Problem 1: Hybrid encryption – implementations

(a) Implement a hybrid encryption using ElGamal and AES. You are allowed to use ready-made ElGamal and AES.

In the contributed file hybrid.py (lecture webpage), you find a prepared template in Python that already provides function for ElGamal and AES encryption as well as some utility functions and testing code that you might need. I recommend to use that code. If you wish to use another language, you will have to find your own ElGamal and AES routines.

You should check that hybrid_decrypt(sk, hybrid_encrypt(pk, msg)) returns msg. It is OK if you only allow encrypting messages whose length is a multiple of 16 bytes (blocklength of AES).

(b) [Bonus problem] The ElGamal implementation used in hybrid.py might leak whether the message msg is a quadratic residue. Using the methods developed on the previous exercise sheet (problem “Encoding messages for ElGamal”), fix the functions elgamal_encrypt and elgamal_decrypt to avoid this leakage. (You need to make sure that elgamal_decrypt(sk, elgamal_encrypt(pk, msg)) still returns msg.)

Problem 2: Textbook RSA and hybrid encryption

A common variant of textbook RSA is the following: During key generation, the modulus N is chosen as usual. We chose e as e := 3 (instead of random). Then d is chosen with ed ≡ 1 mod \( \varphi(N) \) (as usual). The public key is \( pk = (N, e) \) and the secret key is \( sk = (N, d) \). Encryption and decryption are as usual. (We call this encryption scheme 3RSA in the following. Let \( E_{3RSA} \) denote the corresponding encryption algorithm.) Let \( E_{AES} \) denote the AES encryption algorithm.

From 3RSA we can construct a hybrid encryption scheme as follows: \( E(pk, m) := (E_{3RSA}(pk, k), E_{AES}(k, m)) \). Here \( k \) is a random AES-key (256 bits). And \( pk \) is a 4096-bit 3RSA key (i.e., |N| = 4096).

Break the hybrid encryption scheme. That is, show how to efficiently compute \( m \) given a single ciphertext \( E(pk, m) \).

Hint: For what values \( m \) is \( m^3 \mod N \) the same as \( m^3 \)? What is \( E_{3RSA}(pk; k) \)?
Problem 3: Malleability of ElGamal

Remember the auction example from the lecture: Bidder 1 produces a ciphertext $c = E(pk, bid_1)$ where $E$ is the ElGamal encryption algorithm. Given $c$, Bidder 2 can then compute $c'$ such that $c'$ decrypts to $2 \cdot bid_1 \pmod{p}$ (where $p$ is the modulus from the ElGamal public key $pk$). This allows Bidder 2 to consistently bid twice as much as Bidder 1.\footnote{As long as $bid_1 < p/2$, that is. Otherwise $2 \cdot bid_1 \pmod{p}$ will not be twice as much as $bid_1$. However, for large $p$, $bid_1 \geq p/2$ is an unrealistically high bid.}

Now refine the attack. You may assume that $bid_1$ is the amount of Cents Bidder 1 is willing to pay. And you can assume that Bidder 1 will always bid a whole number of Euros. (I.e., $bid_1$ is a multiple of 100.)

Show how Bidder 2 can consistently overbid Bidder 1 by only 1%. What happens to your attack if Bidder 1 suddenly does not bid a whole number of Euros?

**Hint:** Remember that modulo $p$, one can efficiently find inverses. For example, one can find a number $a$ such that $a \cdot 100 \equiv 1 \pmod{p}$.\footnote{As long as $bid_1 < p/2$, that is. Otherwise $2 \cdot bid_1 \pmod{p}$ will not be twice as much as $bid_1$. However, for large $p$, $bid_1 \geq p/2$ is an unrealistically high bid.}