## 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

## scelta di $e$

- si cerca di scegliere $e$ non troppo grande e tale che nella scrittura binaria di $e$ ci siano pochi 1
- $e$ piccolo = cifratura più veloce
  - nell'alg. square and multiply bisogna calcolare pochi quadrati
- meno 1 = cifratura più veloce
  - il numero di 1 è collegato al numero di moltiplicazioni nell'alg. square and multiply
- valori usati sono $e = 3$ (inizialmente, ma dà problemi), $e = 2^{16} + 1 = 65537$
- per il valore $2^{16} + 1$ sono necessarie 17 moltiplicazioni - per un $e < \phi(N)$ casuale, la stima è di circa 1000 moltiplicazioni
- non vuol dire che anche $d$ sarà piccolo

## RSA – broadcast attack

- se l'esponente $e$ è piccolo e usato da molti utenti ho un possibile attacco (broadcast attack)
- se Alice manda lo stesso messaggio $m$ a $B_1$, $B_2$, $B_3$ che usano lo stesso esponente di cifratura $e = 3$ e chiavi $N_1$, $N_2$, $N_3$ prime fra loro
- Eve conosce $m^3$ mod $N_1$, mod $N_2$, mod $N_3$
- si ha $m < N_1, N_2, N_3$ – allora $m^3 < N_1 \cdot N_2 \cdot N_3$
- applicando il Teorema cinese del resto
- Eve riesce a trovare $x \equiv m^3$ mod $N_1 \cdot N_2 \cdot N_3$
- quindi $x = m^3$ come interi, non modulo qualcosa!
- può fare la normale radice cubica e decifrare

## Wiener low exponent attack

- $d$ non deve essere piccolo
- se $\sqrt{6}d < N^{1/4}$ e $q < p < 2q$ c'è un attacco dovuto a Wiener
- permette di ricavare rapidamente $d$
- se $N$ ha 1024 bit, $d$ deve avere almeno 256 bit
- la decifratura è computazionalmente pesante - questo è un problema per esempio per le smartcard
- si può usare il teorema cinese del resto per velocizzare la decifratura

## RSA – side channel attack

- sono attacchi all'implementazione dell'algoritmo, non all'algoritmo
- uno è il timing attack
- analogia: uno scassinatore che cerca di capire la combinazione di una cassaforte dal tempo impiegato a "girare le rotelle"
- per l'RSA, si cerca di capire qual è la scrittura binaria dell'esponente di decifratura $d$ osservando il tempo impiegato a decifrare
- nell'algoritmo square and multiply, si fa una moltiplicazione solo quando nella scrittura binaria di $d$ c'è un 1
- posso risalire al numero di 1 in $d$
- e alla loro posizione
- un altro è il power monitoring attack

## Crittosistema di Rabin

- Sia $N = pq$, $p$, $q$ primi con $p \equiv q \equiv 3 \pmod 4$. Sia $\mathcal{P} = \mathcal{C} = \mathbb{Z}_N$.
- Lo spazio delle chiavi è

$$\mathcal{K} = \{(N, p, q) \,|\, N = pq\}.$$

- Se $k = (N, p, q)$ è una chiave, poniamo
- $e_k(x) = x^2 \pmod N$
- $N$ è la chiave pubblica
- $d_k(y) = \sqrt{y} \pmod N$
- $p$, $q$, sono la chiave privata

# Ambiguità nel crittosistema di Rabin

- Come capire quale delle quattro radici quadrate è quella giusta?
- si aggiunge ridondanza
- se $k+1$ è la lunghezza del modulo $N$
- i messaggi avranno al più $k$ bit
- trasmetto $\lfloor \frac{2}{3}k \rfloor$ bit di testo in chiaro e ripeto le ultime cifre
- la decodifica giusta è quella che mostra questo pattern

## un esempio

- Ex. Scelti $p = 271, q = 311, N = 84281$, lunghezza $= 16 + 1$
- Bob pre-calcola $1 = -70 \cdot 271 + 61 \cdot 311$; $s = -70, t = 61$.
- dei 16 bit di un messaggio, 10 saranno il testo in chiaro e 6 saranno ridondanza
- Alice deve cifrare 1101111001; ripete le ultime 6 cifre e ha 1101111001111001 che dà $x = 40569$
- cifratura: $40569^2 \pmod{84281} \equiv 4393 = y$
- per decifrare si calcola $y_p = y^{(p+1)/4} = 4393^{68} \equiv 81 \pmod{271}$ e $y_q = y^{(q+1)/4} = 4393^{78} \equiv 139 \pmod{311}$
- le radici sono $\pm spy_q \pm tqy_p = \mp 70 \cdot 271 \cdot 139 \pm 61 \cdot 311 \cdot 81$
- otteniamo $79755 = (1001101111000101 1)_2$, $4526 = (1000110101110)_2, 40569$, $43712 = (1010101011000000)_2$

## come utilizzare un PKCS

- affinché un CS sia sicuro, la cifratura deve essere randomizzata
- in un cifrario simmetrico (per esempio AES) si sceglia un'opportuna modalità di funzionamento (per esempio CBC)
- e in un PKCS?
- sia $f : A \to B$ una funzione trapdoor one-way (come quella RSA)
- $f$ si utilizza per generare la chiave

## come utilizzare un PKCS

- per cifrare si usa $f : A \to B$ funzione trapdoor one-way
- insieme a un CS simmetrico in modalità sicura (CBC-AES)
- Alice sceglie in modo casuale $a \in A$ e calcola $f(a) = b$ (facile - pubblico)
- Alice cifra il messaggio $x$ con la chiave $a$ e ottiene il CT $y$ (usando AES: $y = e_a(x)$)
- trasmette a Bob la coppia $(b, y)$

## come utilizzare un PKCS

- Bob riceve la coppia $(b, y)$
- per decifrare deve conoscere $a = f^{-1}(b)$
- è il solo a possedere l'informazione segreta che gli permette di invertire la $f$
- può ottenere $a$ e decifrare con AES: $x = d_a(y)$
- la randomizzazione "viene" dall'AES
- per maggiore sicurezza, in genere la chiave non è $a$ ma è $h(a)$ dove $h$ è una fuzione hash