testo T in chiaro, perché xc≡1 (Mod. m). In verità alla fine T va riconvertito in caratteri alfabetici, ma questa è un'operazione banale.
Consultando una tabella dei codici ASCII, o scrivendo un codice di poche righe, si vede che il testo T=49 è il codice ASCII del carattere "1". Perciò abbiamo fatto tutta questa gran fatica per criptare il numero "1". *Che cosa non si fa per seguir "**virtute e canoscenza**"!*
Chi vuol vedere esempi meno banali, può usare il mio programma "CifraKey", che però gira sotto *Windows XP* e *Vista* o sotto una Virtual Machine Windows XP, scaricabile dal sito di Oracle e installabile in Windows 7 o 10. Stessa avvertenza, se si vuole usare il mio programma Cripto scaricabile dal mio sito **digilander.libero.it/ottavioserra0 cartella Eseguibili sottocartella Didattici e vari**.

NOTA 1. Sul mio sito troverete la seguente versione di CifraKeY: "**Cifr2KY.EXE**" con i dati:
BASE e CHIAVE PUBBLICA per messaggi fino a 24 caratteri.
Base Modulare (prodotto di due numeri primi):
n=p.q=759755493443800835648638284166357427008031739597 79
Chiave pubblica c=4935668075858293223732962038334825910373728401 2881
Fattorizzando n trovate p e q, poi m=(p-1).(q-1) e infine potreste calcolare la **chiave privata** (SEGRETA) x tale che c.x≡1 (Modulo m).
Se riuscite a fattorizzare n ("Wolfram Mathematica" ci mette meno di 9 secondi) e poi a risolvere la congruenza per calcolare la chiave privata x, sarete in grado di cifrare e decifrare messaggi; se non ci riuscite, potrete cifrare un testo, ma poi non potrete più decifrarlo.

Stesso discorso per la versione "**Cifr2KZ.EXE"** di CifraKeZ:
BASE n e CHIAVE PUBBLICA c per messaggi fino a 44 caratteri.
Base n=
5485718521282194162954248156333395595906897098765200977000992646342270272965382 14963620843
Chiave pubblica c=
4374920472000461909874378464395510737987075823953517211707175482851246416869249 2937775381
Questa volta "WOLFRAM MATHEMATICA" non è riuscito a fattorizzare n (di 90 cifre) dopo un'ora di esecuzione, perciò dubito che riusiate a decifrare un messaggio cifrato con Cifr2KZ**.**
NOTA 2. I programmi citati sono realizzati in *Pascal* e si trovano nel mio sito *digilander.libero.it/ottavioserra0* Cartella *Programmi eseguibili* Sottocartella *Didattici e vari*.
I programmi di cifratura (e decifratura) con chiave casuale li ho implementati anche in Delphi (vedi programmi eseguibili in Delphi, nel mio sito).