National University of Singapore

School of Computing

CS2105 **Assignment 3** (5 Marks) Sem 1 AY22/23

Submission Deadline

11th Nov (Friday) 11:59 pm. 2 marks penalty will be imposed on late submission (Late submission refers to submission or re-submission after the deadline). The submission folder will be closed on **18th Nov (Friday) 11:59 pm**.

Objectives

In this assignment, you will implement a client which would use a session key to get encrypted data from a remote server and another program to check for the integrity of a file using the "Message authentication code". After completing this assignment, you should be able to implement a simple TCP-based client and have a good understanding of

- RSA,
- use of Session Keys to transfer data,
- Cryptographic Hash functions and
- verification of message integrity using Message Authentication Codes.

Exercise 1: Message Integrity

You will be implementing *Message Authentication Code* using SHA256 as the hash function (Refer to slide 47 in Lecture note 10). Given a file, you need to verify if the file content is tampered with. The content of the file is as in Fig 1.

Message Authentication Code (32 Bytes)	Data

Figure 1: File content

Input

The program takes 2 command line inputs

- The file containing the 32 byte "authentication key"

- The data file

Output

If the file contents are not tampered with, print "yes"; otherwise, print "no".

Exercise 2: Session Key

You will be implementing *Session Key* based file transfer using RSA and AES (Refer to slide 34 in Lecture note 10). You need to connect to a TCP server to get the session key followed by the file.
The client needs to perform the following

- Load the RSA Private key from the file `test/rsa_key.bin`
- Connect to the TCP server
- Receive 256B from the server
  - decrypt using RSA (link) to get the 16 B *Session Key*
- Receive data from the server
  - Receive data till connection closes
  - Data size is guaranteed to be a multiple of 16B
- Decrypt the data using AES in ECB mode (link) and write to the file.

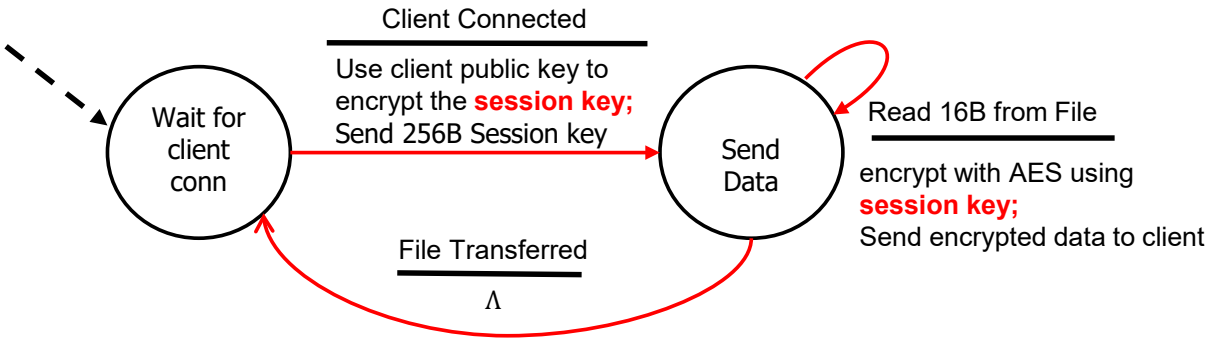

Figure 2: Server FSM

The finite state machine of the server is shown in Fig 2.

Input

The program takes 4 command line inputs

- The file to which the session key is to be stored

- The file to which the data is to be stored
- Hostname of the TCP server
- Port number of the TCP server

Notice

- The `stu` server has the package pycryptodomex (link)
  - The package name is "Cryptodome" instead of "Crypto"
  - SHA256: link
  - RSA Key: link
  - RSA Cipher: link
  - AES: link
- We have provided some template code.
- The outputs are case-sensitive
- There is a low possibility that the Server gets overloaded. So start the assignment early to avoid "congestion" during the last few days.

Grading Rubric

- Message Integrity: 2 Marks (equally divided among test cases)
- Session Key:
  - Correct Session Key: 1 Marks
  - Correct Data: 2 Marks

Group Work

All the work in this assignment should be done individually. However, if you find the assignment too difficult,

- you are allowed to form a group with another student
- maximum two students per group.
- Group submission is subject to **2 marks penalty** for each student.

Under no circumstances should you solve it in a group and then submit it as an individual solution. This is considered plagiarism. There will be no acceptance for excuses such as forgetting to declare as group submission. Please refer to the "Special Instructions for Group Submission" and "Plagiarism Warning" on page 5 for more details.

Grading

We will test and grade your program on the `stu` server. Please make sure that your program runs properly on `stu`. Moreover, you are allowed to use libraries installed in public folders of `stu` (e.g. `/usr/lib`) only.

We accept submission of only Python 3 (in particular, 3.8) program. For Python 3, we use the `python3` program installed in folder `/usr/bin` on `stu` for grading. We will **deduct 1 mark** for every type of failure to follow instructions (e.g. wrong program name).

We will grade your program based on its correctness only. A grading script will be used to test your program and no manual grading will be provided.

Testing Your Program

To test your program, please use your SoC UNIX ID and password to log on to `stu` as instructed on Assignment 0 paper.

- To use the grading script, please upload your program along with the `test` folder given in the package to `stu`. Make sure that your program and the `test` folder are in the same directory. Then, you can run the following commands to test your server program:

  ```
  bash test/Integrity.sh
  ```

  ```
  bash test/Session.sh
  ```

- By default, the script runs through all test cases. You can also choose to run a certain test case by specifying the case number in the command:

  ```
  bash test/Integrity.sh 3
  ```

  - `test/Session.sh` has only one test case

  To stop a test, press `ctrl-c`. If pressing the key combination once does not work, hold the keys until the script exits.

- If you ever encounter this error: **tput: unknown terminal "xterm-256color"** when testing your program using script provided, run the command:

  `export TERM=xterm` once after you log in and before you run the test scripts.

All of you will be connecting to a single server, hence start the assignment early to avoid "congestion" during the last few days.

Program Submission

For individual submission,

- Kindly create a zip file containing your source files (only)

  - `Integrity.py`

  - `Session.py`

- The zip file name should be <Matric Number>.zip where <Matric Number> is your matriculation number which starts with the letter A. An example file name would be A0165432X.zip.
- Submit it to the `Assignment_3_student_submission` folder of LumiNUS Files.
- All file names, including the zip file and all source files within the zip, are case-sensitive.
- In addition, your zip file should not contain any folders or subfolders or irrelevant files.

You are not allowed to post your solutions to any publicly accessible site on the Internet.

Special Instructions for Group Submission

For group submission, please include matriculation numbers of both students in the file name, i.e. `<Matric number 1>-<Matric number 2>.zip`. Submit it to the same `Assignment_3_student_submission` folder. An example file name would be `A0165432X-A0123456Y.zip`. For each group, there should be one designated member who submits the file, to avoid problems caused by multiple branches within a group. **Do not change the designated submitter!** If the group needs to upload a new version, it should be done by the same designated submitter as well.

Plagiarism Warning

You are free to discuss this assignment with your friends. However, you should refrain from sharing your program, program fragments, or detailed algorithms with others. If you want to solve this assignment in a group, please do so and **declare it as group work**.

We employ zero-tolerance policy against plagiarism. If a suspicious case is found, student would be asked to explain his/her code to the evaluator in face. Confirmed breach may result in zero mark for the assignment and further disciplinary action from the school.

Question & Answer

If you have any doubts on this assignment, please post your questions on LumiNUS forum before consulting the teaching team. However, the teaching team will NOT debug programs for students and we provide support for language-specific questions as a best-effort service. The intention of Q&A is to help clarify misconceptions or give you necessary directions.