IERG4998C Final Year Project I

Encrypted Cloud Storage for Smartphones

BY

WONG SHING
1155109027

A FINAL YEAR PROJECT REPORT
SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF BACHELOR OF INFORMATION ENGINEERING
DEPARTMENT OF INFORMATION ENGINEERING
THE CHINESE UNIVERSITY OF HONG KONG

December 2020

Abstract

Owing to the convenience of storing and retrieving the data, cloud storage has been widely used by the public nowadays. With the rise of cloud storage usage, cloud data protection becomes an issue. Recently, more and more IT companies are having data leakage accidents. Hence, in this study, we are going to build an android app that allows users to have end-to-end encryption on their cloud data. The cloud files will be encrypted with users' input password and ChaCha20-Poly1305 AEAD. Without a valid password, the file content is incomprehensible. Besides, users can share the encrypted files using ECC (Elliptic-curve cryptography). Only the receivers with a valid private key can decrypt the file and read the content. Compared to the existing products, our app uses encryption algorithms that are more secure and run faster on mobile devices. We believe our app can provide the users with a better protection on cloud data.

1. Introduction

Amongst different types of attacks, insider attack seems to be the most dangerous one, because insiders are always granted for the highest privileges.

From 2018 to 2020, there are at least five insider attack incidents [1]. In 2018, a Cisco employee, who understood the weakness of Cisco's security mechanisms, purposely deployed malicious code that deleted 456 virtual machines used for Cisco's WebEx Teams application, leading roughly 16000 users of WebEx could not access their accounts for two weeks [1]. In July 2020, Twitter employees suffered a spear-phishing attack, which led to hackers accessing management tools [1]. It led a leakage of 130 private and corporate Twitter accounts, including those of Barack Obama, Bill Gates and Apple. These accounts were then used to promote a Bitcoin scam, and finally, the scam accounts received over $180000 in Bitcoin [1].

We can see that the consequences of the incidents are devastating. In the coming future, these incidents may happen to cloud storage providers. Hence, we promote an android app where users can do end-to-end encryption on their cloud data, which means they are the only one who can encrypt and decrypt the cloud files. Users can get rid of those potential risks. Our app will be open-sourced, which allows the public to examine its performance and reliability. Moreover, to increase the availability, users can upload the encrypted files to more than one cloud storages, the main cloud and the backup cloud.

In section 2, I will introduce an existing app that functions similarly to our app and discusses its disadvantages. In section 3, I will introduce our app and evaluate the chosen encryption algorithms. In section 4, I will show the implementation of our app and describe the evaluation methods. In section 5, I will talk about the works to do in the next semester. In section 6, I will give a conclusion.

Contribution Table:

Task		Contribution
Research	AES	60%
	RSA	30%
	ChaCha20-Poly1305 AEAD	55%
	ECC	60%
Coding	Google Drive Sign in	100%
	PBKDF2	0%
	ChaCha20-Poly1305 Encryption	100%
	ChaCha20-Poly1305 Decryption	0%
	Upload Function to Google Drive	100%
	Download Function to Google Drive	0%
App Design	Layout	50%
	Functionality	50%

2. Related Work

Boxcryptor [2]. Boxcryptor is an existing app that provides similar functions to our app. It uses AES (Advanced Encryption Standard) with a key length of 256 bits, CBC (Cipher Block Chaining) mode and PKCS7 (Public-Key Cryptography Standards 7) padding for files encryption while RSA (Rivest Shamir Adleman algorithm) with a key length of 4096 bits and OAEP (Optimal Asymmetric Encryption Padding) for file sharing. There are several problems with its design. First, the app does not have an integrity check, that the users cannot notice any malicious changes of the cloud data. Second, AES with CBC mode is not secure anymore. It is vulnerable to side-channel attacks such as Power analysis attack, Cache side-channel attack and Lucky 13. For RSA with OAEP padding, it may be vulnerable to adaptive chosen ciphertext attack.

Attacks on AES. AES is vulnerable to Cache side-channel attack [3] and Power analysis attack [4]. For Cache side-channel attack, it is an attack based on the leakage of the information stored in the T-table, a lookup table used for AES key generation. During the encryption process, the CPU will read the lookup table and output a key. To simplify the key generation process, the CPU data cache, a fast memory that allows faster information retrieval, may store some data of the lookup table. Monitoring the behaviours of the cache can thus infer the key. Two types of behaviours are monitored: the evolution in time of that state of the cache (Access-driven attacks) [3] and the differences in the overall execution time (Time-driven attacks) [3]. For Power analysis attack [4], it is an attack towards the initial AddRoundKey operation of AES. AddRoundKey is the last step for key generation in each iteration. It takes ‘state’ and ‘key’ as inputs and outputs their XOR bits. The XOR operation is the security gap that could be exploited to find the secret key [4]. To find the key, the power consumption of the device is monitored. The recorded power consumption is then converted into a digital format and put into the formula – PCC (Pearson Correlation Coefficient). The results are used to obtain the state. Once the state is known, by trying every possible key that fits in the state, the key can be obtained.

Attacks on AES-CBC. Lucky 13 [5] is a specific attack towards CBC mode. This attack based on the flaw in the TLS 1.2 (Transport Layer Security) specification on CBC mode. TLS is a protocol to provide confidentiality and integrity for the data in transit across untrusted networks [5]. It uses MAC (Message authentication code) for integrity checks. For CBC mode, plain text is divided into block size of 16 bytes. If a block is less than 16 bytes, padding bytes are added. The flaw is that TLS servers take a slightly different amount of time to process different TLS records with good and bad paddings [5]. The difference will be shown in the time at which error messages appear on the network. Attackers can then try with different values of padding bytes and do statistical analysis of multiple timing samples. Once they can distinguish the padding bytes, they can perform the padding oracle attack [5]. For CBC mode, the blocks are correlated during encryption and decryption. The blocks are encrypted or decrypted one by one consecutively, which means the current block will be affected if the content of the previous block is changed. By modifying the blocks and comparing the corresponding results, attackers are able to reveal the plain text.

Attacks on RSA-OAEP [6]. RSA-OAEP is the improved version of RSA algorithm. It is mathematically proven to be secure [7], [8], however, its implementation is still vulnerable to adaptive chosen ciphertext attack. This attack is based on the distinguishability of the oracle behaviors [6]. Oracle is a black-box that takes an input and answer a specific, fixed question regarding its input, without being restricted in terms of knowledge [7]. It can be used by any users, including the adversaries. Attackers can input different functions, integers, and ciphertext into the oracle and select information related to the plaintext.

3. Methodology

3.1 Overview

File Encryption

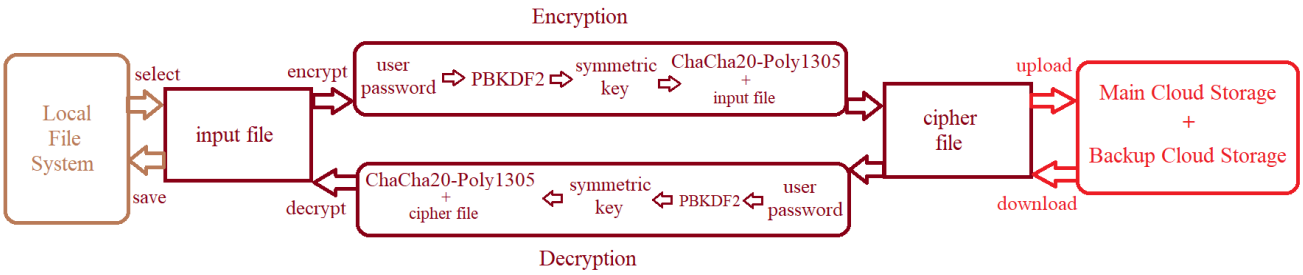


Figure 1. File Encryption

There is a main cloud storage and a backup cloud storage. Main cloud storage is the cloud storage that the user usually uses while the backup cloud storage is used for backing up the encrypted files stored on the main cloud storage. If the main cloud storage is attacked, and the files are deleted by the attackers, the encrypted files can still be found in the backup cloud storage.

For upload, the user first selects a file from the local file system. Second, he inputs a password of the file. The input password is then put into PBKDF2 to generate a key, followed by putting the generated key into ChaCha20-Poly1305 to encrypt the input file. Finally, user can press the upload button to upload the encrypted to the two drives at the same time.

For download, the user first downloads the cipher file from the main cloud storage. Second, he inputs the password of the cipher file. With a valid input password to PBKDF2, the same key can be generated and thus the cipher file can be decrypted using the same algorithm – ChaCha20-Poly1305. The decrypted file is saved to the local file system.

One additional point is that a random number Salt will be generated during PBKDF2, which is needed to be stored. So, the first 16 bytes of a cipher file will be its salt. During encryption, the app will automatically put the salt at the beginning 16 bytes of the cipher file. During decryption, the app will automatically read the first 16 bytes of the cipher file which is the Salt and input it into PBKDF2 to generate the same key for decryption.

File Sharing

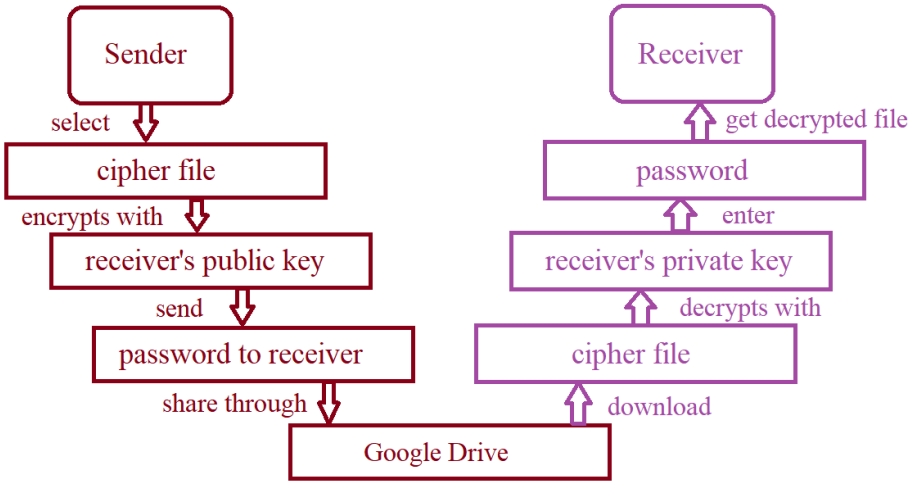


Figure 2. File Sharing (Assume the Main Cloud is Google Drive and the Backup Cloud is Dropbox)

For the sender, he first gets the public key of the receiver. Knowing the public key can do nothing to decrypt the shared file, so it is acceptable to share the public key through different mediums such as QR code, email, Facebook etc. Second, the sender selects a cipher file to share. Third, the app will encrypt the cipher file using the public key of the receiver. After that, a Google sharing dialogue is displayed for the user to input the receiver's email address. Finally, the file is shared through Google Drive Files Sharing.

For the receiver, he first downloads the shared file from Google drive to his phone. Second, the app retrieves his private key to decrypt the shared file. Third, the receiver inputs the password of the cipher file. After that, the app reads the Salt of the cipher file and the receiver input password to generate the key. With a valid password, the cipher file is decrypted and stored in the local file system. For the password, since both the private key and the password are needed to decrypt the cipher file, sending the password to the receiver under the insecure channels like QR Code, Whatsapp and Facebook is acceptable.

3.2 The encryption algorithms

ChaCha20-Poly1305 AEAD [9]. ChaCha20-Poly1305 is a AEAD (authenticated encryption with additional data) algorithm, where ChaCha20 is for encryption while Poly1305 is for authentication. This algorithm attains both the confidentiality and integrity.

ECC [10]. ECC is an approach to asymmetric encryption. It is based on the algebraic structures of the elliptic curves over finite fields and the difficulty of the ECDLP (Elliptic Curve Discrete Logarithm Problem) [10]. Under ECC, we use ECDH (Elliptic Curve Diffie-Hellman Key Exchange) [10] to generate the pairs of private key and public key.

3.3 Reasons of choosing the encryption algorithms

ChaCha20-Poly1305 AEAD

Security. ChaCha20 has a better security performance than AES. First, ChaCha20 is immune to padding-oracle attacks such as the Lucky 13 [11], which is a valid attack towards AES with CBC mode. Second, there is a research [9] on the security of ChaCha20-Poly1305. The research [9] concludes that ChaCha20-Poly1305 is resistant to various types of attacks including differential analysis, linear cryptanalysis and distinguishing attack, guess and determine analysis, algebraic attack, and attacks on initialization Process. Although some side-channel analysis attacks such as power analysis attack and fault inject attack are risks to Cha-cha, there are countermeasures against those attacks. To conclude, currently, no weakness can be found in ChaCha20-Poly1305 AEAD [9].

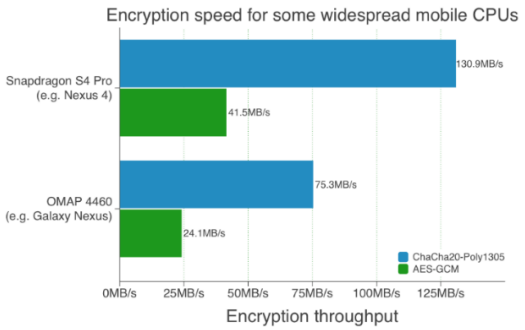


Figure 3. Speed performance of Encryption of AES-GCM and ChaCha20-Poly1305 [11]

Speed. ChaCha20-Poly1305 has a better speed performance than AES. From Fig. 3, taking AES-GCM as an example, we can see that ChaCha20-Poly1305 has an encryption throughput three times that of AES-GCM [11]. This is because ChaCha20 exclusively relies on operations that all CPUs natively support: additions, rotations, and XORs, which makes ChaCha20 encryption more responsive [12], [13]. Unlike ChaCha20, there is no hardware support for AES encryption [12], [13].

Practicability. Since 2014, Google has started to use ChaCha20-Poly1305 for the TLS cipher suite to secure the HTTPS internet connections between Chrome browsers and Google backend server [11]. It can be concluded that ChaCha20-Poly1305 is mature enough to be put in the market. This allows us to make our app production ready.

ECC (Elliptic-curve cryptography)

Security. There exist two well-known attacks towards ECC which are The Pohlig-Hellman attack [14] and the Pollard's rho attack [14]. However, these attacks can be overcome by careful implementation and careful selection of the domain parameters of ECC [14]. There are several research papers [15], [16], [17] stating that ECC is more secure than RSA, especially for resource constraint devices such as smartphones [16]. This is because ECC can attain the comparable security level to RSA with a much shorter key length than RSA. Fig.3 shows the key lengths of RSA and ECC, under a similar security performance. ECC is concluded to have better performance than RSA in software security, hardware security and Wireless LAN security [15].

Key Length		Time (s)	
RSA	ECC	RSA	ECC
1024	163	0.16	0.08
2240	233	7.47	0.18
3072	283	9.80	0.27
7680	409	133.90	0.64
15360	571	679.06	1.44

Fig 3: Table 2: Key generation performance of RSA and ECC [17]

TABLE IV
256 BITS ENCRYPTION, DECRYPTION AND TOTAL TIME (IN SECONDS)

Security	Encryption		Decryption		Total	
Bits	ECC	RSA	ECC	RSA	ECC	RSA
80	7.92	0.55	22.88	19.31	30.80	19.87
112	39.70	0.58	26.33	102.03	66.03	102.61
128	58.43	0.56	27.40	209.60	85.84	210.17
144	77.50	0.57	32.15	311.06	109.65	311.63

Figure 4: 256 bits input data Encryption, Decryption and Total Time (In Seconds) [16]

Speed. ECC is more efficient than RSA because ECC has a much shorter key length than RSA under comparable security level [16]. From Fig. 4, we can see that ECC uses a much shorter time for the whole process of encryption and decryption than RSA under the same security bit level. Speed is the main motivation for us to choose ECC instead of RSA because it works better in resource constraint devices such as smartphones.

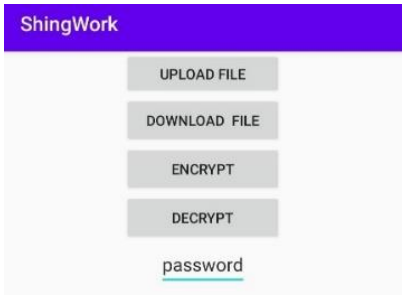
Practicability. Google uses ECC for key generation [18], which is the same as our approach. There may be some difference in the implementation details, but the basic principle is the same. This tells that our approach is likely to be mature enough to put in the market. This result is important because algorithms without the recognition of the industries are inappropriate to be used, even that the algorithms are mathematically proved secure, because there must be some other problems of the algorithm that makes it to be deprecated like difficulty in implementation.

4. Results

4.1 Implementation

1. User selects the cloud storage account. Our app now supports Google Drive only.

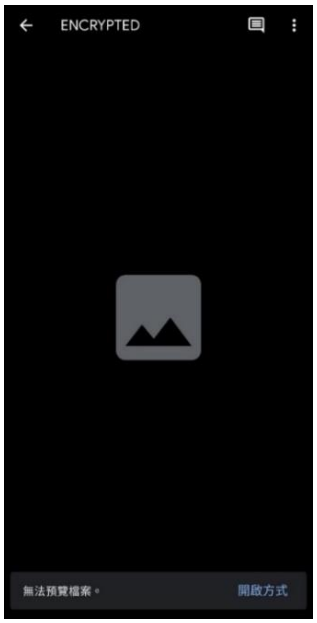
2. User selects the target file from the local file system (FYPIMAGE.jpg) and input a password to encrypt the file. After that, the encrypted file (ENCRYPTED.jpg) is generated.



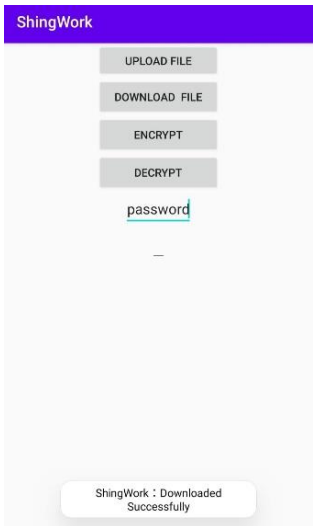
3. User press the button ‘UPLOAD FILE’ to upload the file to Google Drive.

4. User can see the encrypted file ‘ENCRYPTED’ on the drive.

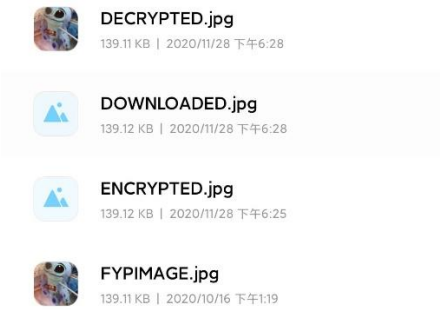
5. The encrypted file cannot be read.



6. User downloads the encrypted file from Google Drive and enter the password to decrypt. (We can now only demonstrate one single file. Selection of the encrypted files From Google Drive is not available.)



7. User can read the decrypted file in the local file system of his mobile phone.



4.2 Evaluation

Performance. At the end of the app development, we will test the speed performance of encryption and decryption, including their throughput for different size of input files and for different mobile phones with different CPUs. The results will be presented as graphs. For the performance on security, it is considered as satisfied if the encrypted files cannot be read on the cloud storage, the encrypted files can only be decrypted with a valid password, and the shared file can only be decrypted with a valid private key.

Usability. We will find test users to use our app and give comments on our app. The comments should include the usability on files selection, files encryption, file decryption and files sharing. Usability has aspects including the ease of finding, selecting, encrypting, uploading, downloading, decrypting the target file(s), and the ease of sharing the file(s) with others.

5. Future Work

We have finished the implementation of the symmetric key encryption – ChaCha20-Poly1305 AEAD as well as the upload and part of the download functions of Google Drive. In the next semester, we will finish the download functions of Google Drive, finish the implementation of the asymmetric key encryption – ECC as well as the upload and download functions of other cloud storages such as Dropbox. Moreover, we will improve the UI (User interface) design and make our app to have an attractive layout.

6. Conclusions

We present an approach to provide additional protection on cloud data, by building an android app which allows end-to-end encryption for cloud files. The algorithms used in our app, ChaCha20-Poly1305 AEAD and ECC, have an overall better performance than that used in Boxcryptor. We offer the confidentiality, integrity and availability at the same time. The app will be further improved in the next semester.

7. Reference

[1] Ekran System, “5 Real-Life Examples of Breaches Caused by Insider Threats,” *5 Real-Life Examples of Insider Threat-Caused Breaches | Ekran System*, 18-Nov-2020. [Online]. Available: https://www.ekransystem.com/en/blog/real-life-examples-insider-threat-caused-breaches. [Accessed: 01-Dec-2020].

[2] Boxcryptor, “Technical Overview,” – *How Boxcryptor's Encryption Works*, 2011. [Online]. Available: https://www.boxcryptor.com/en/technical-overview/. [Accessed: 01-Dec-2020].

[3] M. Neve and K. Tiri, “On the complexity of side-channel attacks on AES-256 - methodology and quantitative results on cache attacks -,” *CiteSeerX*, 2007. [Online].

Available: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=CD28FC60E241D2B7127352966725CD63?doi=10.1.1.74.9129&rep=rep1&type=pdf. [Accessed: 01-Dec-2020].

[4] S. D. Putra, M. Yudhiprawira, S. Sutikno, Y. Kurniawan, and A. S. Ahmad, “Power analysis attack against encryption devices: a comprehensive analysis of AES, DES, and BC3,” *TELKOMNIKA*, vol. 17, no. 3, pp. 1282–1289, Feb. 2019.

[5] Al Fardan, N. J., Paterson, and K. G., “Lucky Thirteen: Breaking the TLS and DTLS Record Protocols,” *2013 IEEE Symposium on Security and Privacy*, pp. 526–540, Feb. 2013.

[6] J. Manager, "A Chosen Ciphertext Attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as Standardized in PKCS #1 v2.0," in *Annual International Cryptology Conference.: CRYPTO 2001, August 2, 2001, Berlin, Heidelberg* [Online]. Available: Springer, https://link.springer.com/chapter/10.1007/3-540-44647-8_14. [Accessed: 2 Dec. 2020]

[7] Y. Romailler, “Breaking RSA OAEP with Manger's attack,” *Kudelski Security Research*, 05-Apr-2018. [Online]. Available: https://research.kudelskisecurity.com/2018/04/05/breaking-rsa-oaep-with-mangers-attack/. [Accessed: 01-Dec-2020].

[8] E. FujisakiJ, T. Okamoto, D. Pointcheval, J. Stern, "RSA–OAEP Is Secure under the RSA Assumption," in *Annual International Cryptology Conference.: CRYPTO 2001, August 2, 2001, Berlin, Heidelberg* [Online]. Available: Springer, https://link.springer.com/chapter/10.1007/3-540-44647-8_16. [Accessed: 2 Dec. 2020]

[9] KDDI Research, Inc, " Security Analysis of ChaCha20-Poly1305 AEAD," Cryptography Research and Evaluation Committees. (CRYPTREC), Japan, 2601, 2016, pp. 2-32. Accessed on Dec. 2, 2020. [Online]. Available: https://www.cryptrec.go.jp/exreport/cryptrec-ex-2601-2016.pdf

[10] Lopez, J., & Dahab, R. (2000, May 22). *An Overview of Ellipti Curve Cryptography* [Scholarly project]. In *CiteSeerX*. Retrieved December 02, 2020, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.37.2771&rep=rep1&type=pdf

[11] E. Bursztein, “Speeding up and strengthening HTTPS connections for Chrome on Android,” *Google Online Security Blog*, 24-Apr-2014. [Online]. Available: https://security.googleblog.com/2014/04/speeding-up-and-strengthening-https.html. [Accessed: 01-Dec-2020].

[12] M. Robshaw, O. Billet, and D. J. Bernstein, “The Salsa20 Family of Stream Ciphers,” in *New stream cipher designs: the eSTREAM Finalists*, Berlin, Germany: Springer, 2008, pp. 84–97.

[13] D. J. Bernstein, “ChaCha, a variant of Salsa20,” 20-Jan-2008. [Online]. Available: http://cr.yp.to/chacha/chacha-20080120.pdf. [Accessed: 02-Dec-2020].

[14] H. Tange and B. Andersen, "Attacks and Countermeasures on AES and ECC," in *2013 16th International Symposium on Wireless Personal Multimedia Communications*

(WPMC), June 24-27, 2013, US, NJ, Atlantic [Online]. Available: IEEE, https://ieeexplore.ieee.org/abstract/document/6618523. [Accessed: 2 Dec. 2020]

[15] D. Mahto and D. K. Yadav, "RSA and ECC: A Comparative Analysis ," *International Journal of Applied Engineering Research*, vol. 12, no. 19, pp. 9053–9061, Jan. 2017.

[16] D. Mahto, D.A. Khan, D.K. Yadav, "Security Analysis of Elliptic Curve Cryptography and RSA," in *2016 World Congress on Engineering: Vol I, June 29, 2016, London, U.K.* [Online]. Available: http://www.iaeng.org/publication/WCE2016/WCE2016_pp419-422.pdf. [Accessed: 2 Dec. 2020]

[17] M. Gobi, R. Sridevi, and R. Rahini priyadharshini, "A Comparative Study on the Performance and the Security of RSA and ECC Algorithm," *International journal of advanced network and application* , pp. 168–171, Mar. 2015.

[18] A. Langley, "Protecting data for the long term with forward secrecy," *Google Online Security Blog*, 22-Nov-2011. [Online]. Available: https://security.googleblog.com/2011/11/protecting-data-for-long-term-with.html. [Accessed: 01-Dec-2020].

Consolidated Logbook for FYP Thesis I

Name: WONG Shing
Student ID: 1155109027

Week 2:

Planning:

I plan to get familiar with Android Studio. I will first set up the Android Studio Environment, to ensure that it is able to work on our computer and able to connect to my phone to check the final layout. Second, I plan to work on the details/contents of Android Studio. For example, creating basic layouts, adding corresponding functions and so on. Overall, I plan to work smoothly on Android Studio.

Done:

I have read documents on the official Android Studio website. I have got some knowledge on the structure and design of Android studio, like activity, manifests, gradle and so on. I have also tried different templates in the Android Studio. Trying to understand why the codes are there and how they are connected. Now, I can roughly work smoothly on Android Studio.

Week 3:

Planning:

After working on the Android Studio, I plan to connect to Google Drive and work on the upload and download. I may look at the Google Drive API document to get the details of it.

Done:

I have found that the Google Drive API Android is deprecated. Therefore, I am still finding other ways to connect Android Studio to Google Drive. I have found some resources online showing the ways to connect to Google Drive after the deprecation of the Official API. I may try to implement it next week.

Week 4:

Planning:

After working on the Android Studio, I plan to connect to Google Drive and work on the upload and download. I may look at the Google Drive API document to get the details of it.

Done:

I have looked at the Google Drive API with Java because the Google Drive API Android is deprecated, and it is migrated to Restful API. I have successfully set up the Google Account Login and Google Drive Access. I can now successfully upload files to google drive. For download or other operations, I am still working on it and wish I can manage it by next week.

Week 5:

Planning:

I plan to finish the proposal. To finish the proposal, I may do some research on the encryption methods, existing products and the methodology of our app. I may have to confirm the entire structure of the project and ensure that all the planning is workable.

Done:

I have found an existing product which is similar to our project namely Boxcryptor. It is an app to encrypt and decrypt files stored on the cloud. We then think of a new functionality which is to build a backup cloud storage, the data will not be lost even if they are cleared on the main cloud storage. For encryption, we have picked some common choices like AES, AEAD and CHACHA. We may later test the speed and security performance of the above encryption method.

Week 6:

Planning:

This week, I plan to look into the part - cryptography. I plan to find out the advantages and disadvantages of the cryptography method. Also, I plan to see if the cryptography can work on the Android Studio.

Done:

I have looked into many different encryption methods and finally got some ideas on it. I have found some common and powerful encryption methods such as AES-GCM, CHACHA and HashMac. And all of them can work on Android Studio. I am happy that the common encryption can work on Android Studio so that it is available for our project.

Week 7:

Planning:

I plan to watch the video that introduces the basic structure of a report and several reminders. Then, I will start writing the interim report.

Done:

I have watched the video and started writing the interim report. To do the interim report, I have done some detailed research on cryptography so that I can do complete analysis on different encryption methods. Besides, I have done some data search to explain why we choose Dropbox and Google Drive among tens of cloud storage platforms.

Week 8:

Planning:

I will be finished writing the interim report. Then, I start to work on the coding part. I would like to finish the upload function of the app.

Done:

I have finished writing the interim report. Finally, we choose ChaCha20-Poly1305 AEAD for symmetric encryption while ECC for public key encryption. I have done a lot of research on different cryptosystems and have put all the analysis and comparison in the interim report. For the coding part of the upload function, I am still working on it and hope to finish it by next week.

Week 9:

Planning:

After the interim report, I think I should work on the coding part, the implementation part. I plan to finish the download function of our app.

Done:

I have finished the download function of our app. Files can be downloaded from the driver to the phone with input of Google File ID. However, there are still some minor problems like only the files uploaded by our app can be downloaded. However, I think I may first put this problem aside and keep going.

Week 10:

Planning:

I have now finished two functions: upload and download. Now, I am going to implement an encryption method.

Done:

I have found the necessary libraries needed for the encryption method. I am looking at the structure for building the encryption function. For instance, I have to include the key stream, algorithm used, original text, output path etc. Also, I am finding ways to store the private keys securely.

Week 11:

Planning:

We have received the comments of the interim report. We plan to update the interim report and make it to be the final report.

Done:

I have improved the content of the interim report. I have added connective wordings in the related work section. I have summarized the methodologies into some charts with explanations. I have also added some current implementations in the report, showing our current process. Also, I have checked all the grammar and spellings.

Week 12:

Planning:

We have done a lot in the written part, that is the report. This week, we plan to work on the coding part. We plan to finish the symmetric key encryption function.

Done:

We have successfully finished the symmetric key encryption function. Users can now encrypt, upload, download and decrypt their files.

The Chinese University of Hong Kong
Academic Honesty Declaration Statement

Submission Details

Student Name	WONG Shing (1155109027)		
Year and Term	2020-2021 Term 1		
Course	IERG-4998-CJ01 Final Year Project I		
Assignment Marker	Professor CHAU Sze Yiu		
Submitted File Name	1155109027 Sem 1 Thesis.pdf		
Submission Type	Individual		
Assignment Number	5	**Due Date (provided by student)**	2020-12-03
Submission Reference Number	2796259	**Submission Time**	2020-12-02 21:11:10

Agreement and Declaration on Student's Work Submitted to VeriGuide

VeriGuide is intended to help the University to assure that works submitted by students as part of course requirement are original, and that students receive the proper recognition and grades for doing so. The student, in submitting his/her work ("this Work") to VeriGuide, warrants that he/she is the lawful owner of the copyright of this Work. The student hereby grants a worldwide irrevocable non-exclusive perpetual licence in respect of the copyright in this Work to the University. The University will use this Work for the following purposes.

(a) Checking that this Work is original
The University needs to establish with reasonable confidence that this Work is original, before this Work can be marked or graded. For this purpose, VeriGuide will produce comparison reports showing any apparent similarities between this Work and other works, in order to provide data for teachers to decide, in the context of the particular subjects, course and assignment. However, any such reports that show the author's identity will only be made available to teachers, administrators and relevant committees in the University with a legitimate responsibility for marking, grading, examining, degree and other awards, quality assurance, and where necessary, for student discipline.

(b) Anonymous archive for reference in checking that future works submitted by other students of the University are original
The University will store this Work anonymously in an archive, to serve as one of the bases for comparison with future works submitted by other students of the University, in order to establish that the latter are original. For this purpose, every effort will be made to ensure this Work will be stored in a manner that would not reveal the author's identity, and that in exhibiting any comparison with other work, only relevant sentences/ parts of this Work with apparent similarities will be cited. In order to help the University to achieve anonymity, this Work submitted should not contain any reference to the student's name or identity except in designated places on the front page of this Work (which will allow this information to be removed before archival).

(c) Research and statistical reports
The University will also use the material for research on the methodology of textual comparisons and evaluations, on teaching and learning, and for the compilation of statistical reports. For this purpose, only the anonymously archived material will be used, so that student identity is not revealed.

I confirm that the above submission details are correct. I am submitting the assignment for:

[X] an individual project.

I have read the above and in submitting this Work fully agree to all the terms. I declare that: (i) the assignment here submitted is original except for source material explicitly acknowledged; (ii) the piece of work, or a part of the piece of work has not been submitted for more than one purpose (e.g. to satisfy the requirements in two different courses) without declaration; and (iii) the submitted soft copy with details listed in the <Submission Details> is identical to the hard copy(ies), if any, which has(have) been / is(are) going to be submitted. I also acknowledge that I am aware of the University's policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the University website http://www.cuhk.edu.hk/policy/academichonesty/.

I declare that I have not distributed/ shared/ copied any teaching materials without the consent of the course teacher(s) to gain unfair academic advantage in the assignment/ course.

I also understand that assignments without a properly signed declaration by the student concerned will not be graded by the teacher(s).

Signature (WONG Shing, 1155109027)

2 Dec 2020

Date

Instruction for Submitting Hard Copy / Soft Copy of the Assignment

This signed declaration statement should be attached to the hard copy assignment or submission to the course teacher, according to the instructions as stipulated by the course teacher. If you are required to submit your assignment in soft copy only, please print out a copy of this signed declaration statement and hand it in separately to your course teacher.