TU Clausthal

Exercise – 6

Published on: 05.06.2022

Deadline: 12.06.2022 – 1:59pm

Task(s):

- Download the handout zip archive here:

  https://sync.academiccloud.de/index.php/s/7bOP41vTBJIi2Bw

- Programming language: Python 3.10
- You can use this Virtual Machine for a pre-installed environment: Link (Password: `5cnN59dzVEm5atc`)
- Please watch the "Python-Exercise-Tutorial" summarizing how to do the python programming exercises.
- General Instructions
  - Unzip the handout zip archive
  - The handout contains a Pipfile. You can install the dependencies for the exercises by running `pipenv install`. (You might have to install pipenv and pyenv first)
  - Activate the python virtual environment using `pipenv shell`.
  - In the first sub-directory, you will see two files:
    1. solution.py
    2. driver.py
  - **You only need to modify the „solution.py“ file**. More detailed instructions on where you need to insert your code can be found in this file. The automated grading mechanism can grade your solution only if you follow the structure provided in the „solution.py“ file. For E06, you need to override the methods of the „Solution“ class where indicated.
  - You can use „driver.py“ to verify whether your program would pass the grading: `python3 driver.py`.
  - This file will give you feedback on your solution.
- Create a **zip** file of your submission:
  zip -r E06-<Your StudIP Username>.zip E06 Makefile Pipfile
- *Remember that your solution zip file should have **exactly** the same file format as the handout zip file.*
- To make it easier, you can just run `make zip` in the top-level handout folder to automatically create a zip archive with the correct directory structure.

- You can make a copy of the produced zip file named ''E06-submission.zip'' and run ''make'' to check if your file can pass the automatic grading.
- To submit your solution, upload the zip file to the in the timed submission folder in StudIP, „E06-Submissions“.

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.

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.