Introduzione alla matematica della crittografia a chiave pubblica

Giovanni Franza
(AICA – Progetto EUCIP IT Administrator)

Una presentazione

- Giovanni Franza, classe '56
  - Collaborato alla progettazione di TelePay, uno dei primi sistemi di pagamento sicuro con carte di credito su Internet
  - Collaborato alla realizzazione di Ellips, una delle prime PKI in ambito bancario
  - Responsabile del coordinamento della definizione dei contenuti di EUCIP – IT Administrator, certificazione europea di competenze ICT.

Tutto è numero

- Crittografia classica
  - lavora sui simboli
    - principalmente su lettere e numeri
- Crittografia a chiave pubblica
  - lavora con i numeri
  - siccome i numeri sono limitati si lavora su pezzi di informazione, manipolati come numeri

Alice, Bob and Eve

- Quando si parla di cifra appaiono questi personaggi
  - Alice e Bob sono *personalizzazioni* di **A** e **B**
  - Eve è un'abbreviazione/corruzione di eavesdropper (quello che origlia)
  - Quindi Alice e Bob dovranno parlarsi senza che Eve riesca a capire cosa si dicono

Un primo esempio

- Alice spedisce il messaggio
  - in una cassa lucchettata
- Bob glielo rimanda
  - aggiungendo un suo lucchetto alla cassa
- Alice ri-spedisce il messaggio
  - Alice rimanda la cassa a Bob dopo avere levato il suo lucchetto
- Alla fine funziona
  - perchè c'è solo lucchetto di Bob.

Perché non funziona

- I lucchetti sono paralleli, la crittografia è seriale
  - mettere e togliere un lucchetto non influisce sugli altri
  - se cifro il testo due volte per riaverlo devo decifrarlo usando le due chiavi in ordine inverso a come le ho applicate.
    - testo > cifrato con a > cifrato con b > x
    - x > decifrato con b > decifrato con a > testo

L'utilità degli orologi

- Un utile strumento matematico:
  - I sistemi di numerazione chiusi
    - Esempio banale: i minuti dell'ora negli orologi
- Il suo compagno di strada:
  - I numeri primi
    - Sono distribuiti in maniera non facilmente predicibile (c'è la *congettura di Riemann* la cui dimostrazione o confutazione è oggetto di un premio cospicuo non ancora ritirato)

L'orologio con 13 ore

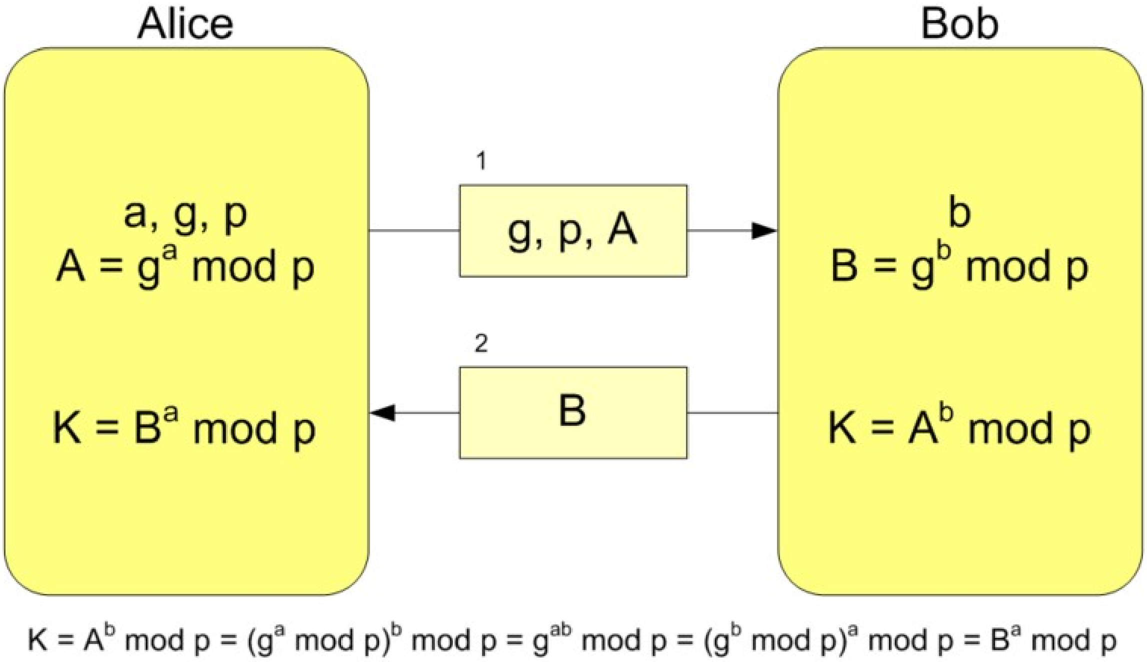
- Particolarmente importanti sono i sistemi di numerazione chiusi basati sui numeri primi
  - perchè hanno delle primitive
    - numeri le cui potenze generano, uno alla volta, tutti i numeri dell'insieme, secondo uno schema che non crea ripetizioni.
- Con loro si può creare una chiave condivisa

Diffie-Hellmann-Merkle

- Questi 3 signori hanno costruito un algoritmo che permette ad Alice e a Bob di condividere una chiave numerica senza che Eve la possa ricostruire
  - tutto si basa sulla matematica dell'orologio ovvero sui sistemi di numerazione chiusa

Diffie-Hellmann-Merkle

- **g** base, **p** modulo, **a b** parti della chiave **A B** dati trasmessi, **K** chiave condivisa
- $K = A^b \bmod p = (g^a \bmod p)^b = g^{ab} \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p$



Diffie-Hellmann-Merkle

- Alice spedisce **base**, **modulo** e il **resto** al modulo di (**base** elevata ad **a**) ovvero **A**
- Bob rispedisce il **resto** al modulo della (**base** elevata a **b**) ovvero **B**
- Eve ha **A** e **B** da cui non può risalire ne ad **a** ne **b** e quindi non può costruire **K**

Perchè Alice e Bob vedono?

- Perchè la matematica degli orologi è perversa
  - Alice ha **B** da Bob ma ha anche **a**
  - Bob ha **A** da Alice ma anche **b**
  - se eleviamo **B** alla **a** e facciamo il resto otteniamo la stessa cosa che elevando **A** alla **b** e facendo il resto
  - Ovviamente ciò vale per qualsialsi **a** e **b** che Alice e Bob scelgono ma solo per particolari basi e moduli

Perchè Eve è cieca?

- Perchè Alice e Bob non trasmettono **a** e **b** ma **un resto**
  - Dal resto non si può risalire alla potenza perchè non si sa quanti giri ha fatto l'orologio.
  - L'operazione di resto fa perdere parte dell'informazione

Dimostrazione - 1

- **Alice** calcola **(n^a)%m** e lo invia a **Bob**.
- **Bob** calcola **(n^b)%m** e lo invia ad **Alice**.
- **Alice** calcola **(((n^a)%m)^b)%m** per avere la chiave.
- **Bob** calcola **(((n^b)%m)^a)%m** per avere la chiave.
- Scopo della dimostrazione è dimostrare come i due calcoli portino al medesimo risultato.

Dimostrazione - 2

- Si inizi con l'osservare l'espressione **(n^a)%m**
- Estrarre il modulo è equivalente a sottrarre il più grande multiplo del modulo contenuto in **n^a**:
  (n^a)%m = n^a -k m
  dove **k = (n^a) / m**
- Elevando alla b-esima potenza entrambi i membri si ottiene
  ((n^a)%m)^b = (n^a - km)^b

Dimostrazione - 3

- Dato che il secondo termine è un'elevazione a potenza di un binomio, si può applicare la formula di Newton. In questa formula appaiono i due termini del binomio elevati a varie potenze e moltiplicati per dei coefficienti rappresentati nel triangolo di Tartaglia:
  (x + y)^z = x^z + t1 x^(z-1) y + t2 x^(z-2) y^2 + …… + y^z

Dimostrazione - 4

- Questa formula, che usa altre variabili, mostra come solo il primo termine **(y^z)** non sia multiplo del secondo termine **y**. Applicando questo concetto alla formula precedente si può scrivere:
  ((n^a)%m)^b = (n^a - km)^b
  ((n^a)%m)^b =(n^a)^b -sm
  dove s è un intero opportuno
- Osserviamo che il risultato è positivo e maggiore di zero.

Dimostrazione - 5

- A questo punto si può applicare l’operazione di modulo ad entrambi i membri dell’equazione ottenendo:
  (((n^a)%m)^b)%m=
  ((n^a)^b -sm)%m
- L’estrazione del modulo **m** nel secondo membro dell’espressione può essere rappresentata come:
  ((n^a)^b) %m

Dimostrazione - 6

- Concludendo ciò che si trova in mano Alice è:
  (((n^a)%m)^b)%m = ((n^a)^b)%m
  = (n^(ab))%n
- Per Bob basta scambiare **a** con **b**:
  (((n^b)%m)^a)%m=(n^(ba))%n
- E chiunque sa che **n^(ba)** è lo stesso numero di **n^(ab)** per cui rimane vero che il numero ottenuto da Alice è uguale a quello ottenuto da Bob.

Dimostrazione - 7

- Ma i numeri primi ? A cosa servono ?
  - Attenzione alle dimostrazioni, noi abbiamo diviso per **m**, se non fosse primo avremmo finito con il fare sfracelli ...

Tutto qui?

- Certamente no
  - l'algoritmo Diffie-Hellman-Merkle serve solo per condividere una chiave
  - risolve il problema della distribuzione delle chiavi
  - si deve appoggiare poi su algoritmi di cifratura simmetrica
    - non è necessariamente un limite: di solito si fa così anche perchè gli algoritmi simmetrici sono **veloci** e permettono di trattare **flussi interminabili di dati**.

Chiavi Pubbliche ?

- La matematica degli orologi permette anche di fare un'altra operazione
  - se uso opportunamente le cose posso usare la matematica degli orologi per trovare due esponenti tali che:
    - un qualunque numero elevato ad un certo esponente e sottoposto al resto mi da un altro numero
    - questo secondo numero, elevato al secondo esponente e sottoposto al resto mi ridà il numero di partenza

La ricetta delle chiavi - 1

- si scelgono a caso due numeri primi, **p** e **q** , abbastanza grandi da garantire la sicurezza dell'algoritmo
- si calcola il loro prodotto **n = p q** chiamato **modulo**
- si sceglie poi un numero **e** (chiamato esponente pubblico), più piccolo e coprimo (senza divisori comuni) con il prodotto **(p-1)(q-1)**

La ricetta delle chiavi - 2

- si calcola il numero **d** (chiamato esponente privato) tale che il resto di **e * d** al modulo **(p-1)(q-1)** sia **1**
- un messaggio **m** viene cifrato facendo il resto a **n** di (**m** elevato alla **e**) ottenendo così $\mathbf{c = m^e \% n}$
- il messaggio **c** viene decifrato facendo il modulo a **n** di (**c** elevato alla **d**) $\mathbf{m = c^d \% n}$

La ricetta delle chiavi - 3

- La chiave pubblica è costituita da **n** ed **e** mentre la chiave privata è costituita da **n** e **d**.
- I fattori **p** e **q** possono essere distrutti, anche se spesso vengono mantenuti all'interno della chiave privata.

Piccolo teorema di Fermat

- Per dimostrare matematicamente la correttezza dell’algoritmo RSA è utile prima dimostrare la correttezza del piccolo teorema di Fermat:
- Scelto un numero primo **p** e un numero intero qualsiasi **z** vale la relazione **(z^p)=z modulo p**.
- Se **z** non è multiplo di **p** vale anche la proprietà **(z^(p-1))%p=1**.

Dimostrazione prima parte

- Si dimostrerà che il teorema è vero per z=0.
- Si dimostrerà che se è vero per un certo valore Z, allora è anche vero per il valore Z+1.
  - Combinando le due dimostrazioni allora è vero per z=0 (prima) quindi per z=1 a causa della seconda e, dato che è vero per z=1 ciò lo rende vero anche per z=2 e così via, all'infinito.

Dimostrazione - 1

- Per dimostrare che il teorema è vero per **z=0** basta sostituire **0** a **z** nel teorema ottenendo:
  0^p=0 modulo p
- dato che 0 per 0 fa sempre 0 e che 0 diviso per qualsiasi numero diverso da 0 dà comunque 0 con resto 0, il teorema è sicuramente vero.

Dimostrazione - 2

- Per la seconda parte della dimostrazione si supponga che per un certo valore Z il teorema sia valido. Valga cioè l’equivalenza: **Z^p=Z modulo p**
- Si esprima ora il teorema per il valore **Z+1 : (Z+1)^p=(Z+1) modulo p**
- Se si dimostra che questa equivalenza è vera allora il teorema è dimostrato.

Dimostrazione - 3

- Si osservi l'elevazione a potenza **(Z+1)^p**. Si tratta di un binomio per il quale vale lo sviluppo di Newton:
  (Z+1)^p = Z^p
  + (p!/((p-1)!1!))Z^(p-1)
  + (p!/((p-2)!2!))Z^(p-2)
  + … + (p!/(1!(p-1)!))Z + 1
- I termini del tipo p!/(p-k!)k! rappresentano i coefficienti del triangolo di Tartaglia.

Dimostrazione - 4

- Tutti questi sono termini interi e possono essere riscritti nella forma **p(p-1)!/((p-k)!k!)**
- Dato che **p** è un numero primo, **(p-1)!/((p-k)!k!)** dev’essere intero.
  - Se non fosse intero ci si troverebbe nella situazione assurda che un numero frazionale moltiplicato per un numero primo darebbe come risultato un intero, e ciò vorrebbe dire che il numero primo è multiplo del denominatore della funzione, il che non è possibile dato che un numero intero è multiplo solo di 1 e di se stesso.

Dimostrazione - 5

- Da quanto detto il coefficiente è multiplo di **p**.
- Gli unici termini non multipli di **p** sono **Z^p** e **1,** quindi una volta estratto il modulo rimangono solo questi due termini, ovvero: **((Z+1)^p) mod p=(Z^p+1) mod p**
- Ma dall'ipotesi risulta che **Z^p=Z mod p**
- Sostituendo **Z mod p** a **Z^p** della formula precedente si ottiene: **(Z+1)^p=(Z mod p +1) mod p**

Dimostrazione - 6

- Riprendendo l'espressione **(Z+1)^p=(Z mod p)+1) mod p**
- vediamo che può essere ridotta a **(Z+1)^p=(Z+1) mod p**
- Il che dimostra che se il teorema era valido per **Z** allora è valido per **Z+1** e, dato che è valido per 1 allora è valido per tutti i numeri naturali.

Seconda parte - 1

- Asserto: se **z** non è multiplo di **p** vale anche la proprietà **(z^(p-1))%p=1**.
- Ovviamente se **z** è multiplo di **p** allora anche **z^(p-1)** lo è e quindi **(z^(p-1))%p** diventa uguale a zero.
- Se, però, **z** non è multiplo di **p** allora si può scrivere **(z^p)=z mod p**
- ovvero **z*z*z*z*z.....*z=z+xp**
  dove **x** è il risultato della divisione intera **(z^p)/p**

Seconda parte - 2

- dividiamo i due termini per **z** otteniamo **(z*z*z*z*...*z)/z=1+xp/z**
- se **z** non è multiplo di **p** allora **x/z** deve essere intero. Se non lo fosse, allora **xp/z** non sarebbe intero e quindi si otterrebbe un numero intero (z^(p-1)) sommando 1 ad un numero non intero (**xp/z**).
- Quindi, chiamando **y** il numero intero **x/z**, si potrebbe scrivere: **z^(p-1) = 1 + yp**
- Quindi **(z^(p-1))%p = 1**

Cifra secondo RSA

- Vengono scelti due grandi numeri primi, **p** e **q**.
- Viene calcolato il modulo **n** come prodotto tra i numeri **p** e **q**.
- Viene scelto un grande numero intero **d** che deve risultare primo relativamente al prodotto **(p-1)(q-1)**.
- Viene calcolato un numero **e** tale che **1<= e <= (p-1) (q-1)** e tale che **de%((p-1)(q-1))=1**.
- Il messaggio da cifrare è un numero **M** minore di **n**.
- L'informazione cifrata **C** viene ottenuta con il seguente calcolo: **C=(M^e)%n**.
- L'informazione decifrata **D** viene ottenuta con il seguente calcolo: **D=(C^d)%n**.

Dimostrazione - 1

- Per eseguire la dimostrazione si consideri l'operazione di cifratura **C = (M^e)%n**
- Questa operazione può essere anche scritta come **C = M^e - sn**
  - dove **s** è il risultato della divisione intera **(M^e)/s**.
- Sostituendola nella formula di decifratura si ottiene:
  D = (C^d)%n = ((M^e -sn)^d)%n

Dimostrazione - 2

- Sviluppando l'elevazione a potenza **d** del binomio **(M^e-sn)** si nota che tutti i termini dello sviluppo che contengono **sn** non contribuiscono al modulo e che quindi si può semplificare il polinomio come
  D = ((M^e)^d)%n = (M^(ed))%n
- Ricordando che **ed%((p-1)(q-1))=1** si può scrivere che **ed = 1+k(p-1)(q-1)**
  dove **k** è un opportuno numero intero.

Dimostrazione - 3

- Sostituendo questo nella formula precedente si può scrivere:
  M^(ed)%n = M^(1+(k(p-1)(q-1)))%n = (M M^(p-1)^(q-1)^k) %n
- questo si può anche scrivere come **M^(ed)%n=(M M^k^(q-1)^(p-1))%p*q**
- Per il primo teorema di Fermat, dimostrato nella sezione precedente, si può scrivere:
- **M^k^(q-1)^(p+1) = 1+zp** dato che **p** è primo (z è un intero opportuno)

Dimostrazione - 4

- Se **M** non è multiplo di **p** allora:
  M^(ed)%pq = M (1+zp)%pq
- Analogamente, operando per q
  M^(ed)%pq = M (1+yq)%pq
- Ciò significa che Mzp%pq ed Myq%pq debbono essere eguali ovvero che
- **Mzp=Myq+wpq**
  dove w è un opportuno numero intero

Dimostrazione - 5

- **Mzp=Myq+wpq**
- Essendo che M,z,y,p e q sono tutti interi e che q e p sono mutuamente primi deve risultare che y e z siano multipli, rispettivamente, di p e q.
- Ciò detto, i termini zp%pq e yq%pq si riducono a 0 e quindi le formule **M (1+zp)%pq** e **M (1+yq)%pq** si riducono a **M%pq** ovvero a **M**

Utilizzo

- Occorre quindi ricordarsi alcune cose:
  - Si possono cifrare numeri grandi sino a n (p*q)
    - quindi niente flussi
  - L'algoritmo contiene calcoli *costosi* in termini di elaborazione e quindi è pesante

Sicurezza

- Tutta basata sulla difficoltà di
  - generazione dei numeri primi
  - fattorizzazione
    - ho **n** ed **e** e dovrei reperire **d**
      - per farlo dovrei scomporre **n** in **p** e **q** in modo da ricavare **p-1 * q-1** e da questo ricavare **d**

Insicurezza - 1

- Non viene dai PC ne dai cluster
  - la potenza di calcolo nonostante la legge di Moore non aumenta così velocemente
  - per risistemarsi basta aumentare la lunghezza delle chiavi

Insicurezza - 2

- Viene dai matematici
  - L'affondo finale verrà da un algoritmo inaspettato
  - I numeri primi sono comunque ancora *poco noti*
  - La matematica usata è vecchia di centinaia di anni.

E' davvero così difficile ?

- Assolutamente no
  - se usate un PC Linux una coppia di chiavi RSA si ottiene con il difficilissimo comando
    ssh-keygen -t rsa -b 2048
  - -t rsa significa che usate l'algoritmo Rivest-Shamir-Aldemann
  - -b 2048 significa che generate una chiave lunga 2048 bit

E' davvero così difficile ?

- Esiste un tool che vi permette di fare degli iteressanti giochetti
  - openssl

Ma, in soldoni ?

- In soldoni voi avete due chiavi
  - Una è vostra, riservata, l'altra è pubblica
  - Se Alice vuole mandare un messaggio segreto a Bob allora lo cifra usando la chiave pubblica di Bob, che, essendo pubblica, conosce.
  - Bob, e solo lui, lo decifra usando la sua chiave privata
  - Eve, che non ha la chiave privata di Bob, si attacca al tram.

Davvero Eve non può ?

- E invece si, può
  - Basta che mandi in giro una sua chiave pubblica dicendo che è di Bob
  - Alice la usa per cifrare per Bob
  - Bob non riesce a leggerla
  - Eve, che ha la corrispondente chiave privata, la legge
- Per essere sicura, la chiave dev'essere **davvero** pubblica

GRAZIE PER
LA PAZIENZA!