Mandatory Assignment 2

Implementation

I have implemented a simple version of additive sharing, with the use of a TLS-protocol using OpenSSL in Python.

This project consist of two elements: a server (the hospital) and a group of clients (Alice, Bob and Charlie).

Each client has a secret number that has to be summed up with the other two clients. This protocol is described in the section *Generating 3 shares of a single number*

To ensure a secure channel, I have used a TLS-package called OpenSSL. The reason for using TLS, is to ensure confidentiality, data integrity and authentication. For this, I have generated a certificate that authenticates the users and the server (For reasons behind only generating one certificate, see *Regarding certificates*). The rest is covered by the OpenSSL-package.

Regarding certificates

For this assignment, I have only generated a single key and certificate into one `.pem` file. The reason behind this decision, was that if each parcipant (both server and clients) should have their own certificate, each certificate should be signed by a Certificate Authority. For that Certificate Authority to be trusted, it would need to be trusted by my computer. This means, that for each computer my program is run on, the Certificate Authority would need to be trusted. This process would be cumbersome to say the least. Therefore, I have only generated a single key and certificate, that all parties use.

Generating 3 shares of a single number

To split the secret number into 3 ueven parts, I have made a method called `split_in_three_uneven`. This function generates 3 secret shares. The first two shares (`number1` and `number2`) is two randomly generated number ranging from 0 to p. The last share (`number3`) is generated by using this formula:

$$number3 = (number - number1 - number2) \mod p$$

Each client then simply sums the two secret shares it has recieved along with the share it has kept (`number3`). This number is then send to the hospital.

The hospital finally summarizes the 3 numbers it has recieved and modules that sum with p.

This ensure complete privacy, since none of the clients has any insight into what the other 2 clients secret number is, and the hospital only knows the result for the sum of all three clients numbers.