IINF-2301 Assignment 2: Cryptographic File Sharing

Submission Deadline: Monday, October 8th, 2018 23:59

1 Introduction

In this assignment you will implement and analyse a file-sharing solution. The solution must include a server, which stores one or more files, and a client that can download those files. File transfer must be protected by crypto. You must use both symmetric and asymmetric encryption techniques. You may freely choose with particular algorithms to use in each case, but we suggest using AES for symmetric encryption, and RSA for asymmetric encryption. You are free to decide on key sizes, mode of operation, and the type of padding. The choices you make might be influenced by what is supported in your crypto library. The network communication between the client and the server should be implemented over a standard TCP connection.

2 Part A: Designing a cryptographic file sharing protocol

In Part A of this assignment, you are to implement the server and a client file-sharing application. When the server completed initialization, it should wait for a request from a client. When a client sends a request, the server should transmit the file, similar to what you did in Assignment 1. The challenge here is that the file has to be encrypted during transfer.

Hence, the server has to read the content of the file from its local file system, encrypt the data, and then transfer the ciphertext to the client. The client has to receive the ciphertext, decrypt it and then write the plaintext to a file in its local file system.

2.1 Requirements and restrictions

The server and the client MUST run as separate processes that only communicate over TCP or UDP sockets. Hence, you cannot assume any shared storage infrastructure, like Dropbox or Google Drive. The server should require only the following information:

- The local file path to the file it is going to serve.
- The TCP port number on which it is going to listen for incoming connections.

The client should require only the following information:

- The hostname or IP address of the server it is going to connect to
- The TCP port number on which the server is listening

It is acceptable to hard-code the TCP port number as a constant value in your application. But the server should accept the path to the file to serve as a command-line argument. Likewise, the client should accept the hostname of the server to connect to as a command-line argument. Your applications should not rely on any special format in the file that is exchanged. It should be treated as a binary file. However, for testing purposes we recommend using a text file in the beginning of

development. The contents of the file to server should NEVER be transmitted in plaintext over the network. You should treat the network as inherently hostile and assume that a third party is listening in to your all your networking traffic. You may assume that the connected client is legitimate and that the listening third party is not able to change the data exchanged over the network.

As we require that you use BOTH symmetric and asymmetric encryption, we recommend using asymmetric encryption to exchange a symmetric crypto key between the client and the server, and then use that key to symmetrically encrypt and decrypt the contents of the file.

2.2 Acceptable Languages and Libraries

For this assignment, you may use any programming language of your choice. You should keep in mind that we cannot offer much assistance on programming languages that are not know to us, or where we have little experience. Reasonable choices can include but are not limited to Python, Go, Ruby, Java, C#, C++ and C.

Note that you are not required to actually implement a cryptographic algorithm!

Usage of any library offering cryptographic functions is permitted. However, you may not use existing implementations of key-exchange protocols. In particular, you may not use existing libraries for SSL and TLS. If you are in doubt, please contact the TAs.

3 Part B: Security Analysis

In Part-B of this assignment, you are to analyse and scrutinize your file-sharing system with regards to the CIA and AAA security principles, described in the course syllabus. Your findings must be clearly documented in your report.

We do not recommend answering each letter of the acronyms sequentially. It is better if you write a general section in your report that answers all the aspects. Give examples of possible threats that your solution can be exposed to, what kind of information can a hostile third party get access to, and how would they go about gaining said access. Discuss what measures can be taken to achieve better security in your application presented in Part A. This discussion should include a description of a possible redesign or extensions of the existing solution and assumptions that must hold for the redesigned solution to work. We recommend that you mention digital signatures, certificates and certificate authorities during this discussion. Digital Certificates are bound to trust and a third part. Explain how all this is connected and how we can see it in your redesign.

Your discussions and analysis should both be practically related to Part A and extensive, proving that you understand and are able to analyse challenges and possible solutions related to network security. We expect that your work in Part B will require several pages of text in your report.

4 Deliverables

Submit your hand-in via Canvas before the specified deadline. You hand-in should consist of a separate pdf-file, as well as a SINGLE archive file such as a Zip-compressed Archive of your choice(.7z, .tar, .tar.gz, .tar.bz, .tgz, .tbz, etc.) or a WinRAR-archive (.rar). The name of the hand-infile should be in the following format: inf2301-2018.2-1-abc123.tar.gz, where abc123 is your student name.

Inside the archive folder, there should only be a single folder with the same name as the archive filename. This step is to ensure that multiple hand-ins can be extracted safely within the same folder by the grading TA.

Underneath that root folder, there should be a subfolder called “src” containing the source code. The hand-in should also include a README file in the root folder where you give source-code-specific instruction on how to compile and execute your source code.

Your report is handed in separately in Canvas as a pdf-file.

Reasonable use of external non-standard libraries is okay, but only if their use is properly documented in the hand-in and you give instructions on how to obtain them.