Crittografia RSA

Crittografia asimmetrica

- Il metodo RSA è un metodo di crittografia moderna che utilizza una chiave asimmetrica : la chiave utilizzata per crittare il testo è diversa da quella utilizzata per decrittarlo.
- Proprio per questo motivo bisogna distinguere le chiavi : la chiave pubblica, utilizzata per crittare il messaggio e la chiave privata utilizzata per decrittarlo.
- La chiave pubblica viene diffusa dal mittente assieme al messaggio crittato.
- La chiave privata è in possesso del destinatario che la tiene segreta e la usa per decrittare il messaggio ricevuto.

Metodo RSA (1)

- Per utilizzare questo metodo di crittografia si decidono 2 numeri primi n e q molto grandi (attualmente vengono utilizzati numeri con circa 300 cifre);
- Si calcola il prodotto tra i due numeri : $n=p*q$;
- Si calcola la chiave pubblica cercando un numero e coprimo e più piccolo di $(p-1)(q-1)$;
- A questo punto si può calcolare anche la chiave privata : $e*d \equiv 1 \mod((p-1)(q-1))$;

Metodo RSA (2)

- Per crittare il messaggio si calcola $c=m^e \mod(n)$ (m è il messaggio da crittare, e è la chiave pubblica);
- Per decrittare il messaggio si calcola $c^d \mod(n)$ (c è il messaggio crittato, d è la chiave privata).

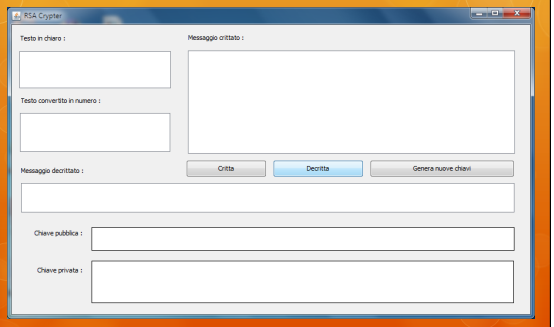
Implementazione RSA in Java

- Il package java.math contiene la classe BigInteger, i cui oggetti rappresentano numeri di lunghezza arbitraria;
- Con questi oggetti i normali operatori matematici **NON** si possono usare;
- Al loro posto vengono utilizzati vari metodi : add, substract, multiply, equals…

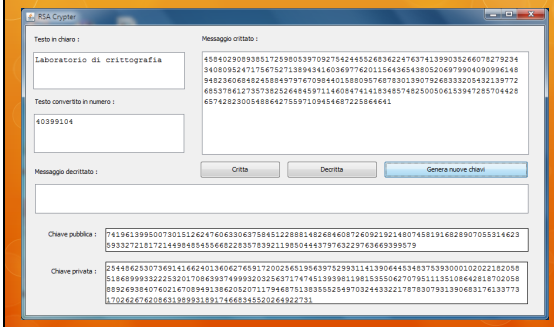
RSACrypter

- RSACrypter è un programma sviluppato in Java che utilizza il metodo RSA per crittare una stringa inserita dall'utente;
- Prima di tutto la stringa viene trasformata in numero, questo numero costituisce il messaggio che viene crittato. Vengono generate le chiavi e viene mostrato il messaggio crittato;
- Cliccando su Decritta il messaggio viene decrittato utilizzando la chiave privata e il risultato viene mostrato in un altro campo di testo.

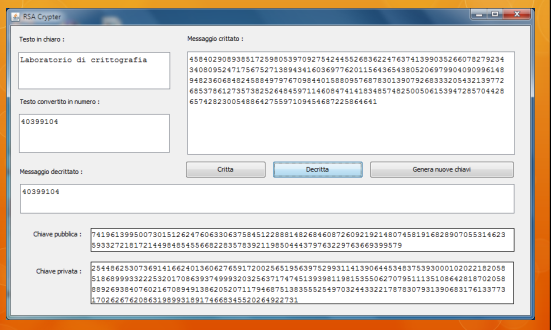
Interfaccia del programma (1)



Interfaccia del programma (2)



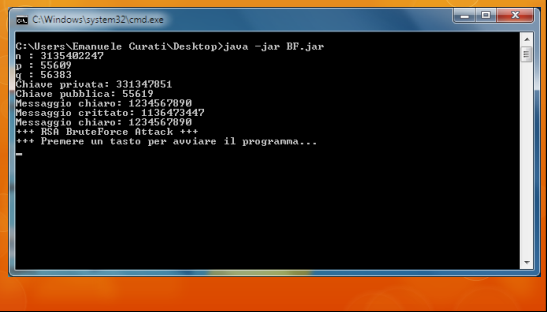
Interfaccia del programma (3)



Bruteforce

- Per forzare questo algoritmo è necessario entrare in possesso dei numeri primi p e q;
- Conoscendo n il programma per trovare p e q si scrive con pochissime righe di codice;
- L'elemento che rende sicuro questo algoritmo è sicuramente la complessità computazionale necessaria per risalire alla chiave privata;
- Con 2 numeri p e q a 16 bit il bruteforce impiega pochi secondi… con numeri a 32 bit dopo 10 ore il programma è ancora in esecuzione;
- È facile intuire come con numeri a 1024 bit sia impensabile utilizzare un bruteforce.

RSABruteForce (1)



RSABruteForce (2)