# Security 1 - Mandatory Hand-in 1

Sam Al-Sapti (sals@itu.dk)

September 29, 2022

## Setup

I have decided to implement the assignment in Python, as it is a simple language, yet powerful enough for the purposes of the assignment. The program can be run by executing `python3 main.py` or `./main.py`.

## Part 1

I start by defining variables `g` and `p`, and then define `group` as a list from 0 to $p-1$, representing the cyclic group $(\mathbb{Z}_p^*, \cdot)$.

I then define the function `encrypt(pk, m)`, that uses the ElGamal encryption algorithm to encrypt a plaintext message `m` with a public key `pk`, and returns a tuple `(c1, c2)`. The algorithm goes as follows:

1. Select a random $r \in \mathbb{Z}_p^*$.
2. Compute $c_1 = g^r \bmod p$.
3. Compute $c_2 = pk^r \cdot m \bmod p$.
4. Return the ciphertext $(c_1, c_2)$.

With this function, Alice encrypts the message `2000` as an integer, using Bob's public key.

## Part 2

For this part, I use a brute-force attack to find Bob's private key. I iterate over `group` and compare the result of $g^{sk} \bmod p$ with Bob's public key, which is known as 2227, for each `sk` in `group`. When an `sk` is found such that $g^{sk} \bmod p = 2227$, the iteration breaks and `sk` is used as Bob's private key to decrypt Alice's ciphertext, using the function `decrypt(sk, c)`. Eve finds Bob's private key to be 66.

The function takes a private key `sk` and the ciphertext as the tuple `c = (c1, c2)`, and follows the following decryption algorithm:

1. Compute $s = {c_1}^{sk} \bmod p$.
2. Compute $m = c_2 \cdot s^{-1} \bmod p$.
3. Return $m$.

Due to type conversion issues occurring when dividing integers in Python, I have utilized Fermat's little theorem[1] to instead compute $m = c_2 \cdot s^{p-2} \bmod p$ in step 2.

Using Bob's private key that Eve has found, Alice's ciphertext is successfully decrypted, yielding the original plaintext `2000`.

# Part 3

In this part of the assignment, Mallory is assumed to know that the original plaintext reads `2000`.

The ciphertext contains $(c_1, c_2)$, where $c_2 = pk^r \cdot m \bmod p$. By choosing an appropriate modifier $d$, Mallory can successfully compute a new $c_2'$, as long as $d \cdot m < p$ holds. Since Mallory knows the plaintext to be `2000`, he can choose $d = 3$ to compute $c_2' = pk^r \cdot 3m \bmod p$ and modify the ciphertext to $(c_1, c_2')$. Since $m = 2000$ and $3m = 6000$, Bob will get the plaintext `6000` when he decrypts the modified ciphertext.

[1] https://en.wikipedia.org/wiki/Fermat%27s_little_theorem