DAT 510: Assignment 2

Submission Deadline: 23:59, Friday, Oct. 14, 2022

Implement Secure Communications

In this assignment, you will implement a secure communication scenario, which utilizes cryptographic primitives in a similar but rather simplified way as the real-world applications.

Part I.

After learned the subject of cryptography, Alice and Bob decide to secure their communications with three types of cryptographic primitives: symmetric ciphers, pseudo-random generators and public-key crptography (PKC)-based key exchange protocol. Below are their step-by-step procedure for realizing secure communications:

Step 1. Alice and Bob agree on the global parameters for DH-like key exchange: the cyclic group $G = \langle g \rangle$ they will use. E.g., a cyclic group $\mathbb{Z}_p^*$ with generator $g = 2$ (choose prime as $p = 2q + 1$ from a prime), or a cyclic group generated from an elliptic curve E. (refer to the textbook)

Step 2. Following **Diffie-Hellman's key exchange** scheme, Alice and Bob generate their own key pairs (PR_a, PU_a) and (PR_b, PU_b) separately

Step 3. Alice sends her public key PU_a to Bob, and Bob sends his public key PU_b to Alice;

Step 4. Alice generates a shared key K_{ab}, so does Bob;

Step 5. Alice and Bob are concerned about the strength of the shared key K_{ab}, so they agreed to use a same cryptographically strong pseudo-random number generator (**CSPRNG**), which takes K_{ab} as the seed, to generate a secret key K for subsequent encryption/decryption;

Step 6. Alice chooses a **symmetric cipher** to encrypt her file with the key K, and sends the encrypted file over public channel (e.g., Internet) to Bob;

Step 7. Upon receiving the encrypted file from Alice, Bob use the same key K and symmetric cipher to decrypt it and obtain Alice's file in the clear form;

Step 8. Both Alice and Bob are happy with their achievement: keep their communications confidential from adversaries in the middle.

The above process is illustrated in Figure 1. In this assignment, you are asked to implement this process for Alice and Bob with the cryptographic primitives you have learned.

Generally speaking, your program will be composed of three main components:

1. **Key Exchange**: such as DH's scheme, elliptic curve-based DH's scheme
2. **CSPRNG**: such as BBS generator, RC4 and block-cipher based PRNG

3. **Symmetric Cipher**: such as DES, 3DES, AES, SDES, KES (from Assignment 1)

You are free to choose your favourite ones or design them by yourself.

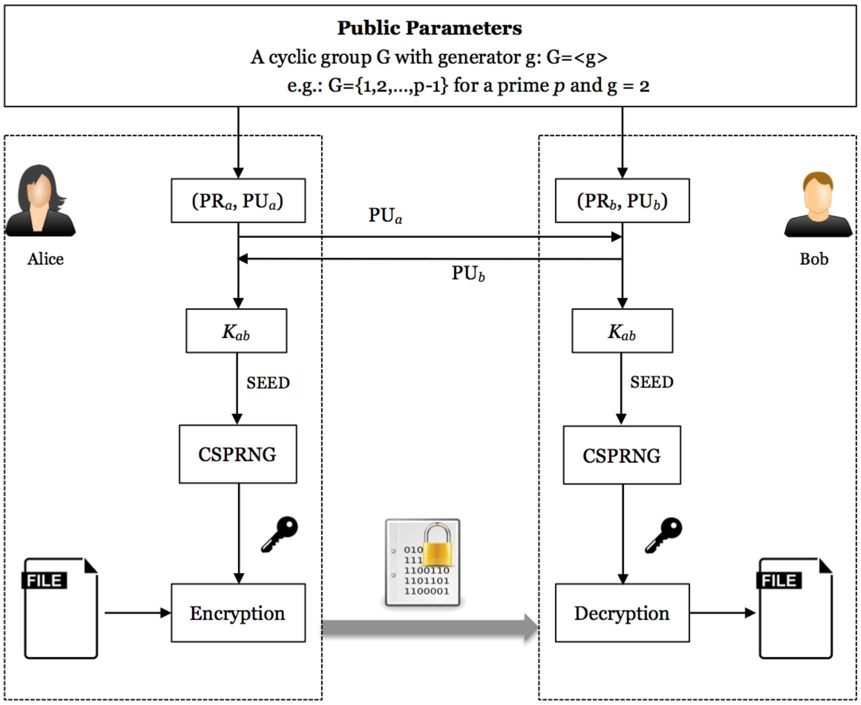

Figure 1: Secure Communications for Alice and Bob

Your program **should**:

- display the following information:
  - the cyclic group and public parameters you use;
  - the public keys of Alice and Bob;
  - the key K_{ab} shared by Alice and Bob;
  - the secret key K that will be used for encryption and decryption;

  the parameters and keys should be in the form of either decimal or hexadecimal;
- take input from Alice for encryption, the format for encryption can be binary/character string, files, etc,
- take input from Bob for decryption, the format for decryption can be binary/text file;
- show Bob can decrypt Alice's encrypted file as expected

  namely, show $Dec(Enc(IN_a, K), K) = IN_A$, where IN_A represents Alice's input file;
- have some kind of user interface, but not necessary GUI;

Your program is

- allowed to use existing libraries of **Symmetric Ciphers** (such as KES you implement in Assignment 1, or SDES, TripleSDES, 3DES, AES in built-in libraries);

- **not** allowed to use libraries for **CSPRNG** and **DH Key Exchange** (you have to implement them by yourself);

Part II.

You have implemented a simple web server in Assignment1 which receives ciphers and decrypts it based on the previously saved keys. However, in most real-world applications it is common for the key to be decided by both parties at the time of connection using the Key exchange methods. In this part you will use the Program You made in **Part I** to expand your web server to exchange keys before connecting and send data using the acquired key. Alice and Bob would each run an instance of the program and try to exchange keys at first. Later on, they will start to send and receive data using the key. The above process is illustrated in Figure 2, how your final application shall look and how the nodes need to communicate.

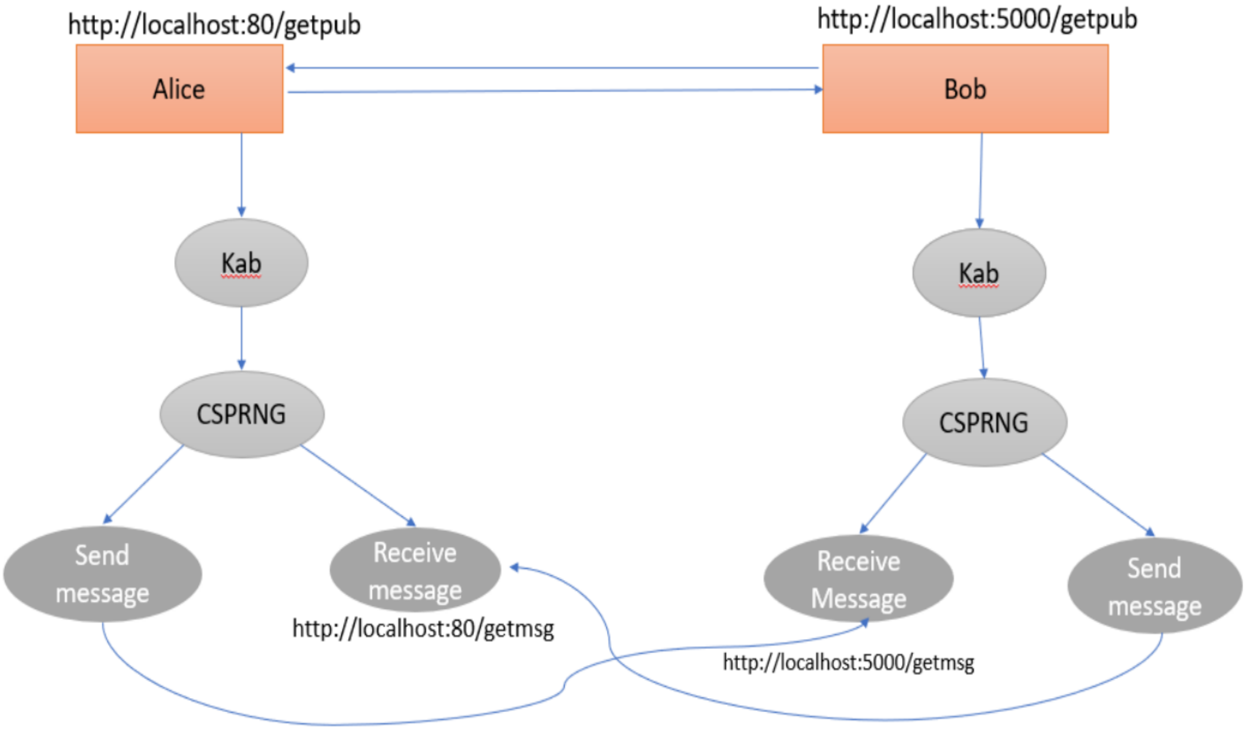
http://localhost:80/getpub
http://localhost:5000/getpub
Alice
Bob
Kab
Kab
CSPRNG
CSPRNG
Send message
Receive message
http://localhost:80/getmsg
Receive Message
Send message
http://localhost:5000/getmsg

Figure 2: Web server Implementation of Secure Communications

You are free to choose your design for the program as long as it meets the following results :

- Alice and Bob trying to find other party and share the public key generated by them.

- Both parties send and receive data (you can send strings, files, photos . . .) and show the result in the other side.
- Each side can be allocated sample demo data so that whenever they connect they can send it to the other side using the key generated and agreed by both of them.

Assignment Approval (by TA and SA)

Assignment approval will have a weight on your grade for the assignment. If you are not going to get the approval before deadline, your assignment will not be evaluated and you will fail the assignment.

What need to be done to get the approval of the assignment:

1. Show all parts of assignment is working i.e, show the code with proper comments, results.
2. Code should have a proper README file that describes the contents of the directory and any special instructions needed to run your programs (i.e. if it requires and packages, commands to install the package. describe any command line arguments with the required parameters).
3. Source code submitted for the assignment should be your own code. If you have used sources from the internet everything should be added to the references. If you used someone's code without reference, that will also be treated as plagiarism.
4. Provide the references in Code and Report, show these parts for TA's and Student Assistants.
5. You should **NOT** use available libraries/packages/classes for implementing the core functionality of the assignment.

You may use any "reasonable" programming language for part one of the assignment. Reasonable languages include: Java, C, C++, Python, MatLab, R and others with permission of Jayachander Surbiryala (Email: `jayachander.surbiryala@uis.no`) or Chunming Rong (Email: `chunming.rong@uis.no`).

Assignment Submission

Deadline: 23:59, Friday, Oct. 14, 2022 (submit your assignment through canvas)
Final submission:

1. Source Code
   - Source code submitted for the assignment should be your own code. If you have used sources from the internet everything should be added to the references. If you used someone's code without reference, that will also be treated as plagarism.
   - Source code should be single, compressed directory in .tar.gz or .zip format.
   - Directory should contain a file called README that describes the contents of the directory and any special instructions needed to run your programs (i.e. if it requires and packages, commands to install the package. describe any command-line arguments with the required parameters).
   - You should **NOT** use available libraries/packages/classes for implementing the core functionality of the assignment.
2. A **separate** report with PDF format
   - Texts in the report should be readable by human, and recognizable by machine;
   - Other formats will **NOT** be opened, read, and will be considered missing;
   - Report should follow the formal report style guide in next page.
   - Each student should write an individual report. Each report will be checked for plagiarism. If it is copied from some where else, you will fail the assignment.

NOTE: Please upload the archive file in *.zip, *.tar only and report in *.pdf format only to the website https://stavanger.instructure.com/.

Note: The assignment is individual and can **NOT** be solved in groups.

Project Title

Abstract

A one-paragraph summary of the entire assignment - your choices of cryptographic primitivesa and their parameters, procedure, test results, and analysis.

1. Introduction

A description of the scientific background for for your project, including previous work that your project builds on. (Remember to cite your sources!) The final sentence (analogous to the thesis statement in a term paper) is the objective of your experiment.

2. Design and Implementation

A detailed description (in paragraph format) of the design, procedure, and implementation of your project. This should be the main part of the report.

3. Test Results

Results of testing the software, as you observed/recorded them. Note that this section is only for observations you make during testing. Your analysis belongs in the Discussion section.

4. Discussion

Your analysis of what your testing results mean, and your analysis.

5. Conclusion

A short paragraph that restates the objective from your introduction and relates it to your results and discussion, and describes any future improvements that you would recommend.

References

A bibliography of all of the sources you got information from in your report.