SIL765: Networks and System Security
Semester II, 2020-2021
Assignment-3

April 8, 2021

Submission Instructions

1. You will submit the assignment on Moodle.
2. You can use Piazza for any queries related to the assignment.
3. You should submit a single folder which should contain all the files related to this assignment.
4. The folder should be named as: ⟨Your Entry Number⟩-assignment-⟨Assignment Number⟩.
   Example: 2020EE10350-assignment-3
5. Each file should be named as specified in the problem.
6. You are free to use any programming language (preferably, C++ or Python).
7. Failure to strictly follow the instructions will adversely affect the grades.

Problem: Transport Layer Security (100 Marks)

In this problem, you will prototype the transport layer security (TLS) protocol to securely send a message from the server to the client. The message to be sent is "The OTP for transferring Rs 1,00,000 to your friend's account is 256345." The solution of this problem expects the following functionalities:

- *Trusted third party (TTP):* The TTP issues digital certificates to both the client and the server.
- *Server:* The server generates a public/private key pair, obtains a digital certificate from the TTP, performs the TLS handshake protocol with the client (including the protocol negotiation, authenticated key exchange and key transcript confirmation), and finally performs the TLS record protocol to securely send the message to the client.

- *Client:* The client generates a public/private key pair, obtains a digital certificate from the TTP, performs the TLS handshake protocol with the server (including the protocol negotiation, authenticated key exchange and key transcript confirmation), and finally performs the TLS record protocol to securely receive the message sent from the server.

The TTP, the server and the client have access to the following cipher suite.

- *Asymmetric key algorithm:* ECDSA or RSA,
- *Symmetric key algorithm:* AES or CHACHA20, and
- *Hashing algorithm:* SHA256 or SHA384

You are free to select the specific parameters of the algorithms. However, make sure that the three entities utilizes three different specifications from the available cipher suit (wherever it is possible). For instance, for creating a digital signature, the TTP may utilize RSA with SHA256, the server may utilize ECDSA with SHA384, and the client may utilize ECDSA with SHA256.

The three entities should be prototyped in three files namely: `my_ttp`, `my_server` and `my_client`. Your source codes should present the complete functionality (as discussed above) when the three files are executed in three different terminals in the following order: `my_ttp`, then `my_server`, and finally `my_client`.

Submission

- `my_ttp:` This should contain the trusted third party's source code.
- `my_server:` This should contain the server's source code.
- `my_client:` This should contain the client's source code.
- `readme:` This should be the pdf file containing all the necessary details about your solution. For instance, it should explain the steps to build and execute your code. It should have the screenshots of terminals to demonstrate that your code works as desired. It should contain discussions about the security and efficiency of the protocol.

Grading

- *Functionality (40%)*: The presented protocol should have all the required functionalities as discussed above.
- *Security (20%):* The presented protocol should not be vulnerable to known attacks. In the submitted `readme` file, include arguments about how your schemes prevents potential attacks, such as man-in-the-middle attack, replay attack, downgrade attacks, etc.
- Computational cost (10%): How many (symmetric/assymetric key) operations would it take for the three entities to successfully complete the protocol?

- Communication cost (10%): How many bits are communicated among the three entities to successfully complete the protocol?
- Presentation (20%): The presentation of the source codes and the details in the *readme* file should be clear.