

## una possibile funzione unidirezionale

- moltiplicare due interi a  $n$  bit è **facile** (in  $\mathcal{O}(n^2)$  con l'algoritmo usuale)
- trovare un primo a  $n$  bit, e verificare che è primo, è **facile**
- fattorizzare un numero a  $n$  bit è **difficile** ( $2^{cn^{1/3}}$ )
- si può costruire un crittosistema a chiave pubblica basato su questa osservazione?

## crittosistema RSA

- Sia  $N = pq$ ,  $p, q$  primi. Sia  $\mathcal{P} = \mathcal{C} = \mathbb{Z}_N$ .
- Lo spazio delle chiavi è

$$\mathcal{K} = \{(N, p, q, d, e) \mid de \equiv 1 \pmod{\phi(N)}\}.$$

- Se  $k = (N, p, q, d, e)$  è una chiave, poniamo
- $e_k(x) = x^e \pmod{N}$
- $N$  e  $e$  sono la **chiave pubblica**
- $d_k(y) = y^d \pmod{N}$
- $p, q, d$  sono la **chiave privata**

## Problemi facili

- ① dato un intero  $N$ , vedere se è primo (**test di primalità**)
- ② dati  $c$  e  $M$ , trovare  $(c, M)$ ; se è 1, calcolare l'inverso di  $c$  modulo  $M$  (**alg. di Euclide – polinomiale**)
- ③ calcolare la f.ne  $x \rightarrow x^e \pmod{N}$  (**square and multiply**)

## teorema dei numeri primi

- $\pi(x)$  = numero di primi  $p$  con  $p \leq x$
- esempio:  $\pi(30) = 4 + 4 + 2 = 10$
- **teorema dei numeri primi**: asintoticamente,  $\pi(x) \approx \frac{x}{\log(x)}$
- congetturato da Legendre, Gauss fine 700 – dimostrato da Hadamard, de la Vallée Poussin fine 800
- dato un intero positivo  $N$ , la probabilità che un numero  $< N$  scelto a caso sia primo è  $\pi(N)/N$ , per  $N$  molto grande  $\approx 1/\log(N)$
- esempio: la probabilità che un numero casuale con al più 100 cifre decimali sia primo è  $\approx 1/\log(10^{100}) \approx 1/230$
- **se sappiamo controllare se un intero è primo, trovare primi grandi è facile**

## test di primalità

- dato un intero  $N$  a  $n$  bit, come scoprire se è un primo?
- posso provare a fattorizzarlo per **divisioni successive**: devo fare al più  $\sqrt{N}$  divisioni, quindi è in  $\mathcal{O}(2^{n/2})$  – scopro se è primo, e trovo anche i fattori
- i test di primalità scoprono se  $N$  è primo **senza dire niente dei suoi fattori**
- fino al 2002 si conoscevano solo test di primalità polinomiali **probabilistici**, non **deterministici**
- nel 2002, M. Agrawal insieme a due suoi dottorandi, Kayal e Saxena, hanno trovato un algoritmo **polinomiale** per determinare la primalità
- inizialmente, in  $\mathcal{O}(n^{12})$  – nel 2005, Lenstra e Pomerance lo portano a  $\mathcal{O}(n^6)$
- **primes** è in **P**

## test deterministici e test probabilistici

- un test **deterministico** dà una risposta certa:  $N$  è primo o non lo è
- un test **probabilistico**  $\mathcal{T}$  consiste in una successione di test  $\{\mathcal{T}_m\}_{m \in \mathbb{N}}$  e una successione che va a zero  $\{\epsilon_m\}_{m \in \mathbb{N}}$  tale che,
  - se  $N$  **non** passa il test  $\mathcal{T}_m$  allora **non è primo**,
  - la probabilità che  $N$  superi i test  $\mathcal{T}_1, \dots, \mathcal{T}_m$  e non sia primo è minore di  $\epsilon_m$
- i test di primalità probabilistici più usati sono quello di **Solovay-Strassen** (1977) e quello di **Miller-Rabin** (1980)
- il test di Miller-Rabin deriva da un test deterministico (dovuto a Miller) – che però assume l'ipotesi di Riemann generalizzata

- per mostrare che un numero  $N$  non è primo si mostra che non si comporta come un primo
- si “cercano prove” del fatto che  $N$  non si comporta come un primo
- senza cercare i suoi fattori

## PT di Fermat e test di Fermat

- il PTdF dice che, se  $p$  è un primo e  $1 \leq a \leq p - 1$ , allora  $a^{p-1} \equiv 1 \pmod{p}$
- vogliamo scoprire se  $N$  è primo – se trovo  $a \leq N - 1$  e  $a^{N-1} \not\equiv 1 \pmod{N}$  sappiamo per certo che  $N$  non è primo – senza sapere niente sui suoi fattori
- questo è il **test di Fermat**
  - prendo  $a < N$ , calcolo  $(a, N)$ ; se  $\neq 1$ ,  $N$  non è primo
  - se  $(a, N) = 1$ , calcolo  $a^{N-1}$  – se  $\not\equiv 1 \pmod{N}$  allora  $N$  non è primo
- per esempio,  $2^{322} \equiv 157 \pmod{323}$ ; quindi 323 non è primo ( $323 = 17 \cdot 19$ )

## pseudoprimi

- se però  $a^{N-1} \equiv 1 \pmod{N}$  – non posso concludere che  $N$  è primo
- per esempio  $2^{340} \equiv 1 \pmod{341}$ , ma  $341 = 11 \cdot 31$
- si dice che  $N$  è uno **pseudoprimo** in base  $a$  se  $N$  non è primo ma  $a^N \equiv a \pmod{N}$ . 341 è uno pseudoprimo in base 2.
- possiamo provare diversi valori per  $a$ : per esempio  $3^{341} \not\equiv 3 \pmod{341}$  – quindi 341 non passa il test!

## numeri di Carmichael

- non basta: esistono numeri  $N$  che sono **pseudoprimi** in base  $a$  **per ogni possibile base  $a$**  dove  $1 < a < N$  e  $(a, N) = 1$
- un tale  $N$  si dice **numero di Carmichael**
- per esempio, si può vedere che 561 è un ndC (è il più piccolo)
- il test di Fermat non va bene
- dà comunque un'idea di come può funzionare un test di primalità

## Problemi difficili

- ① dato un intero  $N$ , fattorizzarlo
  - ② dato un intero  $N$ , calcolare  $\phi(N)$
  - ③ dati  $N$  e  $e$ , trovare  $d$  tale che  $(x^e)^d = x \pmod{N}$
- Se Eve riesce a fattorizzare  $N = pq$ , ottiene le info private di Bob: può calcolare sia  $\phi(N)$  sia  $d = e^{-1} \pmod{\phi(N)}$
  - dunque violare l'RSA non può essere più difficile che fattorizzare

## equivalenza fra i problemi difficili

- violare l'RSA non può essere più difficile che fattorizzare
- conoscere la fattorizzazione  $\iff$  calcolare  $\phi(N)$ :
  - $\implies$  ovvio
  - $\impliedby$  conoscere  $N (= pq)$  e  $\phi(N) (= (p-1)(q-1) = N - (p+q) + 1)$  vuol dire conoscere la somma  $(N - \phi(N) + 1)$  e il prodotto  $(N)$  di  $p, q$  – quindi conoscere  $p$  e  $q$
- i problemi 1 e 2 sono equivalenti: passo dall'uno all'altro in tempo polinomiale

## l'esponente di decifratura

- supponiamo che Eve sappia risolvere il pb. 3: dati  $N$  e  $e$ , riesce a trovare  $d$  tale che  $(x^e)^d = x \pmod{N}$
- allora può fattorizzare  $N$ 
  - c'è un **algoritmo polinomiale** che fattorizza  $N$  oppure fallisce con probabilità minore di  $1/2$  – ripetendolo più volte, la probabilità che fallisca diventa molto bassa
- questo fatto non ha solo un interesse teorico
- se sospettiamo che Eve ha trovato  $d$ , **non basta** cambiare l'esponente  $e$  – bisogna scegliere anche un nuovo  $N$

## modulo comune

- inoltre, si può pensare a una versione dell'RSA con una “trusted authority” che distribuisce le chiavi ai vari utenti
- in cui, per ogni utente, la chiave pubblica sia  $(N_u, e_u)$  e quella privata sia soltanto  $(N_u, d_u)$
- dal momento che si può decifrare anche senza conoscere la fattorizzazione di  $N$
- in questa situazione, **non** si può assegnare lo **stesso**  $N$  a diversi utenti
- per quanto visto ora, questo consentirebbe a ogni utente di violare i messaggi di ogni altro utente

## RSA e fattorizzazione

- risolvere uno dei tre “problemi difficili” è computazionalmente equivalente a fattorizzare  $N$
- questo **non basta** a dire che per violare bisogna fattorizzare  $N$
- si potrebbe riuscire a decrittare **senza calcolare** l'esponente  $d$

l'equivalenza si ha con il

**RSA problem:** dati  $e$  e  $N$ , calcolare le radici  $e$ -sime modulo  $N$

non si sa se l'RSA problem è equivalente alla fattorizzazione

## RSA problem e fattorizzazione

**RSA problem:** dati  $e$  e  $N$ , calcolare le radici  $e$ -sime modulo  $N$

- se si sa fattorizzare  $N$ , si può risolvere l'**RSA problem**
- c'è un algoritmo polinomiale che calcola radici  $e$ -sime modulo  $p$  se  $(e, p - 1) = 1$
- si può mostrare (teorema cinese dei resti) che calcolare la radice  $e$ -sima di  $x \bmod p$  e  $\bmod q \Rightarrow$  calcolare la radice  $e$ -sima di  $x \bmod N = pq$
- per provare l'equivalenza basta provare una riduzione del tipo
  - algoritmo efficiente per il calcolo delle radici  $e$ -sime mod  $N \Rightarrow$  algoritmo efficiente per fattorizzare  $N$
- trovare questa riduzione è probabilmente il più importante problema aperto della PKC
- una tale riduzione esiste per  $e = 2$  - radici quadrate. Questo si usa nel CS di Rabin