Chapter 5 Analysis

Since the security of Secure Dropbox relies on nature of different cryptographies which has been justified when being proposed, this chapter will mainly outlines the methodologies taken in evaluating the Secure Dropbox in terms of computation performance and user experience. The method of data collection and analysis are explained and an overview of evaluation result will also be illustrated.

5.1 Evaluation Approaches

The Secure Dropbox was evaluated with the purpose of its computation performance as well as its user experiences as a client end encryption tool. Feedbacks of evaluation were sought from two categories of evaluators: Dropbox users with IT background and Dropbox users without any professional understanding of IT or computer science and work on IT unrelated industries.

Performance evaluation involves the efficiency of cryptography operations in which network situation matters sometimes. Efficiency of several cryptography involved procedure will be all tested based on measurements of text length and time to finish. Network environment counts when using sharing service delivered by Secure Dropbox.

Secure Dropbox was designed to fortify the security confidence of Dropbox users. Because it is difficult to emulate the scenario that Dropbox is under internal hacking and user’s file will be accessed by Dropbox employees who know the file encryption key, the design philosophy and security provided in certain application schema are explained from mathematical perspective and reasoning of its working procedure. The result data are basically extracted from experiencer’s oral expression about their thought of this application. Also, frequency of how misoperations happen is recorded with program context to indicate the usability of Secure Dropbox.

5.2 Performance Testing

5.2.1 Testing Schema

Performance of Secure Dropbox reflected in file encryption and decryption time consumption. Test is conducted under the environment of:

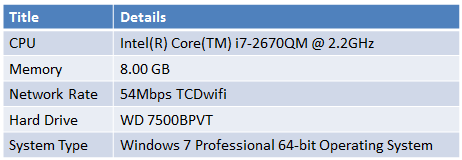


Figure 5.1 Testing Environment

To test the performance, several text file with different length from 1MB to 64MB has been created. Encryption time consumption is achieved during loading file into Secure Dropbox procedure. Decryption time consumption is achieved from both procedure of reading local file and shared file. Downloading time is recorded during reading shared file. For values in final performance record, each line is determined according to average value of 100 times test upon same file. The test is performed automatically with Python script. The results are listed as follows:

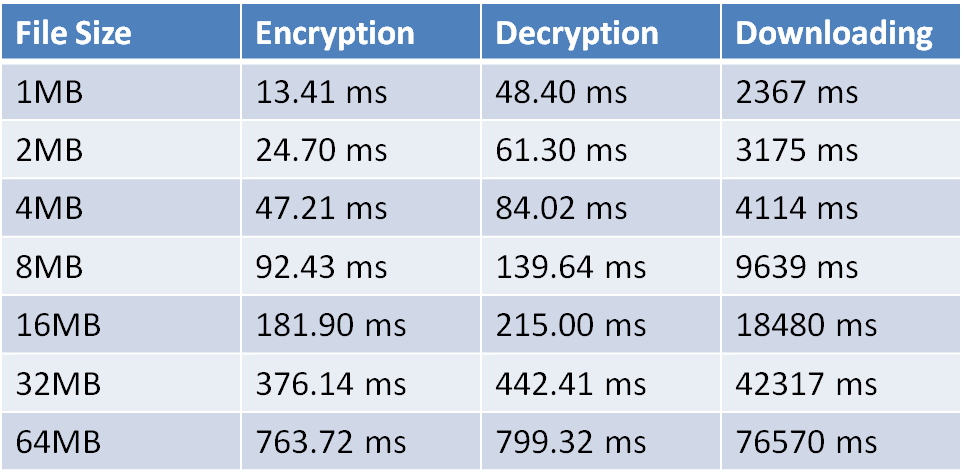


Figure 5.2 Secure Dropbox Cryptography Performances

5.2.2 Performance Evaluation

On the basis of testing result listed above, Secure Dropbox could encrypt a 1MB file within an average of 13.41 ms and this number comes to 763.72 ms when encrypting a 64MB file. It could be observed that the decryption costs roughly 35 ms more than encryption no matter what the file size is. The different time consumption could be attributed to the AES decryption preparation works like unpadding and initial vector fetching since AES encryption and decryption time consumption is theoretically the same. However, in comparison of the time consumption of file synchronization operation when reading shared file, the decryption time period which takes almost single-digit-percentage only could be ignored. For example, it costs more than 1 minute to download a 64MB but decrypted in 0.8 second. The result could be concluded as satisfying in terms of local file cryptography operation while the network situation becomes bottle neck when performing large shared file sharing.

5.3 User Experience

To make user experience evaluation a generic result, Dropbox users with or without IT background are both chosen as participants. In this section the feedback received from both of them will be summarized and discussed.

5.3.1 Session Procedure

Application configuration was conducted with delivering of packed Python egg of Secure Dropbox and readme file. Helps were given when necessary. Users tested Secure Dropbox on their PC or laptop. All the participants finished their sessions individually. During the session, participants were encouraged to finish several proposed tasks with Secure Dropbox client on their own. Participants with IT background were additionally assigned with tasks like Secure Dropbox configuring and performance evaluation. After user experience evaluation, participants were required to fill out a short questionnaire for collecting improvement comments. Users’ behavior confidence and time consumption on each task is recorded as well.

5.3.2 None IT Background User Feedback

Most participants with None IT background asked for help during Secure Dropbox configuration in terms of command line operation and terminologies that involved which were not familiar to them. During usage, Most of them were not skilled in command line operation but got accustomed to it after several trials. Some participants forgot to press enter after finishing the OAuth procedure in browser but just ignored the prompts and kept waiting in command line. All participants performed the file loading operation easily. Participant complained about changing the source file requires another file loading operation is easy to be forgotten. These users also commented with improvement suggestion that it would be better to make reading and writing function integrated in Secure Dropbox client so that another file loading operation will be not essential. However they accepted the reason why this is difficult to make a generic editing interface by explaining the fact that different files call for different editing interface just like for ppt files Microsoft Office Power Point is required for editing. The most time consuming steps for these participants were registration and login since they felt confusing when typing the password to register or login without echo on the screen. Some of them suspended the testing procedure and thought it was a program implementation problem until further help was given. This is designed to imitate the way Linux does when inputting password information but anyway caused extremely high typo happening. Some participants proposed that there should be a contact system with permanent storage of sharing recipients’ Secure Dropbox account name in order to share a file and there should be an interface to share a file with a group of users but not performing sharing operation one by one. One user advised that it would make him less concern about his file’s security if Secure Dropbox could perform encryption on the same file but not on the copy in Secure Dropbox folder so that there will be only one file encrypted instance. Otherwise he had to think about what to do with the plaintext file. Though, he also showed his concern that he was not confident with leaving only one encrypted file instance and worried about something tragic like the encrypted file could not be decrypted properly. Finally, it was advised by almost every participant that they prefer to do operation in GUI or application provides a file system style interface just like Dropbox official does.

5.3.3 IT Background User Feedback

Besides those relatively generic feedbacks, participants with IT background proposed more professional comments. Most of them advised that the file loading dialog should set file suffix filter to only display .txt file as Secure Dropbox only support text file and other file input will be recognized as invalid operation. A few of them concerned that the OAuth procedure via browser and file loading diagram might causes availability problem in pure command line interface. Some participants thought interface of querying file’s metadata in terms of cryptography like algorithm, key length and sharing records like with whom the file had been shared. Some else participants who had been informed that the Secure Dropbox might by nature disable the version control and file recovering service provided by Dropbox advised that an operation logging system should be implemented to trace the records in case of application errors. More significant design defects and implementation advice have been reported during performance evaluation. Most participants thought the application crashed or halted when reading the 64MB shared file and the way to solve is adding a progress by of downloading procedure and also any other time consuming operation. The downloading cost a good time but Secure Dropbox did not indicate the progress of downloading but just leave a blinking cursor in command line which seemed like the application does not respond. In the configure RSA key length step, some participants thought rather than configuration by modifying the python source file directly, an integrated configuration interface would not only make configuration convenient but also guarantee the robustness of Secure Dropbox by parameter checking in case someone changing the key length or any other configurable parameters with illegal expressions or values. Most importantly, a fatal design defects was discovered. Encrypting a large file like 64MB costs less than one second while consumes far longer time to synchronize file into Dropbox with Dropbox official client. However, Secure Dropbox by definition up uploaded file key encryption into KMS as long as the file has been encrypted. That time the file record could be seen in Secure Dropbox client but actually not in Dropbox yet. It works properly when performing local reading since there is encrypted file instance in local file system but exception occurred when trying to share the file with other user when the encrypted file in Dropbox folder is not synchronized yet. Basically it was because the /media interface which is called by Secure Dropbox to generate sharing url cannot find the specified file which is still during synchronization. Another cryptography application design with security weakness was pointed out that once file sharing is cancelled or its url is expired, the file encryption key should be changed at the same time in case of someone is able to get the encrypted file instance and decrypt it directly but not through shared file reading interface of Secure Dropbox client. There was implementation suggestion about integrate Secure Dropbox into file system so that it could be truly used seamless like the command sdls (Secure Dropbox ls) will be a default operation command and directly list files loaded to Dropbox through Secure Dropbox.

5.4 Discussion of Feedback

The feedback received from the both kinds of participants with regard to security was primarily positive. They believe a user end encryption would guarantee their file to be not accessible by Dropbox employees. Despite of those who have no concept about asymmetric cryptography, participants felt more confident with this new way of file sharing in comparison of the plaintext sharing mechanism by Dropbox.

Most of the criticism directed towards Secure Dropbox was concerned with the usability and design defects. Most participants thought user interface should be improved no matter in GUI or command line. The design flaws found during performance test might be fatal factors to availability and security of Secure Dropbox. Future work on this application will be done based on these results.