Homework 3 - ElGamal and RSA Cryptography

Cryptography and Security 2017

- 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 Q_1, Q_2, Q_3, Q_4 and Q_5. You can download your **personal** files on `http://lasec.epfl.ch/courses/cs16/hw3/index.php`
- The answers should all be English phrases in ASCII, except for Exercise 5 where we expect you to give us an integer. **Please provide nothing else. This means, we don't want any comment and any 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, although we prefer a textile with a Sage (python) script. We encourage you to comment your code.
- 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 of 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 **Wednesday the 1st of November** at 22h00.

Exercise 1 RSA with biased plaintext distribution

In spite of his failure with the Vernam cipher, the crypto-apprentice was still motivated to play with several cryptosystems and he decided to try with the RSA cryptosystem this time.

In order to allow not only lower-case alphabets and space but also upper-case alphabets and punctuation marks, the apprentice decided to change the encoding of message. His new encoding firstly changes a string to an array of characters (assuming that each character is only one byte) and then it replaces each character to corresponding ASCII code. For example, a string "Raccoon?" would be encoded as [82, 97, 99, 99, 111, 111, 110, 63].

Since the apprentice did not know how to encrypt an array of integers within single RSA ciphertext, he decided to encrypt a message character by character. Fortunately, the apprentice realized that encrypting a list of elements of $\{0, 1, \ldots, 255\}$ is completely insecure. Indeed, given the encryption c of a an integer x, one can try to compute $c'_i = i^{e_1} \bmod n_1$ for all $i \in \{0, 1, \ldots, 255\}$ and easily recover $x = i$ by finding $c'_i = c$.

In order to overcome this problem, the apprentice decided to introduce some randomness and defined RSA$'[\chi]$. Let $m \in \{0, 1, \ldots, 255\}$ be an ASCII code of a character. Then, the encryption of m under RSA$'[\chi]$ with the public key (e_1, n_1) is defined as $(256 \cdot \lfloor r \rceil + m)^{e_1} \bmod n_1$ and the decryption of a ciphertext c under the key (d_1, n_1) is defined as $(c^{d_1} \bmod n_1) \bmod 256$ where r follows a random distribution χ.[1]

Since the apprentice believed that his cryptosystem is secure, he asked his friend to encrypt an important message using the new cryptosystem RSA$'[\chi]$ with his public key (e_1, n_1). He also said to his friend: "Use the Gaussian distribution[2] $\mathcal{N}(\lfloor n_1/512 \rfloor, \sigma_1^2)$ as the random distribution χ and set σ_1 to the month in which I was born. Our communication will be bullet-proof."

So the friend first transformed the important message Q_1 into an array of integers M_1 by computing $M_1[i] = \mathsf{ASCII}(Q_1[i])$ for $i = 0, \ldots, \mathrm{len}(Q_1) - 1$. The friend then computed a list of ciphertexts C_1, in which $C_1[i]$ is the encryption of $M_1[i]$ under RSA$'[\mathcal{N}(\lfloor n_1/512 \rfloor, \sigma_1^2)]$ with the public key (e_1, n_1).

Now, you should show his intuition was incorrect. In your parameter file, you will find the modulus n_1, the public key e_1 and the array of ciphertexts C_1. Recover the message Q_1 and write it in your answer file. (This means that you have to provide a **"meaningful" English phrase in ASCII!**)

Exercise 2 Rabin Decryption

A friend of our crypto-apprentice wanted his help to decrypt a ciphertext C_2. His friend told him that the message is encrypted by the Rabin Cryptosystem whose key generation and encryption algorithms work as follows:

- *Key Generation:* pick two random prime numbers p and q, set $N = pq$. The public key is N and the secret key is (p, q).
- *Encryption:* the plaintext $x \in \mathbb{Z}_N$ Output the ciphertext $c = \mathsf{x}^2 \bmod N$.

After reading Chapter 3 of the course, our crypto-apprentice has the confidence to help his friend. This friend gave the apprentice a ciphertext C_2 (which is an encryption of a plaintext X_2), and the corresponding public key N_2 and the secret key (p_2, q_2). The friend also provided an extra information to him which is that X_2 is an encoding of a "meaningful" English ASCII-string Q_2, such that $X_2 = \mathsf{ascii2int}(Q_2)$.

The encoding $\mathsf{ascii2int}(\cdot)$ is used to encode ASCII strings as integers. We encode a string $s = s_1 s_2 \ldots s_r$ of r ASCII characters as $\mathsf{ascii2int}(s) = \sum_{i=0}^{r-1} \mathtt{ASCII}(s_{i+1}) \cdot 2^{8 \cdot i}$, where $\mathtt{ASCII}(c)$ is the ASCII value of a character c.[3] For example, the encoding of the string "Red Fox!" would be 2411799947438744914.

Help the crypto-apprentice and his friend to decrypt the ciphertext C_2 and find the message Q_2! You will find in your parameter file the ciphertext C_2 and the private key (p_2, q_2). Decrypt C_2, find the plaintext Q_2 and write it in your parameter file. (This means that you have to provide a **"meaningful" English phrase in ASCII!**)

[1] The notation $\lfloor r \rceil$ denotes rounding the real number r to the nearest integer in the usual way.

[2] $\mathcal{N}(\mu, \sigma^2)$ is a Gaussian distribution whose mean is μ and variance is σ^2.

[3] Since ASCII characters can be represented by 8-bit binary strings, this encoding is injective and can be inverted.

Exercise 3 RSA with small exponent

When the apprentice cryptographer recovered from the character-wise-RSA fiasco, he felt like having one more attempt at using the RSA cryptosystem correctly. He generated the modulus $n_3 = p_3 \cdot q_3$ such that p_3 and q_3 are two random primes of 1029 bits each. Intending to reduce the computational complexity of the encryption, the apprentice chose $e_3 = 3$, which happened to be coprime with $\varphi(n_3)$. Then he computed $d_3 = e_3^{-1} \bmod \varphi(n_3)$, and distributed the public key (e_3, n_3) among his friends.

Alice, one of these friends, wanted to send an important message Q_3 to the apprentice. She first encoded the message Q_3 (composed of ASCII characters only) as an integer x_3 using the encoding function ascii2int from Exercise 2. Alice then encrypted x_3 with the (plain) RSA cryptosystem and the public key (e_3, n_3), obtaining a ciphertext c_3 which she sent to the apprentice. You have intercepted both the public key (e_3, n_3) and the ciphertext c_3. You shrewdly guess, that the message Q_3 should consist of no more than 86 ASCII characters. Teach the apprentice a lesson and recover the secret message!

In your parameter file, you will find the modulus n_3 and the cipher text c_3. Recover the secret message Q_3 and write it in your answer file. (This means that we expect a **"meaningful" English phrase in ASCII characters**!)

Hint: Is knowing that $x_3^{e_3}$ is smaller than $\gamma \cdot n_3$ with a $\gamma < 2^8$ useful?

Exercise 4 Iterative ElGamal

Feeling ashamed by the failures with RSA, our apprentice decided to try ElGamal which is a probabilistic encryption scheme by design. He used ElGamal in the group $\mathbf{Z}_{p_4}^*$ where p_4 is a prime number. He picked a generator g_4 of $\mathbb{Z}_{p_4}^*$ and generated the public key is y_4 following the ElGamal key generation algorithm. He understood the cryptosystem very well, but he was not sure how to generate the randomness for the encryption. He knew that he should not repeat the randomness.

To be sure on this, he picked an initial random number $r_{init} \in \mathbb{Z}_{p_4}^*$ and a small constant integer γ which is the last two decimal digits of r_{init} (e.g. if $r_{init} = 123456789$ then $\gamma = 89$). He used $r_{init}\gamma$ as a random number for the first encryption c_{41}. Then, he generated a new random value r_{new} in each encryption as $r_{new} = r_{prev}\gamma$ where r_{prev} is the random value he used while encrypting the previous message.

You obtained the first ciphertext c_{41} and the i^{th} ciphertext c_{4i}. You do not know i but you are sure that our apprentice did not encrypt many messages (at most 20) until you intercepted c_{4i} .

In your parameter file, you have the ciphertexts $c_{41} = (u_{41}, v_{41})$ and $c_{4i} = (u_{4i}, v_{4i})$, the public parameters of ElGamal p_4, g_4 and the public key y_4. The ciphertext c_{41} is an encryption of the integer m_{41}, which is an encoding of the (ASCII) string "`Hello World!`" under the function ascii2int (from Exercise 2). The ciphertext c_{4i} is an encryption of the integer m_{4i}, which is an encoding of and (ASCII) English phrase Q_4 under the function ascii2int. Recover the message Q_4 and write it in your answer file. (This means that we expect a **"meaningful" English phrase in ASCII characters**!)

Exercise 5 Quadratic Residue

In your parameter file, you will find four prime numbers $p_{51}, p_{52}, p_{53}, p_{54}$. Find the integer Q_5, which is the smallest quadratic residue modulo n_5 that is strictly greater than $B(\text{Sciper})$, where $n_5 = p_{51} \cdot p_{52} \cdot p_{53} \cdot p_{54}$, the function $B : \mathbb{Z} \to \mathbb{Z}$ is defined as $B(x) = x^{\lceil x/50000 \rceil}$ and Sciper denotes *the first six digits* of your own sciper number (we put your sciper number in your parameter file for your convenience).[4]

[4] $\lceil \cdot \rceil$ is the ceiling function that rounds a real number up to the nearest bigger-or-equal integer.