Multi-Key Privacy in Cloud Computing

Ankit Shetty[1], Rachana Vannal[2], Uzair Pawaskar[3], K. Suresh Babu[4]

[1,2,3]BE Student, Department of Information Technology, MES Pillai College of Engineering , Navi Mumbai, India
[4]Assistant Professor, Department of Information Technology, MES Pillai College of Engineering, Navi Mumbai, India

---***---

Abstract— *The project helps to have a secure data exchange to and from the cloud through and android application. Integrated cryptographic techniques such as Advanced Encryption Standard (AES), Rivest–Shamir–Adleman (RSA), and Hash-based Message Authentication Code (HMAC) are used to encrypt the data. Android as a User Interface and Firebase cloud services are used to store data on the cloud.*

Keywords— Cloud Computing, Android, AES, RSA, HMAC.

1. INTRODUCTION

The proposed system helps users to find information by providing an easy login into the system to upload the files in the device/ retrieve the uploaded files. The data files in the cloud would be first encrypted using encryption algorithms i.e. security algorithms and then be stored on the cloud. The user can again login into the system after a while and again access the files by decrypting the selected files i.e. file of the user's choice. For decrypting the user will be provided with a private security key which would help the decryption of files. The secret key is used in order to provide the authentication, authorization and integrity of the saved data.

1.1 TECHNIQUES USED

A. Advanced Encryption Standard (AES) Algorithm

It is one of the most common and widely used symmetric block cipher algorithms worldwide. This algorithm has its own particular structure to encrypt and decrypt sensitive data and is applied in hardware and software all over the world. It is extremely difficult for hackers to get the real data when encrypting by AES algorithm.

Till date there is no evidence about the crack of this algorithm. AES has the ability to deal with three different key sizes such as AES 128, 192 and 256 bit and each of this ciphers has 128 bit block size. The Advanced Encryption Standard (AES) algorithm is one of the block cipher encryption algorithms that was published by the National Institute of Standards and technology (NIST) in 2000. The main aim of this algorithm was to replace the DES algorithm after some vulnerable aspects appeared.

01. Key generation

Our proposal is based mainly upon the following; a key generator which is able to generate new keys depending upon three parts (M1, M2 and S) and a set of operations. Results are merged to check the size of output equal to the size of the plain text. In case the size of the output is equal to the size of the plain text, therefore, the merged key can be entered as a secret key.

The below algorithm illustrates the steps of the key generation. AES-128 bits of Encryption and Decryption We describe in detail the rounds of Advanced Encryption Standard (AES-128) Algorithms in order to encrypt files or documents and each round consist of four sub-processes as follows.

Procedure: Encryption and Decryption

a. Sub Bytes:

Byte Substitution is a nonlinear byte substitution that operates independently on each byte by looking up on a fixed table is called (S-box) and the result of the 16 input bytes is a matrix of as well as four columns as well as four rows.

b. Shift Rows:

It is shifted to the left for each of the four rows of the matrix and transformation operates on the rows, it cyclically shifts on bytes in each row and bytes that fall off of the row are reinserted on the right side of the same row. The shift is carried out as follows:

- Unchanged of The first row.
- Shifted one byte (one position) to the left of the second row.

- Shifted two byte (two positions) to the left of the third row.
- Shifted three byte (three positions) to the left of the fourth row.
- The output is a novel matrix containing the same byte but shifted.

c. Mix Columns:

Mix Columns is the transformed process of all the columns that contain four bytes by means of certain mathematical functions. Such a function takes for each column four bytes as input and the result is four totally novel bytes, and novel bytes is replace with original bytes .The result is a novel matrix containing 16 new bytes. This step is not executed in the final round.

d. Add round key:

In this Add Round Key operation which executed (XOR) operation to the 128 bits of the round key. If this is the final round then the result is encrypted documents or files. Executed XOR operations on results from mix column and round keys. For AES 128,128 bit XOR operations are executed. Otherwise, the resulting 128 bits are interpreted as 16 bytes and then we begin another similar round.

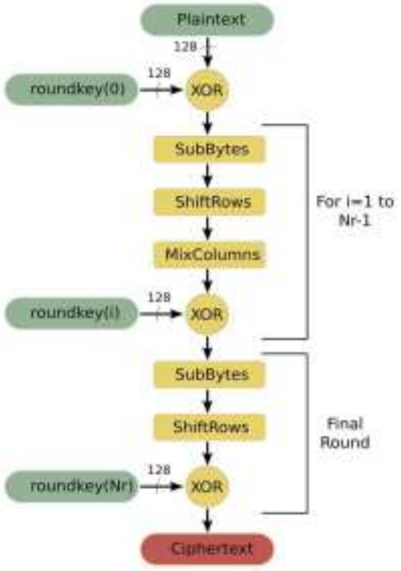

Fig 1. AES ALGORITHM.

The above figure depicts the process of AES Algorithm with the elements involved in the process and the rounds needed in the algorithm.

B. RSA (Rivest–Shamir–Adleman) Algorithm

It is one of the first public-key cryptosystems and is widely used for secure data transmission. In such a cryptosystem, the encryption key is public and it is different from the decryption key which is kept secret (private). In RSA, this asymmetry is based on the practical difficulty of the factorization of the product of two large prime numbers, the "factoring problem". The acronym RSA is made of the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who first publicly described the algorithm in 1977.

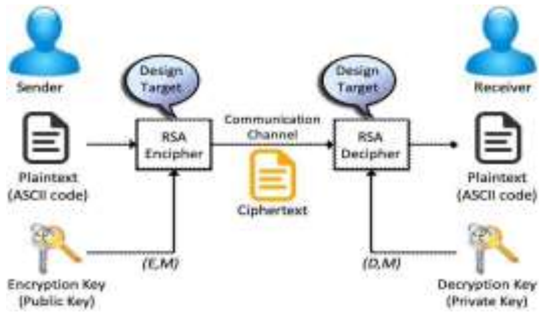

Fig 2. RSA ENCRYPTION PROCESS OF A TEXT.

- Key Generation:

The keys for the RSA algorithm are generated in the following way:

1. Choose two distinct prime numbers a and b.
    - For security purposes, the integers a and b should be chosen at random, and should be similar in magnitude but differ in length by a few digits to make factoring harder. Prime integers can be efficiently found using a primality test.
    - a and b are kept secret.
2. Compute $n = ab$.
    - n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
    - n is released as part of the public key.
3. Compute $\lambda(n)$, where λ is Carmichael's totient function. Since $n = ab$, $\lambda(n) = \mathrm{lcm}(\lambda(a),\lambda(b))$, and since a and b are prime,

$\lambda(a) = \varphi(a) = a - 1$ and likewise $\lambda(a) = b - 1$. Hence $\lambda(n) = \mathrm{lcm}(a - 1, b - 1)$.

- $\lambda(n)$ is kept secret.

4. Choose an integer e such that $1 < e < \lambda(n)$ and $\gcd(e, \lambda(n)) = 1$; that is, e and $\lambda(n)$ are coprime.
    - e having a short bit-length and small Hamming weight results in more efficient encryption – the most commonly chosen value for e is $2^{16} + 1 = 65{,}537$. The smallest (and fastest) possible value for e is 3, but such a small value for e has been shown to be less secure in some settings.
    - e is released as part of the public key.

5. Determine d as $d \equiv e^{-1} \pmod{\lambda(n)}$; that is, d is the modular multiplicative inverse of e modulo $\lambda(n)$.
    - This means: solve for d the equation $d \cdot e \equiv 1 \pmod{\lambda(n)}$. d can be computed efficiently by using the Extended Euclidean algorithm.

d is kept secret as the private key exponent.

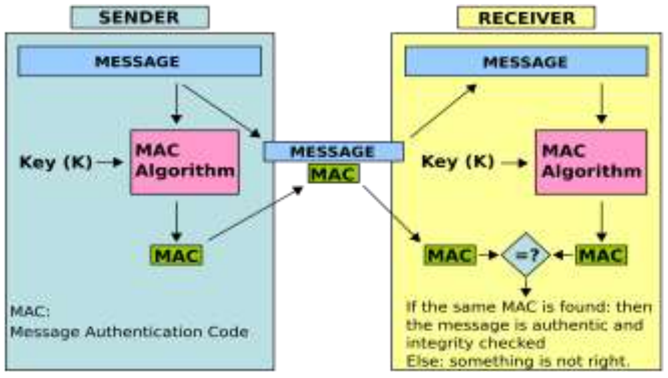
SENDER
RECEIVER
MESSAGE
Key (K)
MAC Algorithm
MAC
MESSAGE
MAC
MESSAGE
Key (K)
MAC Algorithm
MAC
=?
MAC
MAC: Message Authentication Code
If the same MAC is found: then the message is authentic and integrity checked
Else: something is not right.

Fig 3. RSA PROCESS

C. Hash-Based Message Authentication Code (HMAC) Algorithm

In cryptography, an HMAC (sometimes expanded as either keyed-hash message authentication code or hash-based message authentication code) is a specific type of message authentication code (MAC) involving a cryptographic hash function and a secret cryptographic key. It may be used to simultaneously verify both the data integrity and the authenticity of a given message, as with any MAC. Any cryptographic hash function, such as SHA-256 or SHA-3, may be used in the calculation of an HMAC; the resulting MAC algorithm is termed HMAC-X, where X is the hash function used (e.g. HMAC-SHA256 or HMAC-SHA3).

The cryptographic strength of the HMAC depends upon the cryptographic strength of the underlying hash function, the size of its hash output, and the size and quality of the key.

HMAC does not encrypt the message. Instead, the message (encrypted or not) must be sent alongside the HMAC hash. Parties with the secret key will hash the message again themselves, and if it is authentic, the received and computed hashes will match.

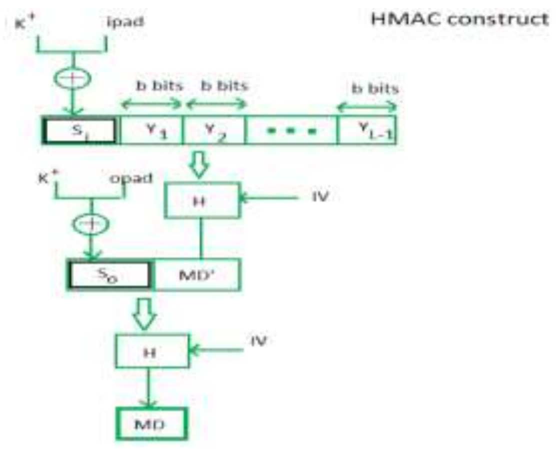

Fig 4. HMAC CONSTRUCT DIAGRAM

2. PROPOSED SYSTEM:

2.1 General Description of The Application

PGPR-RootExplorer application is an application that applies the concept of digital signatures using RSA and SHA-256 algorithms and AES-256 block cipher algorithm for the encryption process.

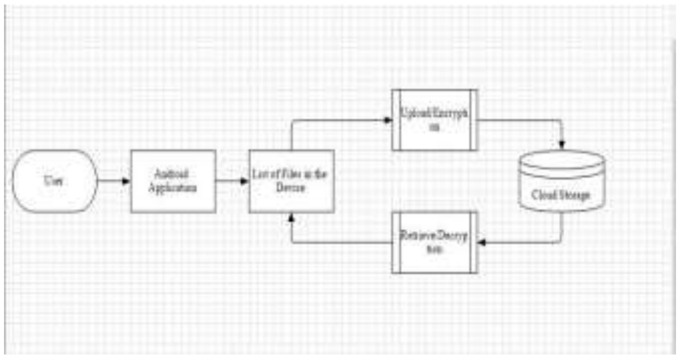

Fig 5. BLOCK DIAGRAM

The application is implemented in Android Java programming that guarantees the integrity, confidentiality, authentication and non-repudiation. Java language is used because it is more mobile, object-oriented, portable, and open source. It has two schemes which are the protection scheme and verification scheme. The protection step is on the 4 digit passcode to open the application and ENCRYPTION and the DECRYPTION process is on the main page.

The details of the two processes are of the following:

1 .Signing and Passcode: On the protection schemes , two processes are running: the securing process (encryption) and the authenticating process (signing). The generation of the private key and the public key is done before the encryption and signing process is executed. The scheme of securing and authenticating documents is done in a simple manner following the scheme in Fig. 6 & 7.

2) Encryption Data: The encryption process is done by selecting the data which we want to encrypt and then add a key to it. (see Fig. 11). The file is firstly encrypted using a key that has been used previously in the encryption process. After this process, the file is hashed and then digital signature calculations using a public key that has been generated and stored in the protection stage is performed. The verification process is the process of calculating the digital signature value of the hashed document using the public key. If appropriate, it will display a notification that the document is successfully decrypted and proven to be authentic. If it does not match, the notification will show that the document was not authentic or has been a change.

2.2 Implementation of PGP-RootExplorer

The implementation of File Encryption includes steps using PGP application. The steps in the implementation of PGP application are as follows:

1) Login process: When the user runs the application, a welcome message will appear to start logging the process. Then the user needs to fill the USERNAME and PASSWORD fields. By pressing the LET'S GO button, the login process begins.

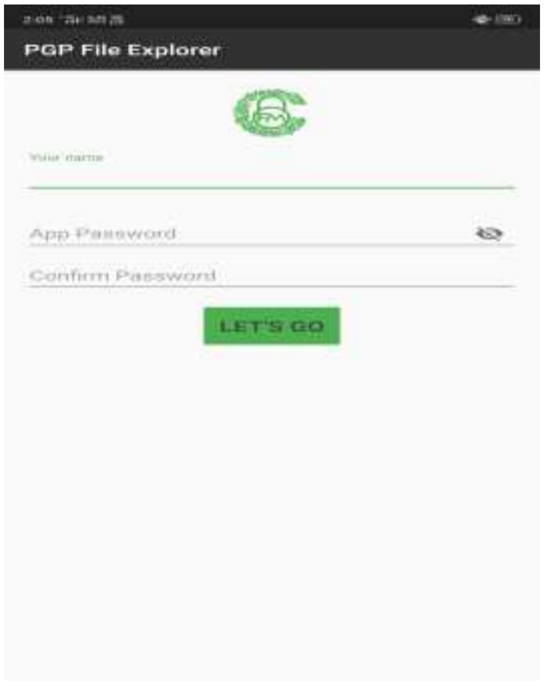

Fig 6. LOGIN PROCESS

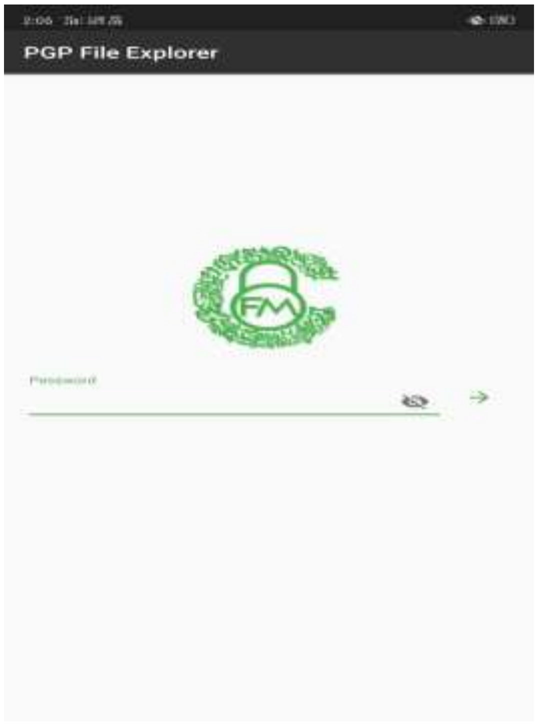

Fig 7. LOGIN PAGE

2) Main Screen Contents: When the user successfully logs into the application all the contents available in the device will be shown similar to the file manager, the user just needs to select the files which are going to be encrypted or decrypted once this is done.

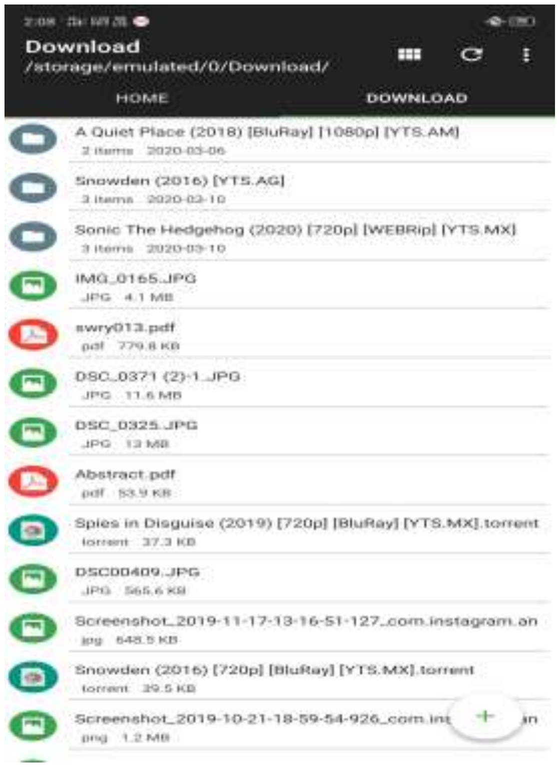
Download
/storage/emulated/0/Download/
HOME DOWNLOAD
A Quiet Place (2018) [BluRay] [1080p] [YTS.AM]
2 items 2020-03-06
Snowden (2016) [YTS.AG]
3 items 2020-03-10
Sonic The Hedgehog (2020) [720p] [WEBRip] [YTS.MX]
3 items 2020-03-10
IMG_0165.JPG
JPG 4.1 MB
swry013.pdf
pdf 779.8 KB
DSC_0371 (2)-1.JPG
JPG 11.6 MB
DSC_0325.JPG
JPG 13 MB
Abstract.pdf
pdf 53.9 KB
Spies in Disguise (2019) [720p] [BluRay] [YTS.MX].torrent
torrent 37.3 KB
DSC00409.JPG
JPG 565.6 KB
Screenshot_2019-11-17-13-16-51-127_com.instagram.an
jpg 648.5 KB
Snowden (2016) [720p] [BluRay] [YTS.MX].torrent
torrent 39.5 KB
Screenshot_2019-10-21-18-59-54-926_com.in
png 1.2 MB

Fig 8. FILES ON THE DEVICE

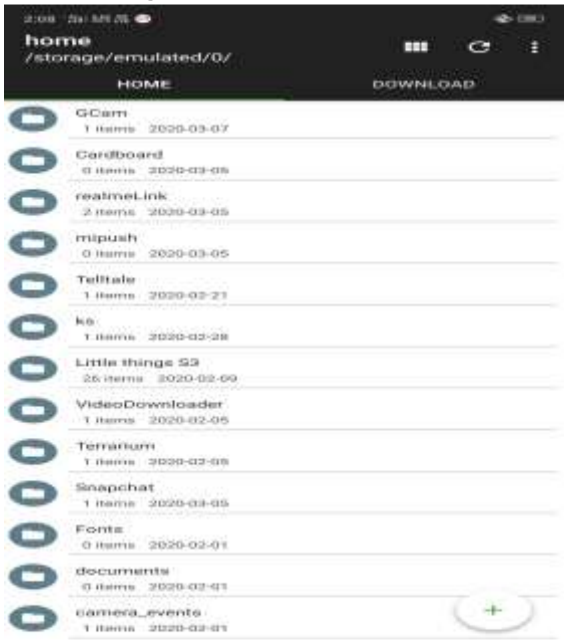
home
/storage/emulated/0/
HOME DOWNLOAD
GCam
1 items 2020-03-07
Cardboard
0 items 2020-03-05
realmeLink
2 items 2020-03-05
mipush
0 items 2020-03-05
Telltale
1 items 2020-02-21
ks
1 items 2020-02-28
Little things S3
26 items 2020-02-09
VideoDownloader
1 items 2020-02-05
Terrarium
1 items 2020-02-05
Snapchat
1 items 2020-03-05
Fonts
0 items 2020-02-01
documents
0 items 2020-02-01
camera_events
1 items 2020-02-01

Fig 9. MAIN SCREEN CONTENTS

3) Set the Key: Once the files have been selected the password key will be given to the user and that key will be used to encrypt every file even after a wholesome amount of time. Once the key is set the data will be encrypted using that key and the same key can be used for the decryption process too.

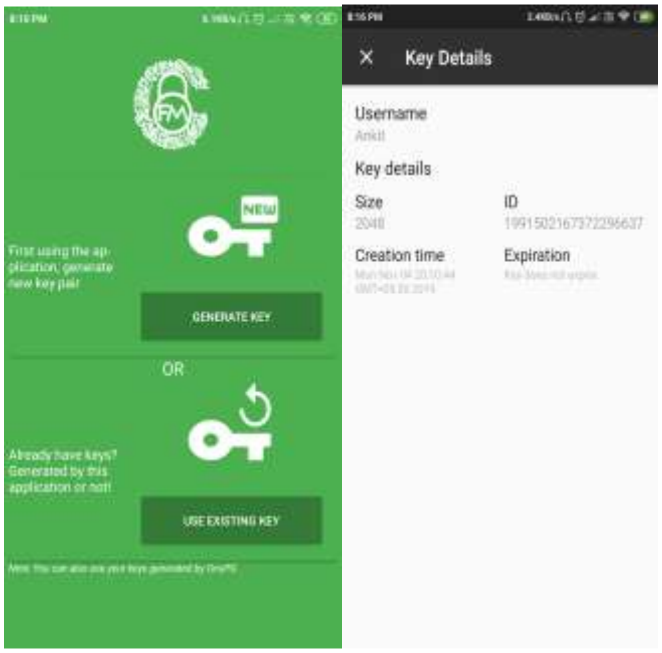
First using the application, generate new key pair
Already have keys? Generated by this application or not!
Key Details
Username
Ankit
Key details
Size 2048
ID 1991502167372296637
Creation time
Expiration

Fig 10. KEY DETAILS

4) Upload data to cloud: Uploading files to cloud will give a data backup solution to the user if something goes sideways.

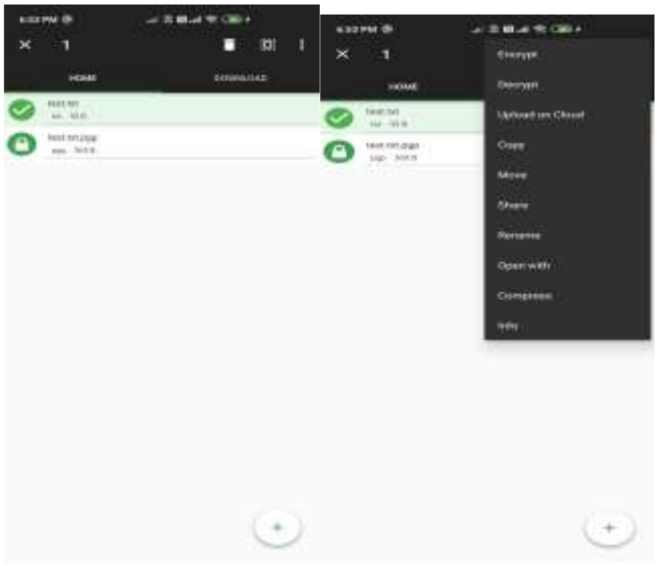

Fig 11. ENCRYPT, DECRYPT AND UPLOAD OPTIONS

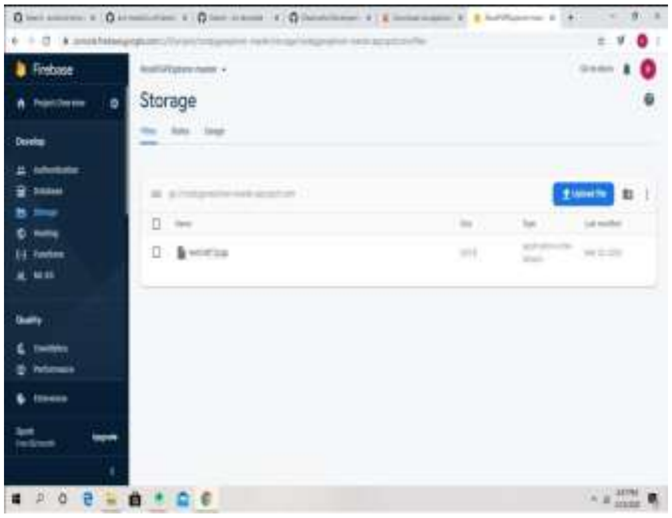

Fig 12. DATA UPLOADED TO FIREBASE

The project uses Firebase as a cloud storage platform as it is convenient to use for android based projects. It has a user friendly interface and is supported by Google Cloud, Firebase Storage offers you almost endless storage space for storing your data.

2.3 Actors Present in Proposed system:

1) User: The user will be the person who uses this application to encrypt data or send Encrypted data via the main process.
2) Database: To store or retrieve files that are Encrypted or decrypted
3) Cloud: To securely save a backup of the files.
4) Receiver: The person who will receive Encrypted files and key and will decrypt by doing the same process.

The design is made user-friendly so that any person who is not a developer can use this application.

3. ACKNOWLEDGMENT

We would especially like to thank our Guide Prof. K. Suresh Babu who encouraged us as well as helped us in various activities.

We give a special gratitude to our H.O.D of Information Technology Department Dr. Satish Kumar L. Varma for this opportunity and motivate us to do innovative things that will be beneficial for our future.

We are thankful to all those who provided us with the opportunity to complete this project. We would like to thank our Principal Dr. Sandeep Joshi who always encouraged us and motivated us.

4. CONCLUSION

In this project a combination of two types of encryption algorithms and one authentication algorithm has been used in order to build a high security system. Using all the three algorithms in the android application one can use this system as a secure gateway for storing and sharing files through cloud or using local storage. This system uses these three algorithms to perfectly utilize each other's advantages to counteract their own weaknesses.

5. References

1. Harba Eman Salim Ibrahim, "Secure Data Encryption by Combination AES, RSA and HMAC", 2018

2. Faheem Gul, Aaqib Amin, and Suhail Ashraf, "Enhancement of cloud computing security with secure data storage using AES" , July 2017

3. Khnd Sri Sandhya, K. Venkat Rao,"Privacy-Preserving and Dynamic Multikey Generation over Encrypted Cloud Data",October 2016

4. Pathak Namita N, and Meghana Nagori, "Enhanced security for multi cloud storage using AES algorithm", 2015

5. Preetha M, and M. Nithya, "A study and Performance Analysis of RSA Algorithm." June 2013