Homework 3 - RSA Cryptography and Elliptic Curves

Cryptography and Security 2019

- You are free to use any programming language you want, although SAGE is recommended.
- Put all your answers **and only your answers** in the provided SCIPER-answers.txt file. This means you need to provide us with all `Q` values specified in the questions below. You can download your **personal** files from the following link: `https://lasec.epfl.ch/courses/cs19/hw3/index.php`
- You will find an example parameter and answer file on the moodle. You can use this parameters' file to test your code and also ensure that the types of `Q` values you provided match what is expected. For instance, the variable `Q1_p` should be an integer, whereas `Q2_r` is a list of integers. **Please do not put any comment or strange character or any new line** in the .txt file.
- We also ask you to submit your **source code**. This file can of course be of any readable format and we encourage you to comment your code. Notebook files are allowed, but we prefer if you export your code as a text file with a sage/python script.
- The plaintexts of most of the exercises contain some random words. Don't be offended by them and Google them at your own risk. Note that they might be really strange.
- If you worked with some other people, please list all the names in your answer file. We remind you that you have to submit your **own source code** and **solution**.
- We might announce some typos/corrections in this homework on Moodle in the "news" forum. Everybody is subscribed to it and does receive an email as well. If you decided to ignore Moodle emails we recommend that you check the forum regularly.
- The homework is due on Moodle on **Thursday the 7th of November** at 22h00.

Exercise 1 Factoring by orders of elements

Suppose that N is a *strong* RSA number (`Q1_N` in parameter file), i.e. multiplication of two distinct large primes p and q such that $\frac{p-1}{2}$ and $\frac{q-1}{2}$ are also primes.

Let x be an element in $\mathbb{Z}_N^*$ and define $x_p = x \bmod p$ and $x_q = x \bmod q$. Let k_p be the order of x_p in $\mathbb{Z}_p^*$, k_q be the order of x_q in $\mathbb{Z}_q^*$. In your parameter file, you are given a pair (x, k) (variables `Q1_x` and `Q1_k`) where x is a uniformly sampled element from $\mathbb{Z}_N^*$ and k is either k_p or k_q.

Your goal is to recover the smaller factor p of N (Beware: you will lose points if you return the larger one). Provide your answer as `Q1_p` in your parameter file.

Exercise 2 7 deadly residues

Inspired by the Legendre symbol, our apprentice extends the idea into 7-th residues.

Initially, her definition applies strictly to primes p such that $p \equiv 1 \pmod 7$. Given such prime p, let g be the smallest generator of $\mathbb{Z}_p^*$. For $x \in \mathbb{Z}_p^*$, she defines the symbol $(\cdot)_p$ such that $(x)_p$ takes the value from $\mathbb{Z}_7$ that satisfies:

$$g^{(x)_p} \cdot z^7 \equiv x \pmod p \text{ for some } z \in \mathbb{Z}_p^*.$$

Later, she generalizes the idea to a particular set of composite numbers n that are multiplication of distinct primes modulo 7, i.e. $n = p_1 \cdot p_2 \cdot \cdots \cdot p_\ell$ where $p_i \equiv 1 \bmod 7$ for all $i \in [1, \ell]$. For $x \in \mathbb{Z}_n^*$, she then goes on to define:

$$(x)_n = ((x)_{p_1} + (x)_{p_1} + \cdots + (x)_{p_\ell}) \bmod 7$$

With respect to such composite number n, she says that $x \in \mathbb{Z}_n^*$ is a

- *good residue* if there exists $z \in \mathbb{Z}_n^*$ such that $z^7 \equiv x \pmod n$,
- *fake residue* if the previous condition fails, but $(x)_n = 0$,
- *non residue* if both above conditions fail.

You are given a composite number n (`Q2_n`) that satisfies the above criteria, along with its factors $p_1, p_2 \ldots p_\ell$ as a list (`Q2_factors`). Moreover, a list of elements (`Q2_elements`) $x_0, \ldots, x_\ell \in \mathbb{Z}_n^*$ are given for you to decide whether each element is a *good residue* (encode as 0), *fake residue* (encode as 1) or *non residue* (encode as 2). Provide your answers in `Q_2r` as a list of 0, 1, 2 values as integers denoting respectively the residue status of the given elements. **Hint:** Primes p_i are chosen small enough so that you can completely factorize $p_i - 1$ values.

Exercise 3 Thunderstruck!

Angry about his previous failure, Batman decides to forget about ElGamal cryptosystem forever and use RSA instead. What he does not take into consideration is thunder!

Being the smartest superhero ever, Batman selects two RSA primes p, q, each of bit length 1024, and a fairly large exponent s as the secret key, and uses the plain RSA encryption. He also implements the RSA decryption using Chinese Remainder Theorem to enhance the performance[1]. As he does not want to receive spam fan messages, he does not share his public key exponent either.

What he does not take into consideration is that there are a lot of thunderstorms in Gotham city. Each time a thunderstorm happens, the input voltage of all electronic devices in Gotham city spikes, causing some malfunction. Bat-computers are not immune against this spike either. Each time a lightning strikes, the input voltage of the processors spike, causing some computational error. Fortunately this voltage spike only affects the processor for a short period of time.

Being well aware of this phenomenon, Batman's archenemy, Joker has found Batman's public key (N, e), but as he does not want you to interfere with his business he is only telling

[1] You can check how this can be done by looking at slide #271.

you what the value N is, and is asking you young cryptographers to betray your favorite caped superhero.

In your parameters file, you can find the array `Q3_transcript` which includes several ciphertext, plaintext tuples respectively, i.e. the first element of the tuple is the ciphertext. Some of the decryptions have not been done correctly, because of the voltage spike. You can also find the modulo `Q3_N` in your parameter file. Your goal is to factorize the large number N having this information, and write down the **larger** prime divisor of N in your solution file as `Q3_prime`.

Hint: The most time consuming part of the decryption is computing m^d modulo each prime, so if a computational error occurs, it is probably during this computation.

Exercise 4 Diffie-Hellman Key Exchange Over $\mathbb{Z}_p^*$

In your parameter file, you will find a prime number p, three integers g, X, Y where g is a generator of $\mathbb{Z}_p^*$, $X = g^x \bmod p$ and $Y = g^y \bmod p$ for $x, y \in \{1, \ldots, p-1\}$.These variables are `Q4_p`, `Q4_g`, `Q4_X`, `Q4_Y` respectively. Compute $K = g^{xy} \bmod p$ and write it down in your answer file under `Q4_K`.

Hint: Why is the Diffie-Hellman key exchange insecure over a group of composite order?

Exercise 5 The Adventures of the Crypto-Apprentice: Return Of Vernam Cipher

After his failure with the Vernam cipher, the apprentice found that it was not a good idea to use UTF-16 for ASCII characters with small period of the key stream. After the lecture on the elliptic curves, the apprentice got a brilliant idea of the key generation of the Vernam cipher. His idea was to use a random point $P = (P_x, P_y)$ of an elliptic curve E as a shared secret between the sender and the receiver and a seed of the key sequence. Let $K = K_0 K_1 \cdots$ be a key sequence. Then, $K_i = \mathsf{x}([2^i]P) \bmod 2$ ($K_i = 1$ if $[2^i]P$ is the point at infinity $\mathcal{O}$) where $\mathsf{x}(P) = P_x$ and $[2^i]P$ is a scalar multiplication between an integer 2^i and a point P. Since a point addition on the elliptic curve requires both P_x and P_y, the apprentice believes that it is hard to guess next bit without knowing P. In order to send an elliptic curve, the apprentice decided to send p, a and b for the elliptic curve $E = \{\mathcal{O}\} \cup \{(x, y) \in K^2 \mid y^2 = x^3 + ax + b\}$ where $K = \mathbb{Z}_p$. With a message M and a random point P, the encryption works as in Figure 1.

procedure ENC(E, M, P)
  $M' \leftarrow \perp$
  for $i = 0$ **to** $|M| - 1$ **do**
    $M' \leftarrow M' \| M_{i,0} \| \cdots \| M_{i,7}$ where $M_{i,0}$ is MSB of M_i and $M_{i,7}$ is LSB of M_i
  end for
  for $i = 0$ **to** $|M'| - 1$ **do**
    $K_i \leftarrow \mathsf{x}([2^i]P) \bmod 2$
    $C_i \leftarrow K_i \oplus M'_i$
  end for
  return $C_0 \| C_1 \| \cdots \| C_{|M'|-1}, \mathsf{y}([2^{2|M'|}]P)$
end procedure

Figure 1: Exact implementation of the encryption. $|M|$ denotes the length of the message M.

In order to check if decryption is done correctly, the apprentice decided to attach y-coordinate of $[2^{2|M'|}]P$ which is never been used for the key stream, so that it should not leak any information about the key. Since you broke the apprentice's previous Vernam cipher, the apprentice asked you if you can break this again.

In your parameter file, you will find the prime number p as `Q5_p`, two integers a and b as `Q5_a`, `Q5_b` which define the elliptic curve E. You are also given the ciphertext C (as `Q5_C`), y (as `Q5_y`) where y is the y-coordinate of $[2^{2|M'|}]P$, and C is a bit string. Moreover, you have n (`Q5_n`) which is the order of the elliptic curve E. Decrypt C and write it under `Q5_pt` in your answer file as a ASCII string. (This means that you have to provide a **"meaningful" English phrase in ASCII!**)