50.005 – Programming Assignment 2

Secure File Transfer

Team Members:

Ashlyn Goh Er Xuan	1002840	Cl02
Wong Chi Seng	1002853	Cl02

Instructions to Run

Prerequisite

Java is required to run the program.

Running the Program

Before running the programs, you need to make the following changes to the static variables. They can be found at the top of each program.

Running the Server (both CP-1 and CP-2):

- Change the static variables `privateKeyPath` and `serverCertPath` to the absolute paths of your private key file (.der file) and server certificate file (.crt file) accordingly. For our project, the private key file is named `example.org.der` while the server certificate file is named `example.org.crt`.

Running the Client (both CP-1 and CP-2):

- Change the static variables `filename` and `filepath` to the absolute file name and file path of the file you wish to transfer respectively.
- Change the static variable `CACSEcrtpath` to the absolute path of the CA's certificate (in our project, it is named as `cacse.crt`)
- Lastly, change the static variable `serverAddress` to the IP address of the computer running the server program (use "`localhost`" if you are running both on the same machine)

For both protocols, run the server program before running the client program. Upon successful file transfer, the file will be transferred to the server and can be found at the same directory.

Problem with Original Protocol

The problem with the original protocol is that it does not prevent a playback attack. Hence, an attacker can maliciously repeat a valid data transmission. In our case, the attacker can store information without authorisation and then retransmit it back to the client to trick the client into transferring the file.

To prevent the playback attack, we introduced a nonce into our protocol. The client generates a nonce and sends it to the server. On the other hand, the server must return the nonce that is encrypted with its private key back to the client. Thereafter, the client would check if the

decrypted nonce (with server's public key) matches the original nonce sent. More details are included in the specification diagram in the next section.

Protocol Specifications

AP Protocol

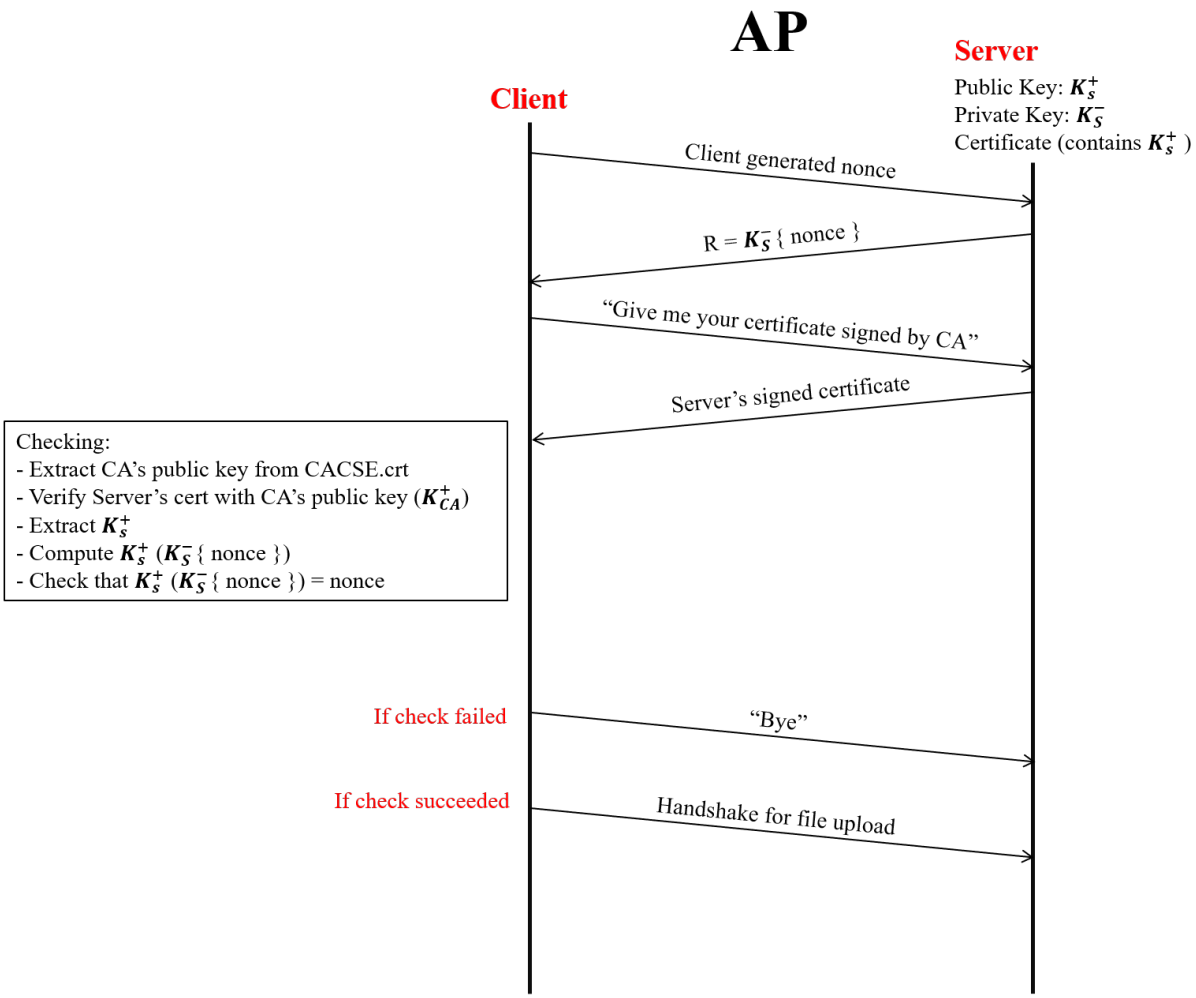

Figure 1: Authentication Protocol Specification

CP-1 Protocol

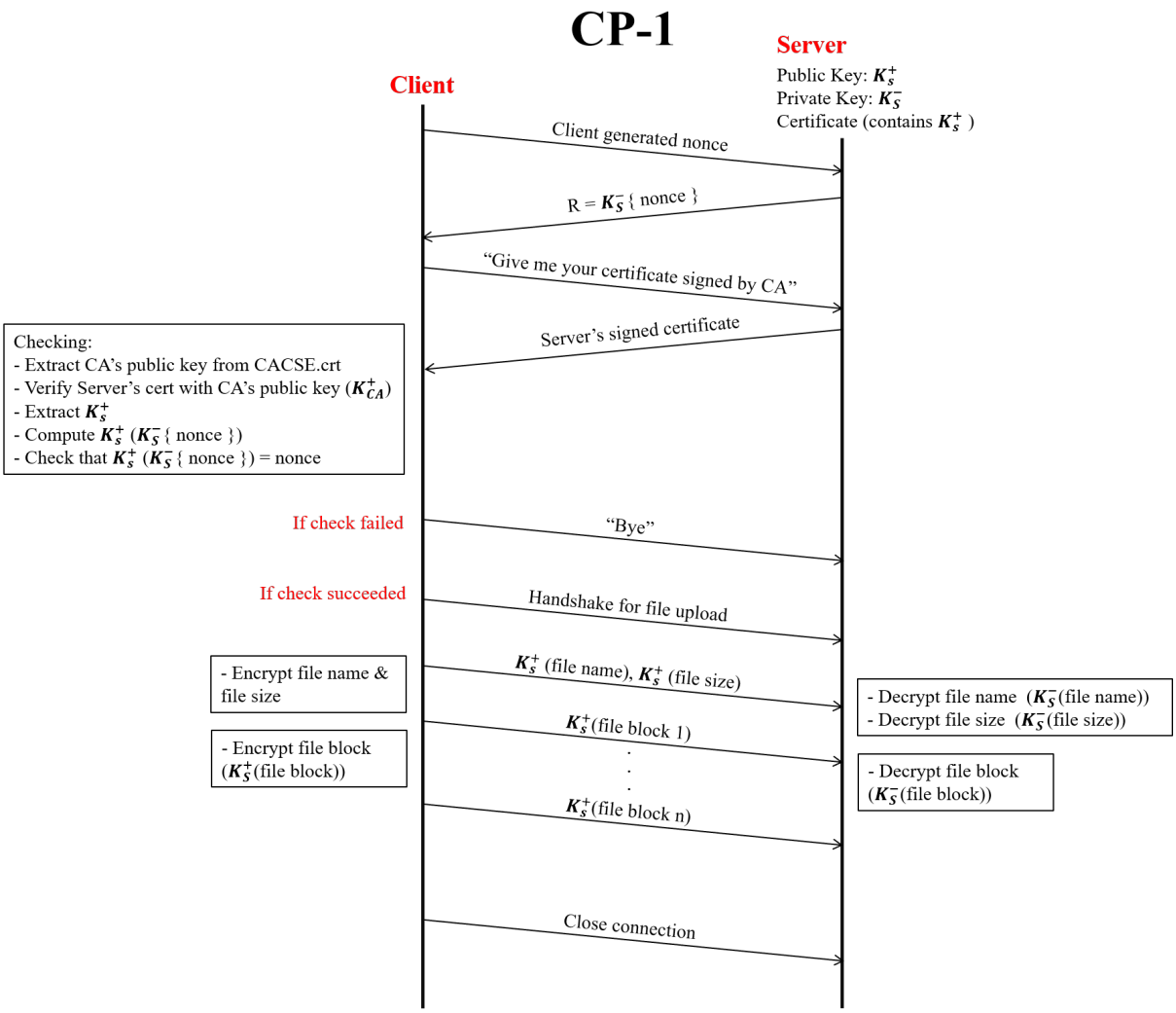

Figure 2: CP-1 Specification

CP-2 Protocol

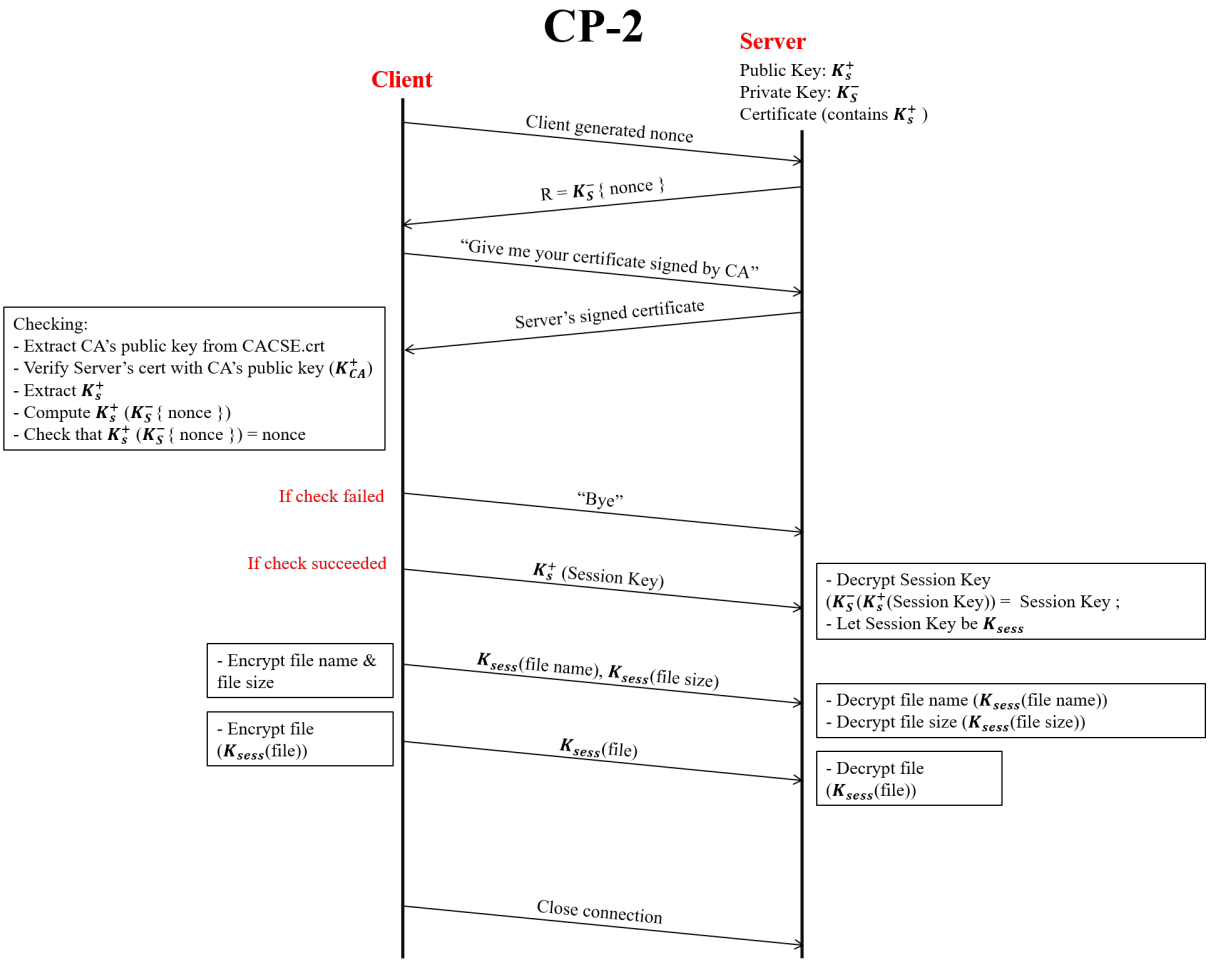

Figure 3: CP-2 Specification

Results

Table

Protocol	File Size (KB)	Time Taken (ms)	Throughput (KB/ms)
CP-1	33	1521.1146	0.021694618
	66	2461.2165	0.026816007
	123	4172.5188	0.029478597
	232	8527.9013	0.027204818
	400	12417.3577	0.032212972
	1484	50963.4025	0.029118935
CP-2	33	39.5271	0.834870254
	66	38.7928	1.701346642
	123	39.7007	3.098182148
	232	58.1925	3.986768054
	400	117.0683	3.41680882
	1484	194.2125	7.641114758

Plots

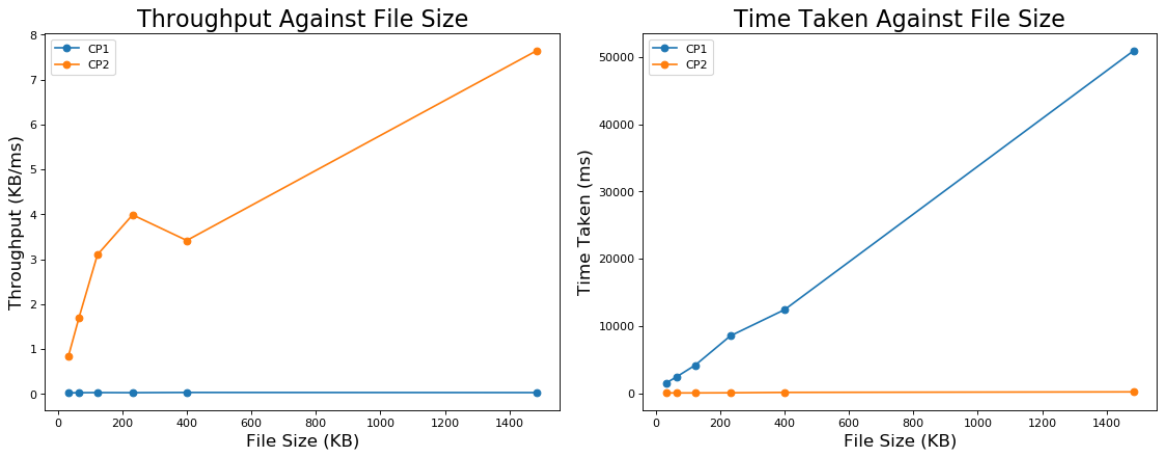

Figure 4: Throughput and Time Taken Against File Size

From the plots, the general trend is that the time taken for file transfer increases with increasing file size. Similarly, the throughput for CP-2 shows a general increase with increasing file size. However, the throughput for CP-1 remains relatively constant with increasing file size.

We also observe that it takes a much shorter time to transfer a large file using the AES encryption (using CP-2) than using RSA (using CP-1).

Conclusion

For the Confidentiality Protocol (CP) 1, each file is being cut into blocks and encrypted separately with RSA encryption as compared to CP-2, where the whole file is being encrypted with AES encryption and sent over. The time taken to encrypt every block and subsequently decrypt it back into the file would contribute heavily to the difference in timings between the two protocols. Since a smaller throughput would correspond to a longer time taken, it is unsurprising that CP1 gives a much smaller throughput as compared to CP2 (which takes a shorter amount of time).

From the graphs and the recordings, the timings and throughput seem to increase as the file size sent increases. We can also infer that the predicted behaviour is indeed true as the two protocols differ greatly in transfer timings and throughput for similar files.