

8/11/2021

lezione di crittografia
ore 16

RSA



RSA

CREAZIONE delle chiavi

Bob (DESTINATARIO)

- sceglie p e q , numeri primi molti grandi

t.c.

$$n = p \times q \text{ abbia almeno}$$

- 2048 cifre binarie (protezione dati
fino al 2030)

- 3072 ' (prot. oltre i 2030)

$$\boxed{k(\text{pub}) = \langle n, e \rangle}$$

$$k(\text{priv}) = \langle d \rangle$$

tempo polinomiale
(Miller-Rabin)

- calcola

$$n = p \times q$$

$$\phi(n) = (p-1)(q-1)$$

tempo polinomiale

- sceglie $e < \phi(n)$, $\text{MCD}(e, \phi(n)) = 1$

- calcola $d = e^{-1} \bmod \phi(n)$

(d esiste ed è unico perché $\text{MCD}(e, \phi(n)) = 1$)

tempo polinomiale
(alg. Euclideo Esteso)

Cifratura e Decifrazione

m : messaggio

sequenze binarie trattate come un numero (int)

$$m < n$$

(altrimenti m e $m \bmod n$ producono lo stesso critogramma)

$$m \geq n,$$

m si divide in blocchi di

$$b = \lfloor \log_2 n \rfloor \text{ bit}$$

In pratica si fissa un limite b comune.

$$\Rightarrow m < 2^b < n$$

CIFRATURA

$$C = m^e \bmod n \rightarrow$$

tempo polinomiale
con algoritmo di
esp. veloce

DECIFRAZIONE

$$m = C^d \bmod n$$

ESEMPIO

$$p=5 \quad q=11$$

$$k[\text{pub}] = \langle 7, 55 \rangle$$

$$k[\text{priv}] = \langle 23 \rangle$$

$$n = p \cdot q = 55$$

$$\varphi(n) = (p-1) \cdot (q-1) = 40$$

$$e=7 \quad \text{MCD}(7, 40)=1$$

$$d = 7^{-1} \pmod{40}$$

$$EE(7, 40) \rightarrow \langle 1, \cancel{-17}, \dots \rangle$$

$$EE(40, 7) \rightarrow \langle 1, 3, -2 - 3 \lfloor \frac{40}{7} \rfloor \rangle = \langle 1, 3, -2 - 3 \cdot 5 \rangle$$

$$EE(7, 5) \rightarrow \langle 1, -2, 1 + 2 \lfloor \frac{7}{5} \rfloor \rangle = \langle 1, -2, 3 \rangle$$

$$EE(\cancel{5}, 2) \rightarrow \langle \cancel{1}, 1, 0 - 1 \cdot \lfloor \frac{5}{2} \rfloor \rangle = \langle 1, 1, -2 \rangle$$

$$EE(2, 1) \rightarrow \langle \cancel{1}, 0, 1 - 0 \cdot \lfloor \frac{2}{1} \rfloor \rangle = \langle 1, 0, 1 \rangle$$

$$EE(1, 0) \rightarrow \langle \cancel{1}, 1, \cancel{0} \rangle$$

$$d = -17 \pmod{40} = 23 \pmod{40} = \boxed{23}$$

Chunro

$$EE(@, 0) \rightarrow \langle 0, 1, 0 \rangle$$

$$\langle d, \times \cancel{y} \rangle = \langle d, \cancel{y} \rangle \times^{-} y [\frac{\alpha}{\beta}]$$

$$(23 \cdot 7) \pmod{40}$$

$$\cancel{161}$$

$$M = 28$$

$$C = 28^7 \bmod 55$$

$$\textcircled{m} = C^{23} \bmod 55$$

~~•~~ CORRETTEZZA RSA

$$\mathcal{O}(\underbrace{\mathcal{G}(m, k[\text{pub}])}_{c}, k[\text{prv}]) = m$$

$$(m^e \bmod n)^d \bmod n = m$$

$\underbrace{}_c$

$$\boxed{m^{ed} \bmod n = m}$$

de dimostrar

1) $m \in n$ co-friuli $\text{MCD}(m, n) = 1$

$$(1) \quad m^{\phi(n)} \equiv 1 \pmod{n} \quad \text{th. Euler}$$

$$(2) \quad e \cdot d = r\phi(n) + 1$$

def. di inverso

$$ed \equiv 1 \pmod{\phi(n)}$$

$$m \stackrel{ed}{\mod} n = m^{1+r\phi(n)} \mod n = m \cdot m^{r\phi(n)} \mod n =$$

(2)

$$= m \cdot (m^{\phi(n)})^r \mod n = m (1)^r \mod n$$

(1)

$$= m \mod n = m$$

\cdot

m < n



2) $\text{MCD}(n, m) \neq 1$

$$p \mid m \quad \text{oppure} \quad q \mid m$$

(ma non entrambi;
perché solitamente $M \geq n$)

$$p \mid m \quad e \quad q \nmid m$$

$$p/m \Rightarrow$$

$$m \equiv 0 \pmod{p}$$

$$q \nmid m$$

Few

$$m^k \equiv 0 \pmod{p}$$

$$m^k - m \equiv 0 \pmod{p}$$

$$\underline{k = e \cdot d}$$

$$\Rightarrow m^{ed} - m \equiv 0 \pmod{p}$$

$$q \nmid m$$

$$m^{ed} \pmod{q} = m^{1 + r\phi(n)} \pmod{q} = m \cdot m^{r(p-1)(q-1)} \pmod{q} =$$

$$= m \cdot (m^{(q-1)})^{r(p-1)} \pmod{q} = m \cdot (m^{\phi(q)})^{r(p-1)} \pmod{q}$$

$$= m \cdot (1)^{r(p-1)} \pmod{q} = m \pmod{q} \quad \text{MCD}(m, q) = 1$$

$$\Leftrightarrow m^{ed} \pmod{q} = m \pmod{q}$$
$$\rightarrow m^{ed} - m \equiv 0 \pmod{q}$$

$$m^{ed} - m \equiv 0 \pmod{p}$$
$$m^{ed} - m \equiv 0 \pmod{q}$$

$\downarrow \Rightarrow$

$$m^{ed} - m \equiv 0 \pmod{p \times q}$$

$$m^{ed} \pmod{n} = m \pmod{n} = \underset{\substack{\uparrow \\ m < n}}{m}$$

✓

Sicurezza

Legata alla difficoltà di fattorizzare un numero arbitrario molto grande

fattorizzare \Rightarrow forzare RSA ✓

fattorizzare \Leftarrow forzare RSA ?

calcolo della radice in modulo

$$m = \sqrt[e]{c} \bmod n$$

$$\textcircled{C} \leftarrow m^e \bmod n$$

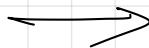
difficile almeno quanto fattorizzare (n è composto)

Calcolare $\phi(n)$ direttamente da n è equivalente a fattorizzare n

Ricavare d direttamente da (n, e) sembra costoso quanto fattorizzare n

FATT. N

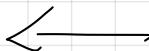
CALCOLO di
 $\phi(n)$ ~~di~~



se fattorizzato \Rightarrow know p e q \Rightarrow know

p e q

$$\phi(n) = (p-1)(q-1)$$



dati n, $\phi(n)$, calcolo p e q in tempo polinomiale

$$\phi(n) = (p-1)(q-1) = \overbrace{p \cdot q}^n - (p+q) + 1$$

$$\phi(n) = n - (p+q) + 1$$

$$(p+q) = n - \phi(n) + 1$$

$$x_1 = n - \phi(n) + 1$$

$$n = p+q$$

$$\rightarrow (p+q)^2 = (p-q)^2 + 4n$$

$$x_1^2 = x_2^2 + 4n \Rightarrow$$

$$x_2 = \sqrt{x_1^2 - 4n}$$

$$x_1 = p+q$$

$$x_2 = p-q$$

$$p = (x_1 + x_2)/2$$

$$q = (x_1 - x_2)/2$$

Come possiamo fattorizzare un intero n ?

La fattorizzazione è un problema difficile, ma non più come un tempo...

- Da un lato la potenza di calcolo aumenta, dall'altro gli algoritmi di fattorizzazione vengono raffinati
- Esistono algoritmi relativamente veloci di complessità subesponenziale
 - General Number Field Sieve (GNFS) richiede $O(2^{\sqrt{b \log b}})$ operazioni per fattorizzare un intero n , con $[\log_2 n] + 1$ bit
 - Un attacco bruteforce ne richiede $O(n)$, i.e., $O(2^{\log n})$
- Con la potenza di calcolo attuale, usando l'algoritmo GNFS è possibile fattorizzare semiprimi fino a circa 768 bit
- Per interi con una struttura particolare, esistono algoritmi di fattorizzazione particolarmente efficienti
- La fattorizzazione e il logaritmo discreto non sono problemi NP-hard, e si possono risolvere in tempo polinomiale su macchine quantistiche

RSA– scelta dei parametri

Scegliere p e q di almeno 1024 bit.

$O(2^{b/2})$

TDEA, AES (bit della chiave)	<i>RSA e DH</i> (bit del modulo)	ECC (bit dell'ordine)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

256

15360

512

RSA – vincoli su p e q

- Scegliere p e q molto grandi (di almeno 1024 bit) per resistere agli attacchi bruteforce
- Sia $p - 1$ che $q - 1$ devono contenere un fattore primo grande (altrimenti n si fattorizza velocemente)
- $\gcd(p-1, q-1)$ deve essere piccolo
conviene scegliere p e q tale che $\frac{(p-1)}{2}$ e $\frac{(q-1)}{2}$ siano co-primi
- Non riusare uno dei primi per altri moduli!

$$\begin{aligned} \xrightarrow{\quad} n_1 &= p * q_1 \\ \xrightarrow{\quad} n_2 &= p * q_2 \\ \text{allora } p &= \gcd(n_1, n_2) \end{aligned}$$

RSA – vincoli su p e q

Scegliere p e q **non troppo vicini** tra loro

$n \sim p^2 \sim q^2$ e quindi anche \sqrt{n} sarà vicino ai primi

basterà quindi un attacco bruteforce che cerca i fattori vicino alla \sqrt{n}

$$p \sim q \\ \cdot p^2 \sim q^2 \sim n$$

$$p-q \sim \log n \\ p-q \sim n^\varepsilon \quad 0 < \varepsilon < 1$$

$$\frac{p+q}{2} > \sqrt{n}$$

$$\left(\frac{p+q}{2}\right)^2 = \left(\frac{p-q}{2}\right)^2 + n$$

$$\left(\frac{p+q}{2}\right)^2 - n = \underbrace{\left(\frac{p-q}{2}\right)^2}_{> 0}$$

$$\left(\frac{p+q}{2}\right)^2 > n$$

→ si scendono gli intv. $> \sqrt{n}$ fini a buone z.t.c.

$$\underbrace{z^2 - n}_{\text{è un quadrato perfetto}} = w^2$$

è un quadrato perfetto —

$$z = \frac{p+q}{2}$$

$$w = \frac{p-q}{2}$$

$$p = z + w\sqrt{-1}$$

$$q = z - w\sqrt{-1}$$

Attacchi con esponenti bassi

- esponenti e e d bassi sono attratti perché accelerano cifratura o decifrazione
- ovviamente d dovrebbe essere scelto sufficientemente grande, per evitare attacchi bruteforce
- **ATTENZIONE**
 - se m ed e sono così piccoli che $m^e < n$, allora risulta **facile** trovare la **radice e-esima di c** , poiché $c = m^e$
non interviene la riduzione in modulo!