Report

Made by: Lucas Hanson (luha@itu.dk) for the course Security 1 at ITU.

How to run

To run the program you have to be in the folder with `main.py`. Then you have to run one of the following commands (it may depend on the version of python or OS you are running):

```
python3 main.py
```

```
python main.py
```

Approach

Firs the hospital will be started. It will then wait for the peers to connect. This is done in `hospital.socket_stuff()`.

When a peer is told to deal (generate all of the semi random values) they will first generate a random number `s` which has a limit in size. This will be followed with generating `s1` and `s2` which are also random and has the same limit as `s`. Then `s3` will be calculated so that if `s1`, `s2`, and `s3` are a added together and then modulo the limit, the result will be `s`. This is done to ensure that `s3` will not be negative. Then `s3` is added to the peers own list of values. This way none of the peers know to each of the other peers secret values. This generation and calculation is done in `peer.deal()`.

After this one of the peers will start to send the other peers `s1` and `s2`. The sending of `s1` and `s2` is done in `peer.send_aggregate_values()`.

Meanwhile the other peers will be listening for the other peers to send their values. When a share is received, the peer will add it to their list of values. The recieving is done in `peer.peer_socket_stuff()`.

They will take turns to send the other peers their values but they will always be listening for the other peers to send their values. The taking turns to send is controlled in the for loop on line 32 in `main.py`

When the peers have received all of the values, they will add them together and modulo the limit, followed by sending the value to the hospital. This is also done in `peer.peer_socket_stuff()`.

When the peer connect to the hospital, the hospital will append the value to its list of values. When the hospital has received all of the values, it will add them together. This is done in `hospital.aggregate()`. Due to the summed values being sent to the hospital being values summed by the different peers it will never know each of the peers secret values.

Since the peers secret are never shared no one will know their personal information.

Communication

The program uses sockets to communicate, using the socket module. The peers will connect to the hospital and the peers will also connect to each other. All of the communication also uses the ssl module, and the solution implements tls. Therefore the folder also contains self signed certificates. These would normally be signed by a certificate authority and only be used for a single peer. Since this is a test, I have used the same certificate for all of the peers and the hospital. The certificates would also be a secret and therefore not version controlled. But for ease of use I have included them in the repository/folder.

Limitations

I assumed that there are passive adversaries. If they where active the peers and hospital could end up with incorrect values. I also assumed that the peers would not be malicious, which would result in incorrect values as well.

There is a problem with the hospital calculating the incorrect sum (though it rarely happens). I am not entirely sure where the issue stems from. But I know that I would have to module with the max somewhere I am not currently.