TU Clausthal

# Exercise – 6

**Published on:** 27.05.2024

**Deadline:** 03.06.2024 – 1:59pm

**Task(s):**

- Clone the Exercise GitHub repository: https://github.com/ETCE-LAB/ETCE-Exercises/
- The exercise is given to you in the form of a jupyter notebook. You can either install jupyter on your personal system, or use https://jupyter-cloud.gwdg.de/ (TIP: Jupyter Cloud allows you to clone the exercise repositorso you don't have to upload the jupyter notebook and other files manually)
- Programming language: Python 3.10+
  - **You only need to modify the „solution.ipynb" file**. More detailed instructions on where you need to insert your code can be found in this file.
  - The final cell in the solution.ipynb file grades your solution.
  - This will give you feedback on your solution.

To submit your solution, upload your modified ‚solution.ipynb' file to Moodle.

Task – IoT Security – Energy sellers

In E04 and E05, you first gathered weather data from different sources (weather sensors/APIs) and aggregated them into a single data set. Subsequently, you processed the data to make predictions based on the results of the gathered information. However, so far, we have discarded the aspect of privacy and security when handling IoT-related data. Especially the transmission of data from the sensors to the processing entity (cloud, edge, etc.) is often prone to data manipulation. Moreover, the sensor data might contain sensitive information that is not meant to be public. Therefore, data in transit must be protected against manipulation and encrypted. In this exercise, the Energy Seller and the Buyer will establish a secure communication channel over a public (eve's-droppable) communication channel, by perfoming an ECDH (Elliptic Curve Diffie-Hellman) Key Exchange.

Steps:

1. Use Argon2id and the provided salt to derive a key from the password "ETCE-SS2021-this-is-safe" (without quotes).
2. Load and decrypt the provided private key to be used by your node using the derived key.
3. Load the provided public key of your peer.
4. Generate an ephemeral public and private key pair and sign the public key with your private key. This signed private key shall then be sent to your peer, who will verify it.
5. Receive a signed, ephemeral public key from your peer and verify the signature.
6. Do a key exchange using your ephemeral private key and your peer's ephemeral public key.
7. As an energy seller, use the resulting shared secret to first decrypt an encrypted purchase request from your peer (the energy buyer), compute and encrypt the price for the request (encryption and decryption using an authenticated encryption mechanism) and send it back to your peer.