Laboratory – Safety and Security

In this laboratory the main reason is to understand how RSA Algorithm works and the way it is used in automotive.

- Exercise 1

Complete **rsa_library.py** with a function called **encrypt** that encrypts the hex number sent by the client, using public key, with the method described in the Documentation.

- Exercise 2

Complete **rsa_library.py** with a function called **decrypt** that decrypts the hex number received by the server, using public key, with the method described in the Documentation.

- Exercise 3

Complete **rsa_library.py** with a function called **low_check** that check if the low part of the hex number is 0x01 (LOW = 0x01).

- Exercise 4

Complete **rsa_library.py** with a function called **number_check** that check if HIGH = ~LOW, where LOW = 0x01.

*** In **Server_gui.py** there are 2 global variables **flag**(number check flag) and **flag_low**(LOW = 0x01). These have to be used in ex. 7.

- Exercise 5

In **Server_gui.py** complete the **start_server** function in order to create the server as described in the documentation. The server needs to listen to the messages from client. Generate the public and private key (using **generate_keypair** function from **rsa_library.py**). You can use 277,239 as argument for the function. Server needs to send the public key and private key to the client. The client needs to store the keys in order to complete the next exercises.

In **Client_gui.py** complete the **start_client** function in order to create the client as described in documentation (using socket). Here you must read public and private key from the server.

- Exercise 6

In **server_gui.py** complete the function **send_key_data**, that is the action function for the **key** button, to encrypt the **unlockCar** variable, with the public key, and send it to the client.

- Exercise 7

In **server_gui.py** complete the function **recv_messages_handler**, this function is listening for the messages sent from the client, to decrypt the received messages and check if there is invalid low part (LOW!=0x01) or if the number doesn't have the format HIGH = ~LOW. If one of the errors appears than send a message to the client. If there is no errors than send a message to the client to let it know that all the conditions are met.

- Exercise 8

In **client_gui.py** complete the function **recv_handler**, this function is listening for the messages sent from the client, to decrypt the received messages.

If the received message is the **unlockCar** variable than set **airbag, corrupted_low** and **corrupted_high** buttons enable.

If the received message is an error message than display in one of the labels (**corrupted_low_label**,**corrupted_high_label**) a text.

If the received message is with no errors, than display a text in **airbag_on_label**

- Exercise 9

In **client_gui.py** complete the function **send_on_data**, that is the action function for the **Airbag on** button, to encrypt the **airbag_on** variable, with the public key, and send it to the server.

- Exercise 10

In **client_gui.py** complete the function **send_corrupted_low**, that is the action function for the **Corrupted low** button, to encrypt the **corrupted_low** variable, with the public key, and send it to the server.

- Exercise 11

In **client_gui.py** complete the function **send_corrupted_high**, that is the action function for the **Corrupted high** button, to encrypt the **corrupted_high** variable, with the public key, and send it to the server.