**UNIVERSITETI I PRISHTINËS “HASAN PRISHTINA”**

**FAKULTETI I SHKENCAVE MATEMATIKO-NATYRORE**

**DEPARTAMENTI I MATEMATIKËS**

**PROGRAMI: SHKENCA KOMPJUTERIKE**



**PUNIM SEMINARIK**

Lënda:Siguria e të dhënave

TEMA: “Simplified SHA”-rasti 2

**Punuan:**Auron Ismajli

Florida Kurtaj

Laureta Durguti

# Kerkesa e Detyres

Bob bën gjenerimin e çelësave publik dhe privat për RSA. Ai zgjedh *p* dhe *q* por nuk është i kujdesshëm kur zgjedh, sepse . Pas gjenerimit të çelësave, ai dërgon çiftin (*n,e)=(797306344204135429053419.920419823)* te Alice përmes internetit.Zbatoni në Java sulmin për të gjetur *p*, *q* dhe *d*? (Përdorni klasën *BigInteger* për numrat dhe metodën *SqrtAndRemanider* për rrënjët katrore me mbetje).

## Hyrje

* Ky program Java llogarit faktorët kryesorë të një numri të madh të caktuar '*n*' dhe kryen aritmetikë modulare.
* Ai përdor klasën *BigInteger* nga *paketa java.math* për të trajtuar numra të plotë me madhësi arbitrare.
* Sulmi qe ceket ne kerkesen e detyrës,është një metodë që përdor faktin që diferenca midis `*p`* dhe `*q`* është e vogël () për të gjetur një nga faktorët kryesorë (*p*) të modulit `*n`*. Ky sulm përfiton nga fakti se nëse diferenca midis `*p`* dhe `*q`* është e vogël, ne mund ta përafrojmë `*p*` duke marrë rrënjën katrore të `*n*` dhe duke e rregulluar atë në mënyrë iterative derisa të plotësojë kushtin `*n = p \* q`*.
* **Algoritmi:**
* Inicializoni objektet *BigInteger* për numrin e madh '*n*' dhe eksponentin '*e*'.
* Llogarit rrënjën katrore të '*n*' dhe shtoni 1 për të marrë '*a*'.
* Llogarit '' dhe zbritni '*n*' për të marrë '*r*'.
* Llogarit rrënjën katrore të '*r*' për të marrë '*x*'.
* Llogarit dy numra, '*p*' dhe '*q*', duke zbritur dhe shtuar '*x*' në '*a*' përkatësisht.
* Këta ka të ngjarë të jenë faktorët kryesorë të '*n*'.
* Llogaritni funksionin totient të Euler '' duke përdorur '*p*' dhe '*q*'.
* Llogaritni inversin modular të '*e*' në lidhje me ''' ' për të marrë '*d*'.
* Shtypni vlerat e '*p*', '*q*' dhe '*d*'.

## RSA

Në kontekstin e kriptimit **RSA**, variablat "*p*", "*q*" dhe "*d*" kanë kuptimet e mëposhtme:

- `*p*` dhe `*q*` janë numra të thjeshtë të njohur si faktorët kryesorë të modulit `*n*`. Moduli `*n*` llogaritet si `*n = p \* q*`. Këta faktorë kryesorë mbahen sekret dhe përdoren në gjenerimin e çelësave publik dhe privat. Njohja e '*p*' dhe '*q*' lejon dikë që të faktorizojë '*n*' dhe potencialisht të thyejë enkriptimin *RSA*.

- `*d*` përfaqëson çelësin privat në enkriptimin *RSA*.Është inversi shumëzues i modulit të eksponentit publik `*e*``*(p-1)(q-1)*`.Çelësi privat `*d*` mbahet sekret dhe përdoret për deshifrim. Përdoret për të llogaritur mesazhin origjinal duke e ngritur tekstin shifror në fuqinë e modulit `*d*` `*n*`.

Në kodin e dhënë, sulmi synon të gjejë vlerat e `*p*`, `*q*` dhe `*d*` duke shfrytëzuar një dobësi në përzgjedhjen e faktorit kryesor. Duke gjetur një nga faktorët kryesorë `*p*` duke përdorur sulmin Ackermann, ne mund të llogarisim faktorin tjetër kryesor `*q*` duke pjesëtuar `*n*` me `*p*`. Pastaj, çelësi privat `*d*` llogaritet duke përdorur `*p*`, `*q*` dhe eksponentin publik `*e*`.

## Zgjidhja e problemit

import java.math.BigInteger;

public class Detyra\_3\_2 {

    public static void main(String[] args)

    {

        BigInteger n=new BigInteger("797306344204135429053419");

        BigInteger e=new BigInteger("920419823");

        BigInteger a=n.sqrt();

        BigInteger one=new BigInteger("1");

        a=a.add(one);

        BigInteger a2=a.multiply(a);

        BigInteger r=a2.subtract(n);

        BigInteger x=r.sqrt();

        BigInteger p=a.subtract(x);

        BigInteger q=a.add(x);

        System.out.print("p="+p+"   q="+q);

        BigInteger phi=(p.subtract(one)).multiply(q.subtract(one));

        BigInteger d=e.modInverse(phi);

        System.out.println("     d="+d);

          }

}