ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 10(8), DOI: 10.17485/ijst/2017/v10i8/104511, February 2017* ISSN (Online) : 0974-5645

SoloEncrypt: A Smartphone Storage Enhancement Security Model for Securing users Sensitive Data

Solomon Babatunde Olaleye[1*], Ishan Ranjan[1] and Shrikant Ojha[2]

[1]Department of Computer Science and Engineering, Sharda University, Greater Noida - 201306, India; olaleye3@yahoo.com, ishan.ranjan@sharda.ac.in
[2]Research and Technology Development Centre, Sharda University, Greater Noida - 201306, India; shri.kant@sharda.ac.in

Abstract

Objectives: To enhance the storage space of smart phones by using cloud infrastructure for storage. However, the biggest challenge is data security. **Methods/Statistical Analysis:** Existing Android smart phones are studied and a new security model named Solo-Encrypt security architecture is proposed. Advanced Encryption Standard (AES) algorithm of 256 bits key length was used to encrypt and decrypt user sensitive data using cloud. **Findings:** Three (3) Android devices were used and the experimental results are presented based on encryption time, decryption time and throughput on each device. The results showed that the proposed work can effectively protect user sensitive data with speed using cloud. **Application/Improvement:** Smart phones data are moved to cloud storage in encrypted form, smart phones storage are enhanced by making more storage spaces available for use and security of user sensitive data is improved.

Keywords: AES, Decryption, Encryption, Storage, Smartphone, Solo-Encrypt Architecture

1. Introduction

Smart phones today have become part of human existence. Human dressing is not complete without a smart phone. Android among the smart phones are most widely used. It has gained popularity based on its features and functionalities. They have been very useful in voice communication, Short Message Service (SMS), snapping pictures, video calls, record keepings, managing contacts etc. Smart phones do not only carryout the basic functions listed but in addition access the Internet, e-mail, network games and for business activities[1].

Moreover, users of smart phones use their mobile phones to store a lot of personal information such as contacts, credit card numbers, debit card numbers, bank account details, Personal Identification Number (PIN), passwords, SMS, e-mail and so on. This user information is so sensitive that they are supposed to be kept confidential. By the smaller sizes of mobile devices they can easily be lost and stolen. Since smart phone is also used to connect to the Internet, there is possibility of threat to information stored. They categorized threats to smart phone into four[2]. These are 1. Application based 2. Web based 3. Network based and 4. Physical based. The application based is malwares and spywares threats that can cause smart phone to behave contrary to its normal functioning. A mobile phone that is regularly connected to the Internet may be affected with web and network based threats. The physical based threat means it can be stolen or got lost. Based on these above discussion and due to the fact that smart phones are widely used all over the world there is the need to provide secure backup of data and restore such data into the device when needed[3,4].

Despite smart phones benefits, there are many issues to be addressed[5]. These issues are battery life, storage capacity, security, network connection dependency, integrating applications among others. Battery life goes down quickly with usage especially when doing serious computation, storage space not always enough because the Operating System (OS) and other system software would have taken a lot of storage space and also user downloaded applications. We want to enhance the smart phone storage on the client side by using cloud infrastructure. However, the biggest challenge is data security. Based on

* *Author for correspondence*

these constraints, this research paper addresses the issues of security on smart phone storage data by using AES to encrypt and decrypt data using cloud storage. Data security whether data at rest or data in transit cannot be today overlooked[6]. Further, numerous threats to mobile phones nowadays have made the issue of security of smart phone storage data very critical. Many organizations also allow their staff to use their mobile devices to carryout organizational businesses and transactions. Hence, the main objective of this work is therefore to enhance smart phone storage data for better end user experience in terms of security of sensitive data.

Smartphone's storage influences the performance of applications in an unexpected ways that are traditionally thought of as network bound. The smart phone storage does not only make the application perform better but also enhances the device energy consumption. But when there is no enough storage space on smart phone, its operation will be slow and will consume more battery. Common experience is that user always deletes some applications in order to create space for new ones. This means that storage plays an important role in smart phone functionalities[7].

The remaining parts of this research paper are arranged as follows; Section 2 discusses existing Android security. Section 3 presented the proposed Solo Encrypt security architecture and its implementation. The proposed work is implemented by developing mobile applications to implement AES algorithm in SoloApp and REST API that the user will interact with in the cloud. The key generation, data encryption and data decryption algorithms are presented. The experimental results comparative performance and evaluation are given in Section 4. Section 5 discusses the outcome of the proposed work. After which the paper concluded in Section 6.

2. Existing Security on Smart phones

There are different versions of Android smart phones in the market. However, from version 4.0, there has been security facility for total disk encryption[8]. The total disk encryption allows the user to encrypt all the data stored on smart phone using the device password for encryption. The encryption process takes time and consumes battery of mobile device. It is to be noted here that once it is initiated for total disk encryption the smart phone becomes unusable to the user. Moreover, encrypting all data on smart phone can reduce system performance, consume already limited resources and increase set up time[9].

Other security mechanisms provided on Android smart phones are the use of passwords, PIN and pattern. All these are less secure, because if a malicious application gains access to the smart phone, the content can be read/altered. This means that user's privacy can be compromised. Passwords, PIN and Pattern are common means of authentication. Passwords can be made up of alphabets, numbers or both. Users always use memorable password, which is vulnerable to dictionary attacks and brute force attacks. A strong password may be difficult to remember by the user[10].

A proposal[2] has been made to secure storage of data on Android based devices. The research work was able to secure local smart phone storage by allowing user to select critical files, data or text and encrypt on their mobile device. This study provided an improved performance than total disk encryption. It uses password based encryption standard to generate key for encryption and decryption on local smart phone storage. The advantage of the proposed work is the ability of the developed application being able to decrypt the encrypted file with ease. Nevertheless, it is to be noted that if encryption is done on local storage only, assuming the smart phone gets lost, the content cannot be read by a third party but the user (owner) too will no longer have access to his/her data. The more reason and very important the use of cloud storage/server for additional benefit of anywhere, anytime access.

A study[11] proposes a lightweight encryption and secure handshaking protocol to protect data on smart phones. It is based on user password and unique identifier of a device. The purpose of the work is to encrypt user data before transferring into cloud. Anytime a user needs to fetch data, authentication will be performed between the mobile device and the cloud provider by means of device information. The work was evaluated on a real mobile device. The method demonstrates user centric encryption authentication for cloud. To judge the performance of the simulated work, the execution time of the algorithm was compared to Data Encryption Standard (DES). It was observed that the longest set up time for each was the limiting factor of the clock cycle time. The study concluded that based on the proposed algorithm, signal set up time is the major challenge faced.

Furthermore, a survey was carried out on Android security framework[12]. It was mentioned that before an application is installed on smart phone, permission is required from the user. The only available option for user of smart phone is to either grant the permission or disallow the application from being installed. Four important issues were raised which are worthy to be noted. They are;

- Permission must be granted before application can be installed.
- Once granted there is no way of restricting the application.
- Access to smart phone resources cannot be limited because permission is based on install time check.
- To uninstall the application is the only way of revoking permissions.

In the light of these security constraints of smart phones, they are not fully secured. There is the need for improving the security of data on smart phones. The survey work concluded by suggesting that application behaviour at runtime should be continuously monitored. In addition based on policy, application should not be allowed to do what it was not granted to do.

Recent study[13] proposes a cloud based system for improving security of smart phones using Modern Encryption Standard (MES) - II algorithm. The developed system secures smart phone data on device prior to cloud transmission. The method adopted was MES-II for encrypting and decrypting of user data using mobile ID. It uses hardware information of mobile. Input data can be in the form of text or pdf file. The input data is encrypted on Android device using MES-II. After the encryption, it is uploaded to cloud using Drop Box. When user needs the cloud data (encrypted data), it is downloaded into user Android device for decryption. The research work concluded that the size of the encrypted text is not varying from the size of the original text.

The advent of Mobile Cloud Computing (MCC) has brought many benefits to mobile devices constraints. There are many hardware and software resources that can be used on demand. While mobile devices are facing many challenges on resources (e.g. storage, battery life and bandwidth), MCC is saying there is a way out. It is a new model for mobile devices where most of the processing and storage of data are moved to powerful computing platforms in the cloud[14]. A review article[15] gives some benefits of using MCC. They are as follows;

- It enhances smart phone battery life. This is achieved by computation offloading because some heavy applications consume battery.
- It enhances data storage capacity. There are numerous cloud servers that provide storage to user due to limited storage space on smart phone.
- There is improved reliability. The cloud providers have several mechanisms for ensuring that user's data in their custody is safe. These data are saved on multiple computers at the cloud provider's side. This is done to improve reliability of their services to their clients.

3. Proposed Security Model/ Implementation

Our proposed security model as shown in Figure 1 is divided into three parts: the user, the service provider and the cloud environment. Each section is explained below.

3.1 User

The user smart phone has a mobile application called Solo App installed into the user's smart phone. The SoloApp is developed using the Android Studio platform to encrypt and decrypt using AES encryption and decryption algorithms. The algorithms are presented below;

Algorithm data Encryption (SK, PT)
Input: Array of Byte of secret key, SK and plain text PT
Output: The output will be Byte of cipher text, CT
Byte ← Byte (SK, PT)
AddRoundKey ← (Byte, w[0, Nb-1])
for i ← 1 to Nr-1 do
SubBytes (Byte)
ShiftRows (Byte)
MixColumns (Byte)
AddRoundKey (Byte, w[i*Nb, (i+1)*Nb-1])
SubBytes (Byte)
ShiftRows (Byte)
AddRoundKey (Byte, w[Nr*Nb, (Nr+1)*Nb-1])
return Byte (CT)

This is an AES algorithm for encrypting data. SK is the secret key, PT is the plain text to be encrypted, Nb is the number of block, Nr is the number of round and w is word which takes in 4 columns Bytes for key expansion. Note that for AES encryption algorithm the last round

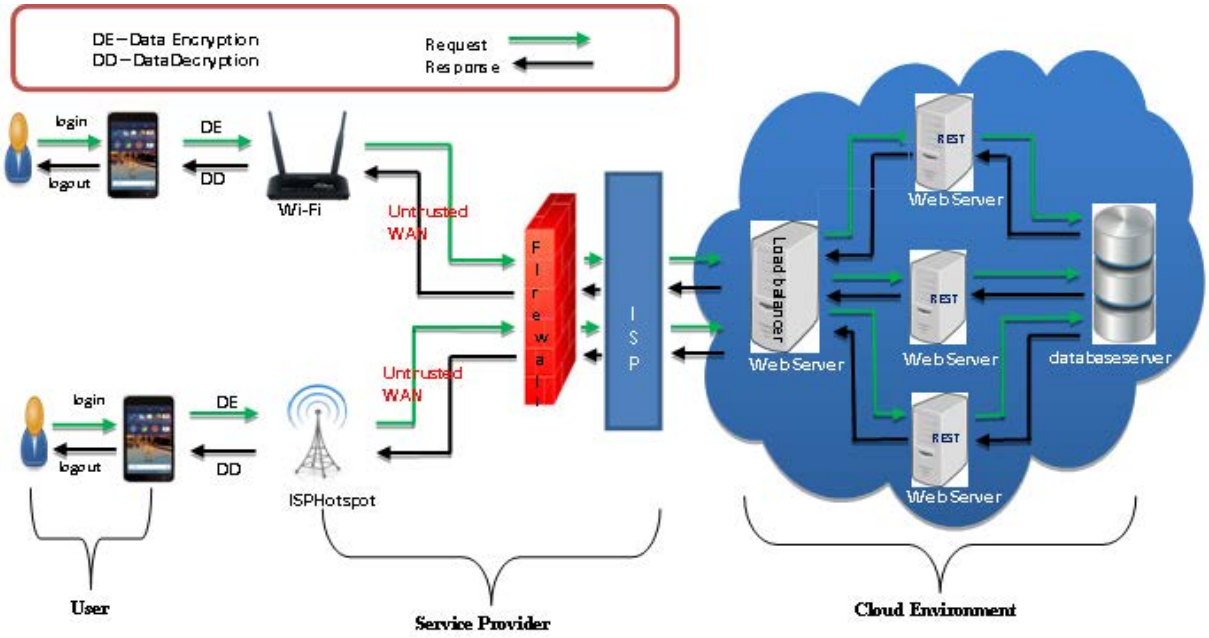

Figure 1. SoloEncrypt security architecture.

does not perform the MixColumns operation. The encrypted files need to be decrypted whenever needed by the user from the cloud server. The decryption is done by using the same secret key that was used for encryption to decryption. Decryption is the inversed of encryption. The algorithm for decryption is stated below;

Algorithm: dataDecryption (SK, CT)
Input: Array of Byte of secret key, SK and cipher text CT
Output: The output will be Byte of plain text, PT
Byte ← Byte (SK, CT)
AddRoundKey ← (Byte, w[0, Nb-1])
for i ← 1 to Nr-1 do
inverse ShiftRows (Byte)
inverse SubBytes (Byte)
AddRoundKey (Byte, w[i*Nb, (i+1)*Nb-1])
inverse MixColumns (Byte)
inverse ShiftRows (Byte)
inverse SubBytes (Byte)
AddRoundKey (Byte, w[Nr*Nb, (Nr+1)*Nb-1])
Return Byte (PT)

Note that just as for encryption algorithm, the AES decryption algorithm last round does not perform the inverse MixColumns operation.

The use of SoloApp mobile application involves user registration. Registration of the user is compulsory and is a one-time activity to register the user's device with cloud server. Launching the application for the first time will enable the user to get the screen shot in Figure 2 which the smart phone user will interact with to register. SoloApp works with Internet connectivity, either using Wi-Fi or 3G/4G of an ISP provider.

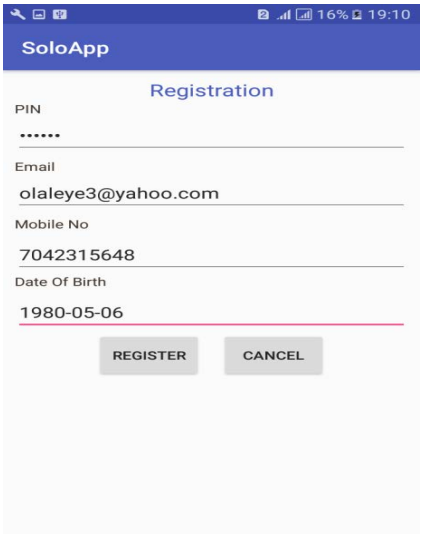

Figure 2. User registration interface.

User is allowed to enter six (6) digits PIN, email id and date of birth. Six digit PIN is used to make the system more secure. ISO 9564-1 allows number of digits to be used in PINs for security authentication to range from 4

to 12 digits. The use of four digits is common. The higher the number the more secure a system and the more difficult to implement. Here, we proposed the use of 6 digits PIN to make our work to be more secured. Once a user presses the register button, the machine/smart phone id, mobile number, PIN number, email id and date of birth are fetched from the user's Smart phone and will be stored in cloud database.

There are facilities to retrieve PIN by the user if he/she forgets his/her PIN as shown in Figure 3a and Figure 3b. The user is requested to enter his/her email id and date of birth in order to be able to retrieve an auto generated PIN which is sent to user email id. The SoloApp allows the user to select file to be encrypted in Figure 4a. It allows user to view encrypted file names before decryption in Figure 4b.

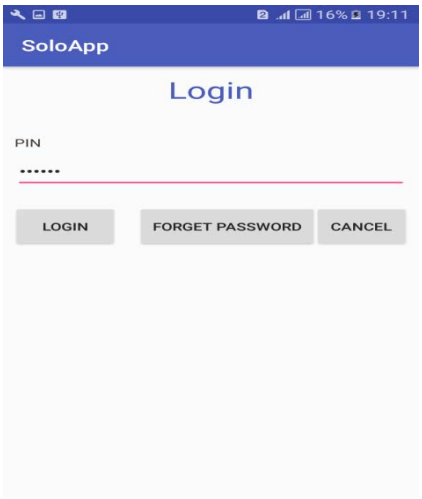

Figure 3a. To retrieve PIN.

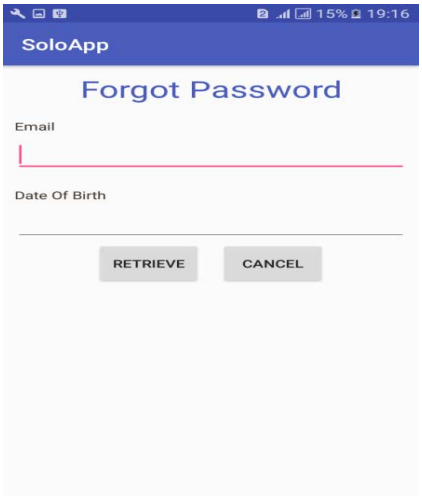

Figure 3b. To retrieve PIN.

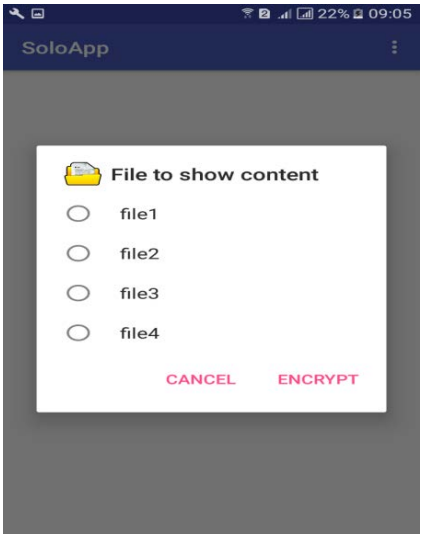

Figure 4a. Select file to encrypt.

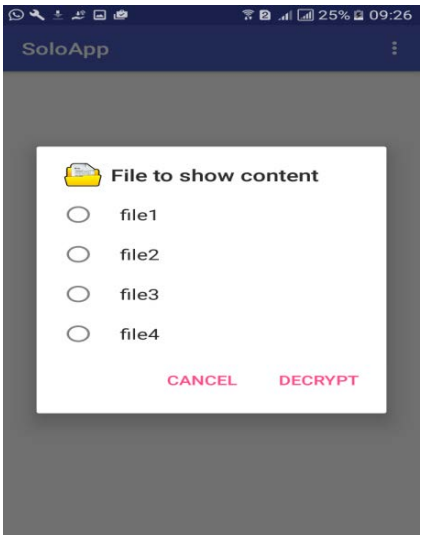

Figure 4b. Select file to decrypt.

3.2 Service Provider

The service provider provides Internet connectivity. The application cannot work without Internet connectivity. User can have access to cloud server either through Wi-Fi connectivity or through the ISP hotspot.

3.3 Cloud Environment

In the cloud environment, REST is deployed as web application that the smart phone users interact with. RESTful web service architecture is used because it is faster, flexible, light weight and uses less bandwidth because messages are passed in Java Script Object Notation (JSON). REST uses HTTP standard for security. It permits other data format apart from JSON, such as

XML, Plain Text, HTML etc. while SOAP permits only XML data format. More importantly, REST is easy to build and implement[16]. Cloud providers today allow cloud users to deploy their own API and pay for services as per their billing requirements. Interaction of user using the SoloApp is always online. The Solo Encrypt security architecture protects data at rest as well in transit. AES algorithm was used and was approved by the National Institute of Standards and Technology (NIST) for protecting data and is secured against brute force attack and dictionary attack.

4. Experimental Results and Evaluation

4.1 Evaluation Parameters

To evaluate the performance of our SoloEncrypt security architecture using our SoloApp, three Android devices of varied configurations are used. They are Samsung Galaxy J5, Xiaomi Redmi Note 3 and Micromax Canvas 6. Detail configurations of the devices are as shown in Table 1. The evaluation parameters used are encryption time, decryption time and throughput which are among NIST- National Institute of Standards and Technology standardised metrics for evaluating an encryption algorithm. All experimental results are presented in tables and charts for clarity.

- Encryption Time: This is the time in milliseconds (ms) that takes an encryption algorithm to convert plaintext to encrypted data (cipher text). It shows the speed of the encryption process[17].
- Decryption Time: The time it takes to regain plaintext from its encrypted form is known as decryption time. The decryption time is expected to be lower than encryption time[18].
- Throughput: The throughput is computed for both encryption and decryption algorithm. The formula for calculating throughput is given thus;

Encryption Throughput = ∑file size (KB)/ ∑ET(s)

Decryption Throughput = ∑file size (KB)/ ∑DT(s)

Where ET is the encryption time and DT is the decryption time.

4.2 Experimental Results

The experimental results are based on each device as recorded in Table 2, Table 3 and Table 4. Actually each file size is repeated 5 times and the average time is recorded against the file size for each encryption and decryption.

Computation of Throughput for each of the Devices

- Samsung Galaxy J5

Encryption Throughput= ∑file size (KB)/ ∑ET(s)
∑file size (KB) =11,500 KB
∑ET(s) = 6.035s
Encryption Throughput=1,905.551 KB/s
Decryption Throughput = ∑file size (KB)/ ∑DT(s)
∑file size (KB) =11,500 KB
∑DT(s) = 5.972s
Decryption Throughput=1,925.653 KB/s

- Xiaomi Redmi Note 3

∑file size (KB) =11,500 KB
∑ET(s) = 5.223s
Encryption Throughput= 2,201.800 KB/s
∑file size (KB) =11,500 KB
∑DT(s) = 5.158s
Decryption Throughput= 2,229.546 KB/s

- Micromax Canvas 6

∑file size (KB) =11,500 KB
∑ET(s) = 2.536s
Encryption Throughput= 4,534.700 KB/s
∑file size (KB) =11,500 KB
∑DT(s) = 2.419s
Decryption Throughput= 4,754.031 KB/s

Table 1. Configuration of devices

	Mobile Devices		
	Samsung Galaxy J5	**Xiaomi Redmi Note 3**	**Micromax Canvas 6**
OS	Android v 6.0.1 (Marshmallow)	Android v5.1.1 (Lollipop)	Android v5.1 (Lollipop)
CPU	Quad-core 1.2 GHz	Hexa-core - 2x1.8 GHz	Octa-core 2.0 GHz
RAM	2 GB	2 GB	3 GB
Internal Storage	16 GB	16 GB	32 GB

Table 2. Encryption and decryption time for Samsung Galaxy J5 (2016)

File Size (MB)	Encryption Time(ms)	Decryption Time(ms)
0.5	294	290
1.0	583	603
1.5	852	832
2.0	1067	1068
2.5	1365	1323
4.0	1874	1856

Table 3. Encryption and decryption time for Xiaomi Redmi Note 3

File Size (MB)	Encryption Time(ms)	Decryption Time(ms)
0.5	253	241
1.0	537	542
1.5	813	786
2.0	893	901
2.5	1104	1094
4.0	1623	1594

Table 4. Encryption and decryption time for Micromax Canvas 6

File Size (MB)	Encryption Time(ms)	Decryption Time(ms)
0.5	130	95
1.0	223	280
1.5	385	342
2.0	476	421
2.5	594	603
4.0	728	678

5. Discussion

Smart phones are widely used computing devices. They are used for business transactions and for personal use. Sensitive data are stored on them. They have limitations among which one is storage space. Our interest is to secure smart phones storage data by using cloud infrastructure. We therefore designed and implemented a secured security architecture named Solo Encrypt security architecture. It was implemented using a mobile application named Solo App that was developed using Android Studio. Solo App uses AES encryption algorithm of 256 bit key length. The NIST in America approved AES usage for protecting user data on mobile devices. The application interacts with php code that runs on server side and REST API for interface which is deployed into the cloud.

We further present our experimental results using three mobile devices Table 1. Encryption time, decryption time and throughput are the evaluation parameters used to evaluate our work. Our experimental results showed that the algorithm encrypts with better speed and it takes less time to decrypt compared with the encryption time in some cases. For Samsung Galaxy J5 (2016), it takes 1874 Milliseconds (ms) to encrypt 4 MB of data while it takes 1856 ms to decrypt the same data. Using Xiaomi Redmi Note 3, it takes 1623 ms to encrypt 4 MB of data and takes 1594 ms to decrypt the same data. For Micromax Canvas 6, it takes 728 ms to encrypt 4 MB of data and 678 ms to decrypt. The differences in encryption and decryption times were based on the differences in each device's configuration. Hence, the encryption time and decryption time for each file size are found reasonable. For the Throughput computation the MB file sizes were converted in KB and Milliseconds converted into seconds. The rise in the Throughput results mean a reduction in smart phone power consumption, this indicates that the more throughput the better the device performance[18,19].

Our major constraint is Internet connectivity. The application cannot work without Internet. For better end user experience there must be network connectivity either 3G/4G network or the use of Wi-Fi from the launch of the application.

6. Conclusion and Future Work

This research work proposed SoloEncrypt security architecture for protecting user sensitive data on smartphones by using cloud infrastructure. The proposed work is user friendly and well secured. The results were found impressive with the encryption and decryption speed. The proposed work was implemented using AES 256 bit key length for the encryption and decryption. Due to resources constraints on smartphones, we shall in our next work implement AES 128 bit key length such that security is not compromised. Other encryption algorithms such as Blowfish, DES, 3DES etc. can be further used to implement the security architecture and the work be evaluated based on encryption and decryption time, Avalanche effect, entropy and more Android testing devices.

7. Acknowledgement

The authors would like to thank Prof. N. B. Singh and Prof. Rajesh Kumar of Research and Technology Development Centre of Sharda University for their unflinching supports and guidance.

8. References

1. Jeong YS, Kim HW, Park JH. An effective locking scheme of smart multimedia devices with convenience and enhanced security. Multimedia Tools Application Springer Science. 2014 Jul; 1-13.
2. Poonguzhali P, Dhanokar P, Chaithanya MK, Patil MU. Secure storage of data on Android based devices. International Journal of Engineering and Technology. 2016 Jun; 8(3):177-82. Crossref
3. Nayadkar PP, Parne BL. A survey on different backup and restore techniques used in mobile devices. International Journal of Computer Science and Information Technologies. 2014; 5(6):8236-8.
4. Nayadkar PP. Automatic and secured backup and restore technique in android. IEEE International Conference on Innovations in Information Embedded and Communication Systems; 2015 Mar. Crossref
5. Donald A C, Oli SA, Arockiam L. Mobile cloud security issues and challenges: A perspective. IJEIT. 2013 Jul; 3(1):401-6.
6. Alomari MA, Samsudin K, Ramli AR. Implementation of a parallel XTS encryption mode of operation. Indian Journal of Science and Technology. 2014 Nov; 7(11):1813-9.
7. Kim H, Agrawal N, Ungureanu C. Revisiting storage for smart phones. ACM Transactions on Storage (TOS). 2012; 8(4):14. Crossref
8. Gotzfried J, Muller T. Analysing Android's full disk encryption feature. Journal of Wireless Mobile Networks, Ubiquitous Computing and Dependable Applications. 2014.
9. Wang Z, Murmuria R, Stavrou A. Implementing and optimizing an encryption file system on Android. Mobile Data Management (MDM). 2012 Jul.
10. Ray PP. Ray's scheme: Graphical password based hybrid authentication system for smart hand held devices. Journal of Information Engineering and Applications. 2012; 2(2):1-12.
11. Zegers W, Chang S, Park Y, Gao YJ. A lightweight encryption and secure protocol for smart phone cloud. IEEE Conference on Service-Oriented Engineering (SOSE); 2015 Jun. p. 259-66.
12. Powar S, Meshram B B. Survey on Android security framework. IJERA. 2013 Mar-Apr; 3(2):907-11.
13. Agrawal DD, Kulurkar P. A cloud-based system for enhancing security of android devices using modern encryption standard-II algorithm. International Journal of Innovations and Advancement in Computer Science. 2016 Apr; 5(4):60-6.
14. Vittapu MS, Sunkari V, Abate AY, Sreenivas N. A proposed solution to secure MCC uprising issue and challenges in the domain of cyber security. Open Journal of Mobile Computing and Cloud Computing. 2015 Feb; 2(1):19-33.
15. Mandavkar P, Patil G, Shetty C, Parkar V. SMS security for android mobile using combines cryptographic algorithms. International Journal of Advanced Research in Computer and Communication Engineering. 2014 Apr; 3(4):6221-5.
16. Kumari V. Web services protocol: SOAP vs. REST. IJARCET. 2015 May; 4(5):2467–9.
17. Singh M, Dhindsa KS. Securing RJSON data between middleware and smart phones algorithms. IJSCE. 2013 May; 3(2):189-94.
18. Patil P, Narayankar P, Narayan DG, Meena SM. A comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and Blowfish. Elsevier B. V. Procedia Computer Science. 2016; 78:617-24. Crossref
19. Arora R, Sharma S. Performance analysis of cryptography algorithms. International Journal of Computer Applications. 2012 Jun; 48(21):35–9. Crossref