Project Cryptography

Chiara Spadafora
chiara.spadafora@unitn.it

Irene Villa
irene.villa@unitn.it

Trento, 10 May 2022

Your project is divided into two parts. These solutions must be encoded in a .mag file called **nameofyourgroup_yoursurname.mag**.

Part I

1. Write a function called *SolovayStrassen(n)* with the following specifications:

```
NAME:  SolovayStrassen(n)
INPUTS:   -n integer to be tested.
OUTPUTS:  -bool, a Boolean value.
```

   The output *bool* is true if and only if n is prime, false otherwise.

   Remark. Since this is a probabilistic algorithm, we **require** that you repeat the test for 10 times before outputting the result than n is prime.

2. Write a function called *Factorization_Lehman(N)* with the following specifications:

```
NAME:    Factorization_Lehman(N)
INPUTS:  -N is the semiprime
OUTPUT:  -p is the smallest prime factor of N
         -q is the largest prime factor of N
```

   Where a semiprime is an integer product of two primes ($N = pq$).

3. Write a function called *IndexCalculus(p,g,h)* to compute the discrete logarithm of an element $h \in \mathbb{F}_p^* = \langle g \rangle$ with the following specifications:

```
NAME:  IndexCalculus(p,g,h)
INPUTS:   -p prime;
          -g primitive root modulo p;
          -h element of Fp*.
OUTPUTS:  -x integer.
```

   Where $g^x \equiv h \mod p$.

Part II

4. The hash function used in this part will always be the identity function.

   Write a MAGMA code which implements the ECDSA algorithm:

   - *ECDSA_Signing(q, E, r, G, d, m)*,
   - *ECDSA_Verification(q, E, r, G, Q, m, R, s)*,

   with the following specifications:

```
NAME:   ECDSA_Signing(q,E,G,r,d,m)
INPUTS: -q is the number of elements of the finite
         field over which is defined the curve
        -E is the elliptic curve
        -G is the base point of E
        -r is the order of the point G
        -d is Alice's private key
        -m is the message to be signed
OUTPUT: -(R,s) is the digital signature

NAME:   ECDSA_Verification(q,E,G,r,Q,m,R,s)
INPUTS: -q is the number of elements of the finite
         field over which is defined the curve
        -E is the elliptic curve
        -G is the base point of E
        -r is the order of the point G
        -Q is Alice's public key
        -m is the message to be signed
        -R is the first part of the digital signature
        -s is the second part of the digital signature
OUTPUT: -bool is a boolean that says if the
         sign is valid
```

   Remark. In the previous functions q, r, d and m are all integers (in particular q is a power prime and r is a prime); E is an elliptic curve defined over $\mathbb{F}_q$; G, Q and S are all points in $\mathbb{E}(\mathbb{F}_q)$; s is an element in $\mathbb{F}_q$; *bool* is a boolean.

   Example. For example, you can check the following code:

```
q:=17389;
E:=EllipticCurve([GF(q)!231, 473]);
r:=1321;
G:=E![11259,11278];
m:=644;
d:=542;
Q:=d*G;

R,s:=ECDSA_Signing(q,E,G,r,d,m);
ECDSA_Verification(q,E,G,r,Q,m,R,s);
```

5. Write a function *ECDLP_PohligHellman(q,E,P,r,Q)* that breaks the Elliptic Curve Discrete Logarithm Problem (ECDLP) with the following specifications:

```
NAME:    ECDLP_PohligHellman(q,E,P,r,Q)
INPUTS:  -q is the number of elements of the finite
         field over which is defined the curve
         -E is the elliptic curve
         -P is the base point of the curve
         -r is the order of the point P
         -Q is a multiple of P
OUTPUT:  -d is the factor of Q with respect to P
```

   Remark. In the previous functions q, r and d are integers: in particular q defines the finite field $\mathbb{F}_q$ and $1 < d < r$; E is an elliptic curve; P and Q are points over the elliptic curve E such that $Q = d \cdot P$. You will find the description of the algorithms in the attached files.

All the code must be **completely written in MAGMA language**, without calling any external program.
The project will be tested and evaluated during the lecture on **24/05/2022**. If the program fails to work correctly, even on one test vector (or if it does not load, or if it is composed by more than one file), and this problem is not fixed before the end of the lecture, then all the team members will fail and they will have to attend again the whole lab session in Spring 2023. During the lecture, the speed of your algorithm will be evaluated as well.

Participants of the team presenting the faster algorithms will receive one extra point.