RSA

- L'algoritmo RSA (dal nome degli inventori Rivest, Shamir e Adleman) è il più famoso algoritmo di crittografia a chiave pubblica
- Inventato nel 1977, poco dopo l'algoritmo di Diffie-Hellman
- Due componenti principali
  - Algoritmo di generazione delle chiavi
  - Algoritmo crittografico vero e proprio

RSA – generazione delle chiavi

- Scegliere due numeri primi *p* e *q*
- Calcolare $n = pq$
- Scegliere *e*, coprimo e più piccolo di $(p-1)(q-1)$
- Calcolare *d* tale che $de \equiv 1 \mod (p-1)(q-1)$

- La coppia (n, e) è la chiave pubblica
- La coppia (n, d) è la chiave privata

- Non è possibile risalire facilmente dalla chiave pubblica a quella privata (e viceversa), in quanto servirebbe conoscere il numero $(p-1)(q-1)$, e questo implica fattorizzare *n* nei suoi fattori *p* e *q* (problema difficile)

Funzione di Eulero

$\Phi(n) = (p-1) \times (q-1)$.

In matematica, la **funzione ϕ di Eulero** o semplicemente **funzione di Eulero** o **toziente**, è una **funzione** definita, per ogni intero positivo **n**, come il numero degli interi **compresi** tra **1** e **n** che sono **coprimi** con **n**. Ex: **$\Phi(8) = 4$ {1,3,5,7}**.

Teorema di Eulero: Se **a** è un numero **coprimo** con **n**, allora:

$$a^{\Phi(n)} \equiv 1 \mod n$$

$d = e^{-1} \mod \Phi(n)$ inverso moltiplicativo di **e** modulo **$\Phi(n)$.**

Numeri coprimi

- Cosa significa 'coprimo'?
- *a* è coprimo di *b* se il massimo comune divisore tra *a* e *b* è 1
- Ad es. 7 e 15 sono coprimi, mentre 8 e 10 no (hanno in comune il divisore 2)

- Nota: se *a* è primo, allora è coprimo di qualsiasi numero che non sia diviso da *a*
- Ad es. 7 è coprimo di tutti i numeri che non sono multipli di 7

RSA – esempio di generazione chiavi

- Siano p=3, q=11
- n=pq=33, (p-1)(q-1)=20
- Scegliamo e = 7 (7<20, 7 coprimo di 20)
- d = 3, infatti 3*7 = 21 ≡ 1 mod 20

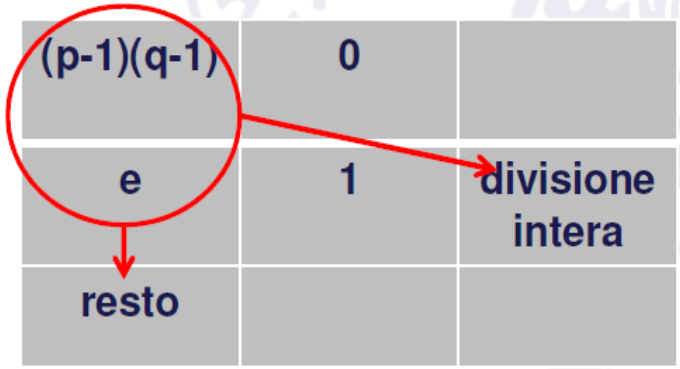
- La chiave pubblica è (33, 7)
- La chiave privata è (33, 3)

Come calcolare d?

- Metodo di Euclide esteso
- Si inizia con questa tabella:

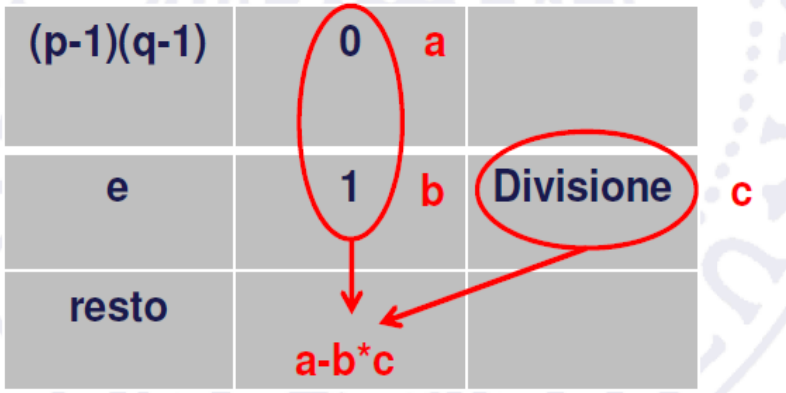
(p-1)(q-1)	0	
e	1	

- Si calcolano il risultato intero e il resto della divisione dei due numeri nella prima colonna e li si salvano nella tabella:



Come calcolare d? / 2

- Nella seconda colonna, terza riga si scrive il valore della seconda colonna, prima riga meno quello della seconda colonna, seconda riga moltiplicato per il risultato intero della divisione appena calcolato.



*Errore frequente: calcolare (a – b)*c anziché a – b*c*

Come calcolare d? / 3

- Il procedimento si ripete, aggiungendo nuove righe alla tabella e calcolandone i valori usando le due righe precedenti. Ci si ferma quando nella prima colonna compare un 1.
- Il risultato nella seconda colonna è il valore d cercato (se negativo, sommare (p-1)(q-1))

- Esempio: trovare d tale che

  $d * 7 \equiv 1 \bmod 20$

Metodo di Euclide esteso

20	0	
7	1	

Metodo di Euclide esteso

20	0	
7	1	2
6		

2 = risultato intero della divisione 20 / 7

6 = resto della divisione 20 / 7

Metodo di Euclide esteso

20	0	
7	1	2
6	-2	

$$0 - 1 * 2 = -2$$

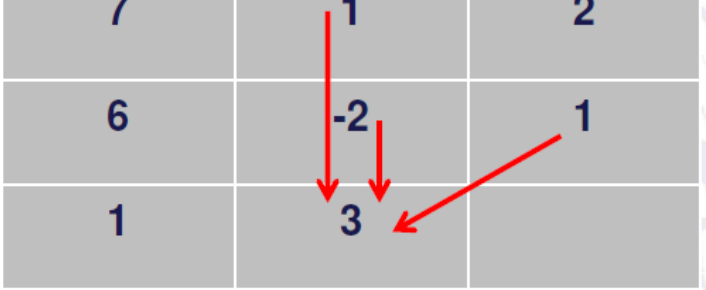
Metodo di Euclide esteso

20	0	
7	1	2
6	-2	1
1		

7/6 = 1 col resto di 1

Metodo di Euclide esteso

20	0	
7	1	2
6	-2	1
1	3	



1 - (-2*1) = 3

d = 3

Infatti $3 * 7 \equiv 1 \bmod 20$

Metodo di Euclide esteso / 2

- Altro esempio. Trovare d tale che
  d * 23 ≡ 1 mod 120

120	0	
23	1	5
5	-5	4
3	21	1
2	-26	1
1	47	

Metodo di Euclide esteso / 3

- Altro esempio. Trovare d tale che

  d * 7 ≡ 1 mod 60

60	0	
7	1	8
4	-8	1
3	9	1
1	-17	

d = -17 + 60 = 43 (se il numero è negativo, si somma il modulo)

(un altro metodo semplice per calcolare d…)

- de mod (p-1)(q-1) = 1
- de = k(p-1)(q-1) + 1
- d = (k(p-1)(q-1) + 1) / e

Provo k=1, 2, 3 …

Finché non trovo un valore INTERO per d

Esercizio

- p=7, q=13, e=…
- Calcolare la chiave pubblica (n,e) e privata (n,d)

n = p x q = 91
Scelgo **e = 11** (11<72 e coprimo con 72)
d = k x 72 + 1 / 11 per k=1,2,3,…….
k=1 => 72+1 / 11 **non INTERO**
k=2 => 144+1 / 11 **non INTERO**
……
……
k=9 => 648+1 / 11 = 649 / 11 = **59** **INTERO**

Chiave pubblica = (91,11)
Chiave privata = (91,59)

[esercizio]

- p=7, q=13, e=11
- n = p*q = 91
- (p-1)*(q-1) = 72
- d*11 mod 72 = 1
- d = -13 + 72 = 59

72	0	
11	1	6
6	-6	1
5	7	1
1	-13	

- 59*11 mod 72 = 1
- Pubblica: (91,11) privata: (91, 59)

RSA – cifratura e decifratura

- Dato un messaggio *m* (0 < m < n)
- Cifratura: calcolare $c = m^e \bmod n$
- Decifratura: calcolare $m = c^d \bmod n$

(n,d) (n,e)
rispettivamente chiave privata e pubblica del destinatario (Bob)

(nota: si basa sull'ipotesi che l'esponenziazione modulare sia un problema difficilmente invertibile – ipotesi RSA).

RSA – Esempio di cifratura e decifratura

- Chiave pubblica: (33, 7) chiave privata: (33, 3)
- m = 15

- Cifratura: $c = m^e \bmod n = 15^7 \bmod 33 = 27$
- Decifratura: $m = c^d \bmod n = 27^3 \bmod 33 = 15$

Si considera il **testo in chiaro** come una **sequenza di bit** e si divide in **blocchi** di **k** bit dove **$2^k<n$**.
Quindi k=5 ($2^5<33$)
m=01111=15 => c=27=11011

Dimostrazione del funzionamento di RSA

- **Teorema del toziente di Eulero**

Se *m* è coprimo di *pq* allora…

$$m^{(p-1)(q-1)} \equiv 1 \bmod pq$$

Dimostrazione del funzionamento di RSA

- $c^d \bmod n = [m^e \bmod n]^d \bmod n = m^{de} \bmod n$
- Siccome $de \equiv 1 \bmod (p-1)(q-1)$ allora
- $m^{de} \bmod n = m^{k(p-1)(q-1)+1} \bmod n$

$$= m \cdot [m^{(p-1)(q-1)}]^k \bmod n$$

Per il teorema del toziente(*), ricordando che $n=pq$

$$= m \cdot 1^k \bmod n$$

$$= m \bmod n$$

- Quindi $c^d \bmod n = m \bmod n = m$ (perché $m<n$)

(*) se m è coprimo di n. In realtà il procedimento vale per tutti gli m, ma la dimostrazione in questo caso va oltre gli scopi di questo corso

Applicazioni reali di RSA

- Per garantire la non invertibilità delle funzioni utilizzate, è importante usare numeri sufficientemente grandi. Nelle applicazioni attuali, solitamente n è un numero di almeno 1024 bit (poco più di 300 cifre decimali).

$2^{1024} \approx (2^{10})^{102} \approx (10^3)^{102} \approx 10^{306}$ **(oltre 300 cifre decimali)**

Rivest, Shamir e Adleman

- Appena scoperto l'RSA, i tre crittografi lanciarono una sfida alla comunità mondiale dei matematici, **cifrando un testo** e fornendo come **chiave pubblica** di **129** cifre:
- **N** = 114 381 625 757 888 867 669 235 779 976 146 612 010 218 296 721 242 362 562 561 842 935 706 935 245 733 897 830 597 123 563 958 705 058 989 075 147 599 290 026 879 543 541
- Chiedevano di fattorizzare il numero nei **due numeri primi** **P** e **Q** che lo componevano per riuscire a decrittare il messaggio, e il premio era di 100$.
- Tale sfida fu risolta 17 anni dopo il 26 Aprile 1994, da un gruppo di 600 volontari di tutto il mondo. I due fattori erano:
- **P** = 3 490 529 510 847 650 949 147 849 619 903 898 133 417 764 638 493 387 843 990 820 577
- **Q** = 32 769 132 993 266 709 549 961 988 190 834 461 413 177 642 967 992 942 539 798 288 533

Conclusioni RSA

- Il **testo in chiaro** viene visto come una **stringa di bit** e viene diviso in **blocchi** costituiti da **k bit**, dove **k** è il **più grande intero** che soddisfa la disequazione $2^k < n$ ($k=\log_2 n$).
- **Il codice RSA viene considerato sicuro perché non è ancora stato trovato il modo per fattorizzare numeri primi molto grandi, che nel nostro caso significa riuscire a trovare p e q conoscendo n.**
- Nel corso degli anni l'algoritmo RSA ha più volte dimostrato la sua **robustezza**: in un esperimento del 1994, coordinato da Arjen Lenstra dei laboratori Bellcore, per "rompere" una chiave RSA di 129 cifre, svelando il meccanismo con cui quella chiave generava messaggi crittografati, sono stati necessari 8 mesi di lavoro coordinato effettuato da 600 gruppi di ricerca sparsi in 25 paesi, che hanno messo a disposizione 1600 macchine da calcolo, facendole lavorare in parallelo collegate tra loro attraverso Internet.
- Data la mole delle risorse necessarie per rompere la barriera di sicurezza dell'algoritmo **RSA**, è chiaro come un **attacco alla privacy** di un sistema a **doppia chiave non sia praticamente realizzabile**. Inoltre, nell'esperimento era stata utilizzata una chiave di 129 cifre mentre i programmi di crittografia attualmente a disposizione prevedono chiavi private con una "**robustezza**" che raggiunge e **supera i 2048 bit**, risultando quindi praticamente **inattaccabili**, visto anche che l'ordine di grandezza dei tempi necessari alla rottura di chiavi di questo tipo è esponenziale e passa in fretta da qualche giorno a qualche **centinai di anni**.
- **$2^{2048} \approx (2^{10})^{204} \approx (10^3)^{204} \approx 10^{612}$ (oltre 600 cifre decimali)**