MINISTERUL EDUCAȚIEI AL REPUBLICII MOLDOVA

UNIVERSITATEA TEHNICĂ A MOLDOVEI

FACULTATEA CALCULATOARE, INFORMATICĂ ȘI MICROELECTRONICĂ

DEPARTAMENULT INGINERIE SOFTWARE ȘI AUTOMATICĂ

LUCRARE DE LABORATOR NR.1

Disciplina : *Securitatea Informațională*

Tema : *Algoritmul DSA*

A elaborat : st.gr.TI-171 f/r , Florea Cristina

A verificat : lect.univ. Poștaru Andrei

2021

Scopul lucrării.

Studierea , analiza metodologiei și implementarea algoritmului DSA

Introducere

Semnăturile digitale pot fi folosite pentru a asigura autenticitatea și integritatea datelor . O semnătură digitală se referă la o valoare calculată cu un algoritm criptografic și adăugată la date în sensul că oricine primește aceste date poate folosi semnătura pentru a verifica integritatea și originea lor. Semnătura digitală se realizează folosind un sistem criptografic cu chei public și o funcție de dispersie (hash).

În august 1991 , Institutul Național de Standarde și Tehnologie al SUA (NIST) a propus un algoritm de semnătură digitală (DSA). În scurt timp, schema DSA a fost adoptată ca standard sub numele DSS(Digital Signature Standard) și este prima schemă de semnătură digitală recunoscută de un guvern. Standardul de semnături digitale este o modificare a schemei de semnături ElGamal.

DSA **(Digital Signature Algorithm**) se bazează pe problema logaritmului discret și se înrudește cu schemele de semnături propuse de Schnorr și ElGamal. Algoritmul necesită o funcție hash h :{0,1} * ® Zq, pentru un întreg q.

O funcție hash (în sensul nerestricționat) este o funcție h care are cel puțin următoarele 2 proprietăți :

1. Compresie – h asociată unui input x (de lungime arbitrară) , un output h(x) de lungime fixă n.
2. Ușor de calculat – având h și input-ul x, h(x) este ușor de calculat.

Descrierea DSA

Deoarece această schemă de semnare se bazează pe problema logaritmului discret , numărul prim p trebuie ales suficient de mare – 512 biți sau chiar 1024 biți. DSA produse semnături de 320 biți.

Se utilizează un număr prim de 512 biți și un factor prim q de 160 biți al lui *p -l* . Calculele se vor realiza în subgrupul Zq al lui Z*p , utilizând un element a Z*p de ordin q.

Algoritmul de creare a cheilor este următorul :

Algoritm 1. Semnătura DSA

Crearea cheilor : Entitatea A își creează o cheie publică și o cheie privată corespunzătoare.

1. Se generează un număr prim q astfel încât 2159 < *q* < 2160
2. Se alege un număr întreg *t,* 0 £ a £ 8 și selectează un număr prim p, unde 2511+64t < *p* < 2512+64t , cu proprietatea ca q divide *p -l*

3. Se selectează un element a de ordin q din Z*p;
   a. Se selectează un element g Î Z*p, ord(g)=p-1, și calculează a = g(p-1)/q mod p;
   b. Daca a = 1 atunci se repeta pasul 3. 1;
4. Se alege un întreg a astfel încât 1 £ a £ q-1;
5. Se calculează y=aa mod p;
6. Cheia publică a lui A este (p,q,a,y); Cheia privată este a ;

Algoritmul următor descrie procesul de generare și verificare a semnăturii.

Algoritmul 2.

Generarea și verificarea semnăturii . Entitatea A semnează un mesaj binar m de lungime arbitrară. O altă entitate B va folosi cheia publică a lui A pentru a verifica semnătura.

1. Generarea semnăturii. A va parcurge pașii :
   a. Se alege aleator un întreg secret k, 0 < k< q;
   b. Se calculează r = (ak mod p) mod q;
   c. Se calculează k – 1 mod q;
   d. Se calculează s = k – 1 {h(m)+ar} mod q;
   e. Semnătura lui A pentru mesajul m este perechea(r,s);
2. Verificarea semnăturii . Pentru a verifica semnătura lui A, (r,s) , B va parcurge pașii :
   a. Se obține cheia public a lui A, (p,q,a,y) ;
   b. Se verifică dacă 0 < r< q și 0 < s< q ; în caz contrar, respinge semnătura;
   c. Se calculează w= s – 1 mod q și h(m);
   d. Se calculează u1 = w*h(m) mod q și u2 = rw mod q;
   e. Se calculează v = (au1yu2 mod p) mod q;
   f. Se acceptă semnătura doar dacă v=r;

Corectitudinea semnăturii

Daca (r, s) este o semnătură legitima a entității A asupra mesajului m, atunci h(m) º -ar+ks (mod q) este adevărată. Înmulțind ambele părți cu w, obținem w ·h(m) + arw º k (mod q). Dar w ·h(m) = u1, rw=u2. Înlocuind, obținem u1+au2 º k (mod q). Ridicam la a ambele părți si obținem (au1yu2 mod p) mod q = (ak mod p) mod q.

Prin urmare v=r.

Exemplu (Semnătura DSA)

Crearea cheilor:

Entitatea A alege p = 1295556481 și q = 145896 astfel încât q divide p-1; aici (p-1)/q = 8880. A alege g = 1124266021 si calculează a = g8880 mod p = 625833668.

Din moment ce a ¹ 1, îl vom păstra. În continuare, A alege a = 137623 și calculează y=aa mod p = 873234058. Cheia publică a lui A este (p = 1295556481, q = 145896, a = 625833668, y=873234058). Cheia privată a lui A este a=137623.

Generarea Semnăturii:

Pentru mesajul m, A calculează r=30462 și s=47587, unde h(m) = 569083831021078013872036425494296265110068199691. Semnătura lui A este (r=30462, 47587).

Verificarea Semnăturii:

Pentru verificarea semnăturii, B calculează w=s-1 mod q = 21547, u1=w ·h(m) mod q = 8105, u2 = rw mod q = 124506. Apoi B calculează v=(au1yu2 mod p) mod q=(625833668810587323405812 4506 mod 1295556481) mod 145896=30462. Din moment ce v=r, vom accepta semnătura.

Securitatea semnăturii DSA.

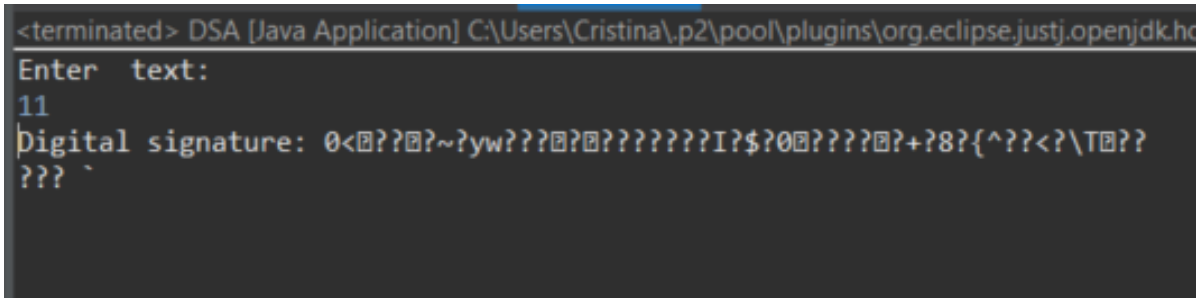
DSS a fost propusă inițial de către NIST cu lungimea cheii de 512 biți . După multe polemici asupra faptului că nu este destul de sigură ,în special pe termen lung, NIST a revizuit DSS pentru a accepta chei de lungime până la 1024 biți. În prezent DSA este considerat sigur cu chei de lungime de 1024 biți.

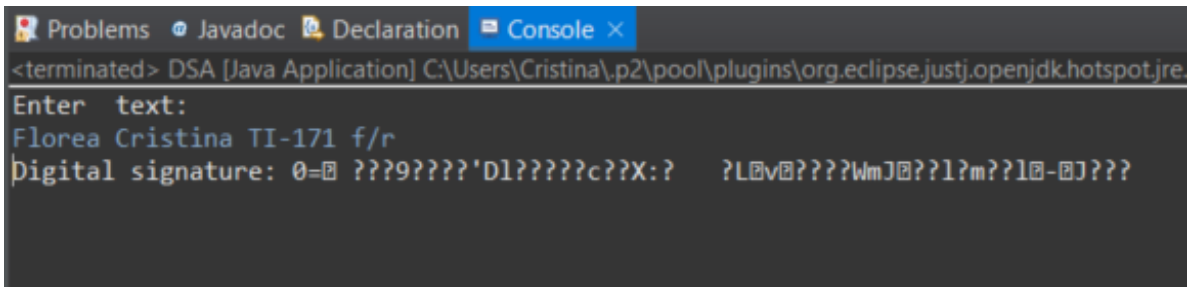
DSA se bazează pe problema logaritmului discret în anumite subgrupuri ale unui corp finit pentru un anumit număr prim p. Problema a fost propusă prima dată (în scop criptografic) în anul 1989 , de către Schnorr .Nu au fost raportate atacuri eficiente pentru această formă a problemei logaritmului discret.

Unii cercetători au avertizat de existența anumitor numere prime periculoase care ar permite ca o cheie să fie spartă cu ușurință . Aceste numere prime sunt rare și pot fi ușor evitate dacă implementarea procedurii de generare a cheii este corectă.

Implementarea

A fost implementat un algoritm care primește un mesaj de la utilizator , căruia îi atribuie o semnătura digitală . Câteva exemple de semnături digitale :





Concluzie.

DSA produce semnături mici (320 biți) și astfel , este des folosită în situațiile în care memoria disponibilă este limitată (smartcard, telefoane mobile ,etc) . La DSA generarea semnăturii este mult mai rapidă decât verificare ei .

Este avantajos ca semnarea să fie rapidă , dar în multe aplicații se semnează de obicei o singură dată , iar semnătura e verificată des.

La general , acest algoritm este considerat imposibil de spart , datorită siguranței mari care este asigurată de câteva puncte (ex. generarea aleatoare a lui p,q,a și k).

Codul sursă .

```
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.PrivateKey;
import java.security.Signature;
import java.util.Scanner;

public class DSA {
    public static void main(String[] args) throws Exception {
        //Introducing some text
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter  text: ");
        String msg = sc.nextLine();

        //Creating KeyPair generator object
        KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("DSA");

        //Initializing the key pair generator
        keyPairGen.initialize(2048);

        //Generate the pair of keys
        KeyPair pair = keyPairGen.generateKeyPair();

        //Getting the private key from the key pair
        PrivateKey privKey = pair.getPrivate();

        //Creating a Signature object
        Signature sign = Signature.getInstance("SHA256withDSA");

        //Initialize the signature
        sign.initSign(privKey);
        byte[] bytes = "msg".getBytes();

        //Adding data to the signature
        sign.update(bytes);

        //Calculating the signature
        byte[] signature = sign.sign();

        //Printing the signature
        System.out.println("Digital signature: "+new String(signature,
"UTF8"));
    }
                                        }
```