Homework 2 – RSA, DH & Elliptic Curves

Cryptography and Security 2023

- ◇ You are free to use any programming language you want, although Python/SAGE is recommended.
- ◇ Put all your answers **and only your answers** in the provided `[id]-answers.txt` file where `[id]` is the student ID.[1] This means you need to provide us with all `Q`-values specified in the questions below. Personal files are to be found on Moodle under the feedback section of Parameters HW2.
- ◇ **Please do not put any comment or strange character or any new line** in the submission file and do **NOT** rename the provided files.
- ◇ Do **NOT** modify the `SCIPER`, `id` and `seed` headers in the `[id]-answers.txt` file.
- ◇ **Submissions that do not respect the expected format may lose points.**
- ◇ 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 normal textual files containing Python/SAGE code. If an answer is incorrect, we may grant partial marks depending on the implementation.
- ◇ Be careful to always cite external code that was used in your implementation if the latter is not part of the public domain and include the corresponding license if needed. Submissions that do not meet this guideline may be flagged as plagiarism or cheating.
- ◇ Some plaintexts may contain random words. Do not be offended by them and search them online at your own risk. Note that they might be really strange.
- ◇ Please list the name of the **other** person you worked with (if any) in the designated area of the answers file.
- ◇ Corrections and revisions may be announced on Moodle in the "News" forum. By default, 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 **November 28th, 2023** at 23h59.

[1]Depending on the nature of the exercise, an example of parameters and answers will be provided on Moodle.

Exercise 1 Let's MOVe.

Crypto apprentice is back ! Since your last task, you've gained in seniority and confidence in your skills, and you have managed to convince the company to move to modern cryptography. They've heard of a new hot topic called "Elliptic Curve Cryptography" and decided to explore it in depth. They've heard that a certain problem called "Discrete Logarithm" needs to be hard on elliptic curves in order to build interesting cryptosystems. As such, they are asking you to solve a good old challenge to assess their confidence in the hardness of this problem.

Given an elliptic curves $E : y^2 = x^3 + ax + b$ defined over a field F_p where p is given under Q1_p determined by the parameters given under Q1_param (in the form [a, b]), and two points P, Q given under Q1_P and Q1_Q. Throughout this exercise, your goal is to compute n such that $Q = n \cdot P$.

Question 1.1 It's cold out there...

Depressed by the terrible weather you've been suffering through for the past few weeks, you are inspired to look at the (embedding) degree of your curve.

Definition 1 (Embedding degree)**.** Given an elliptic curve $\overset{E(\mathbb{F}_p)}{E/F_p}$ of order M, we call an integer k the *embedding degree* of E the smallest $k \geq 1$ such that

$$M | p^k - 1 \iff p^k - 1 \equiv 0 \ (M) \iff p^k \equiv 1 \ (M)$$

1. Compute the embedding degree k of the elliptic curve defined by your parameters and report it under Q1a_k. For Loop increasing k and checking $p^k \overset{?}{=} 1 \ (M)$

Food for thoughts: Can you come up with another elliptic curve of small ($k \leq 6$) embedding degree ?

Question 1.2 Warming up

You decide to raise the temperature and look at the problem in $\mathbb{F}_{p^k}$ rather than $\mathbb{F}_p$, where k is the embedding degree you just computed. You happen to stumble upon the following lemma :

Lemma 1. *Given $P, Q \in E(\mathbb{F}_p)$, let $N = ord(P)$, $gcd(N, p) = 1$ then there exists n such that $Q = nP$ if and only if $NQ = \mathcal{O}$ and $e_N(P, Q) = 1$, where e_N denotes the Weil pairing.*

Definition 2 (Weil Pairing)**.** Given 2 elliptic curves E, E' over a finite field of characteristic p, and $E[N] := \{P \in E(\bar{\mathbb{F}_p}) | N \cdot P = \mathcal{O}\}$ the Weil pairing e_N is defined as the following **bilinear**, non-degenerate map :

$$e_N : E[N] \times E[N] \to \mu_N$$
$$(S; T) \mapsto e_N(S, T)$$

where μ_N is the group of N-th rooths of unity, i.e. $\mu_N = \{x \in \mathbb{F}_{p^k} | x^N = 1\}$, where k is the embeding degree. (Note that you do not need to know *how* to compute e_N as Sage does it for you.)

1. Compute the Weil pairing between the points $S1, S2$ given under Q1b_S1, Q1b_S2 and report it under Q1b_e.

2. Use this lemma to deduce a way to lift the problem to $\mathbb{F}_{p^k}$ and then compute the discrete logarithm of Q with respect to P and report it under Q1b_n.
   You now need to work with the same elliptic curve over $\mathbb{F}_{p^k}$ and lift all your points to this curve. Then choose an additional point R in a specific way (this is for you to figure out), and use some Weil pairing computations between R, P, Q and exploit its bilinear properties to deduce the solution to your problem. MOV attack

$$Q = \underset{\substack{\uparrow \\ \log_P(Q)}}{n} \cdot P$$

Exercise 2 Product Rating Disaster

A company decided to create a product rating system and they need to compute a statistic where they need to calculate the multiplication of all the ratings for their product. They wanted the users to encrypt their ratings before sending.

- They released a procedure KeyGen and published the resulting public key pk and other public parameters *pp*. To generate the public parameters, they used a hard discrete log instance generator GroupGen.
- They instructed customers to use the $\mathsf{Enc}(\mathsf{pk}, r, pp)$ procedure to encrypt a rating r.

Both of these procedures are described below:

```
KeyGen($1^\lambda$)
1 : $(p, g) \leftarrow\$ \mathsf{GroupGen}(\lambda)$
2 : $x \leftarrow\$ \mathbf{Z}_p^*$
3 : $pp \leftarrow (p, g)$
4 : $\mathsf{pk} \leftarrow g^x$
5 : return $(x, \mathsf{pk}, pp)$

Enc($\mathsf{pk}, r, pp$)
1 : parse $pp \rightarrow (p, g)$
2 : $u \leftarrow g^r$          not random
3 : $v \leftarrow \mathsf{pk}^r \cdot r$
4 : $C \leftarrow (u, v)$
5 : return $C$
```

Because of these instructions, the product rating system ended up being broken due to the following reasons:

- They copied the ElGamal Algorithm from COM-401 Slides (no. 236) and used the variable r for randomness and the same variable r to denote a rating. Because of this confusion, the rating value r is also used in place of a random exponent as seen in line 2 and 3.
- They did not specify a range for the ratings and some of the clients ended up submitting very large rating values.

Question 2.1

You are given a public key Q2a_pk, public parameters Q2a_p and Q2a_g, a list of Q2a_N ciphertexts {Q2a_C} that corresponds to the encryption of Q2a_N ratings $\{r_i\}_{i=1\ldots\mathsf{Q2a\_N}}$. Compute the required statistic $\prod_{i=1}^{\mathsf{Q2a\_N}} r_i \mod (p)$ and report it under Q2a_rmult.

```
gen_list($n$, seed)
$x \leftarrow$ seed
$i \leftarrow 0$
while $x \neq 1$ :
    $L[i] \leftarrow x$
    $x \leftarrow x^2 \bmod n$
    $i \leftarrow i + 1$
```

Figure 1: Generator of residues.

Exercise 3 Factor Walk With Me

In the small town of Two Mountains, a serie of dreadful crimes have been committed and you've been sent to assist in the investigation.

Question 3.1

The detective in charge, Agent Copper, shows you the only clue they have found so far: weird (quadratic) residues. They are given to you in Q3a_L and Q3a_y. You notice that the 15 last bits of the values in Q3a_L[2] have been erased (i.e. set to 0). Later that night, a giant appears in your dreams and gives you the following information:

- The list of residues was generated using the function gen_list(Q3a_N, seed) (see Figure 1) for some unknown seed seed $\in \mathbb{Z}_{\text{Q3a\_N}}$. Note that the erasure of the bits occurred after.
  $[seed, seed^2, seed^4, seed^8, ...]$
- The value Q3a_y is a quadratic residue modulo Q3a_N and you should find its roots.

▷ Given Q3a_N, Q3a_L, Q3a_y, find the square roots of Q3a_y. Report them in a list **in ascending order** under Q3a_QR. E.g. Q3a_QR $= [3, 5, 7, 11]$.

Question 3.2

Well done! These were actually coordinates of the location of a victim's diary. In it, you find a list of RSA public keys Q3b_PKS that were generated by multiplying **three** primes, i.e. $n \in$ Q3b_PKS is s.t. $n = pqr$ for some primes p, q, r. You also find the description of an encryption algorithm enc(n, e, m) (see Figure 2) along with three cryptic sentences:

1. *In order to decrypt the RSA ciphertexts, the CRT acceleration trick (slide 263) is used.*
2. *Glitches are frequent in the town, they can happen in the middle of decryption.*
3. *The prime number generator has low entropy.*

The next day, you receive intelligence from the military base located nearby. That includes a RSA public key Q3b_N and a ciphertext

$$(\text{Q3b\_CT\_RSA}, \text{Q3b\_CT\_AES}, \text{Q3b\_CT\_NONCE}, \text{Q3b\_CT\_TAG}) := \text{enc}(\text{Q3b\_N}, 3, m)$$

[2] Due to an error in parameter generation, please ignore the first value in Q3a_L (the one at index 0). This has no impact on the solution.

```
enc(n, e, m)
1 :  $K_{AES} \leftarrow\!\$\ \{0,1\}^{128}$
2 :  $k \leftarrow \mathsf{int}(K_{AES})$
3 :  $\mathsf{ct_{rsa}} \leftarrow k^e \bmod n$ // the AES key is encrypted with RSA
4 :  $\mathsf{ct_{aes}}, \mathsf{nonce}, \mathsf{tag} \leftarrow \mathsf{Enc_{AES}}(K_{AES}, m)$ // $m$ is encrypted with AES
5 :  return $(\mathsf{ct_{rsa}}, \mathsf{ct_{aes}}, \mathsf{nonce}, \mathsf{tag})$
```

Figure 2: Encryption function.

for some plaintext m. In addition, they also give you a list of intercepted RSA plaintext/ciphertext pairs Q3b_PAIRS, where $(x, y) \in$ Q3b_PAIRS are such that $y := \mathsf{dec_{RSA}}(\mathsf{sk}, \mathsf{enc}_{RSA}(\mathsf{pk}, x))$, where $\mathsf{pk} := (n = \text{Q3b\_N}, e = 3)$ and sk is the corresponding secret key.

▷ Given Q3b_N, Q3b_PKS, Q3b_PAIRS, Q3b_CT_RSA, Q3b_CT_AES, Q3b_CT_NONCE, and Q3b_CT_TAG, you must recover the plaintext (a few sentences in English) and report it under Q3b_PT.

HINT: As in Exercise 3 of the previous homework, the goal is to recover K_{AES}; once you have it, recovering the plaintext is simply decrypting with AES-GCM using the following code snippet (after installing the `pycryptodome` package):

```python
from Crypto.Cipher import AES
def decrypt(Q3_ct_aes, Q3_tag, Q3_nonce, rec_k_aes):
    # rec_k_aes is assumed to be an int
    rec_k_aes_bytes = rec_k_aes.to_bytes(16, "big")

    cipher = AES.new(rec_k_aes_bytes, AES.MODE_GCM, Q3_nonce)
    Q3_pt = cipher.decrypt_and_verify(Q3_ct_aes, Q3_tag)
    return Q3_pt
```