INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X

Impact factor: 4.295

(Volume3, Issue3)

Available online at www.ijariit.com

Application of Email Client with Rabbit Algorithm for Android Mobile

Jadhav Vivek Vilas
Pune University
vivekjadhav012@gmail.com

Jagtap Mayur Sampat
Pune University
mayurjagtap.mj123@gmail.com

Sane Hanumant Raghuji
Pune University
sane.hanumant1995@gmail.com

Abstract: To propose an android application which uses Rabbit Algorithm to develop Email Client Application. This particular email client uses rabbit Algorithm for encrypting and decrypting confidential e-mail content. This application can be used to create, edit, send, retrieve and Read a simple email. The email client is designed for smart Android Using Java. Experiments show that The Google Mail service this app can be used as simple email Client with the secure content feature.

Keywords: Android Mobile, Email, Rabbit Algorithm, Email Client.

1. INTRODUCTION

Email as one of many important media Contains confidential content that must be Insured One of the many methods to ensure the secrecy of content is by using encryption. The proposed algorithm is Rabbit, which is a Symmetric algorithm and symmetric algorithm is faster than asymmetric. The e-mail service today is common, it is accessed through the use of smartphones. One of the most the phone's operating system is Android. To maintain Email secret sent from Android smartphone, encrypted and the decryption process must be planted in an email client to Android smartphone. In this work, we present a client application for encryption Email on Android smartphone using Rabbit Algorithm. This is designed to be able to encrypt email content, Send emails, receive emails, read email and decrypt email content. There are many things to consider in manufacturing this application, such as integration of the Rabbit algorithm with the email client, method to create encrypted content and Send it by email, method to retrieve email from email Service provider, and read an email the encrypted content

2. LITERATURES STUDY

A. Android System

Android is developed by Google which is currently running in smartphones. Android was built on the Linux platform and Developed using Java. Android has already been optimized

To be used in the mobile environment. Android is now known as a popular smartphone operating system Android is an open source operating system, so Many developers who pay attention, explore and fix Android Security risk. In other words, Android is secured. Developing third-party applications on Android can Using Java, C or C ++. Using C or C ++ does not improve application performance, but you can ensure that the use of these Languages can make source code more complicated.

B. Email

Email is a method for exchanging messages over the Internet. Connection. E-mail currently uses MIME as its format. E-mail Transaction uses the method of storage and shipping. This method Allows people to send an email and receive an email later without having themselves online the moment an email arrives.

There are defined protocols for email transactions such as POP3 and IMAP are set to retrieve Internet e-mail And SMTP is defined for email delivery. These protocols may be modified with SSL with the aim of making them more insurance. The IMAP protocol allows the e-mail user to manipulate their Email from any device without worrying about your email will be deleted after being read. POP3 makes the opposites. POP3 allows User to download their email from the Internet, but this protocol deleted your email so that the same email is not downloaded next time.

C. Rabbit Algorithm

Rabbit algorithm was created by Fast Software Encryption in 2003 with Martin Boesgaard, Mette Vesterager, Jesper Christiansen, and Ove Scavenius. Rabbit uses 128 bit to encrypt or decrypt 128 bit of plaintext or ciphertext. Rabbit has 17 variables: eight inner states, eight counters, and one carry. There are three schemes of Rabbit: 1. key setup scheme, 2. next state function scheme, 3.extraction scheme. Key setup scheme is used for initiating value of variables. Next state function is used for updating variables value. Extraction scheme is used to create a 128-bit pseudo random string. Cipher text is created by using XOR operation between 128-bit plaintext with 128-bit pseudo random string.

3. PROPOSED SYSTEM ANALYSIS AND SOLUTION

There are some details that should be thought out and designed to make this email client including language development, Encryption and decryption process, e-mail delivery and retrieval, and the elaboration and reading of contents.

A. Developing Language

To develop an Android email client,

Three languages to choose from. Java is seen as the most Proper language since Java has many free libraries that Support and simplify the implementation of the application. In this issue

Java has JavaMail library that focuses on email.

B. Rabbit Design

Rabbit encrypts 128-bit data or plain text with a 128-bit key. Yes, the Plain text is more than 128 bits in length, so plain text must be divided into several parts. These parts can be written as B 1, B 2, Bn-1, and Bn. If the last part of the plaintext, B n, is less than 128 this part will be added with empty spaces until this part has a length of 128 bits. The process of the Rabbit algorithm can be explained shortly as All variables are set to their initial value based on the key Configuration schema; All variables are updated using the following state Function scheme; Extraction creates 128 bits Pseudo-random chain, P n, of variable values; This 128 bit The pseudo-random chain is paired with B n for the XOR Creating C n as the ciphertext. These steps imply that the next state Function scheme and extraction scheme will be called for n + 4 And n times respectively because the next state function is also Called in key configuration scheme for four times. Figure 1. Encryption flow the ciphertext has 128 bits or 16 bytes in length. If the ciphertext it is presented as a set of characters, then there will be 16 Characters. The problem with this is if 8 bits of ciphertext is Translated to a character, then there is no guarantee that the corresponding character is readable, ie "00000000" means "Null", so it will not be visible as character and "00001010" It means "new line" and this will not be visible either. The Proposed solution for this matter is by means of the ciphertext representation as a set of hexadecimal characters. Every 8 bits will be translated into two hexadecimal characters. By doing this, "0000000" It will be written as "00" and "00001010" will be converted to "0A". From Bits or a byte will result in 2 hexadecimal Characters, 128-bit ciphertext will translate to 32 Text of the hexadecimal characters. Therefore, the length of the ciphertext. It will be twice as long as plain text. Encryption flow it can be seen in Figure 1.The decryption is designed to be slightly different. Since each Character in plaintexts transforms into two hexadecimal Characters in the ciphertext, ciphertext should be read by two Characters. If the ciphertext has more than 32 characters, the text should be divided into parts. Each part contains only 32 Hexadecimal characters. Each part is transformed into 128-bit ciphertext. The rest of the decryption process is the same as Encryption process. The decryption steps can be seen in Figure 2.

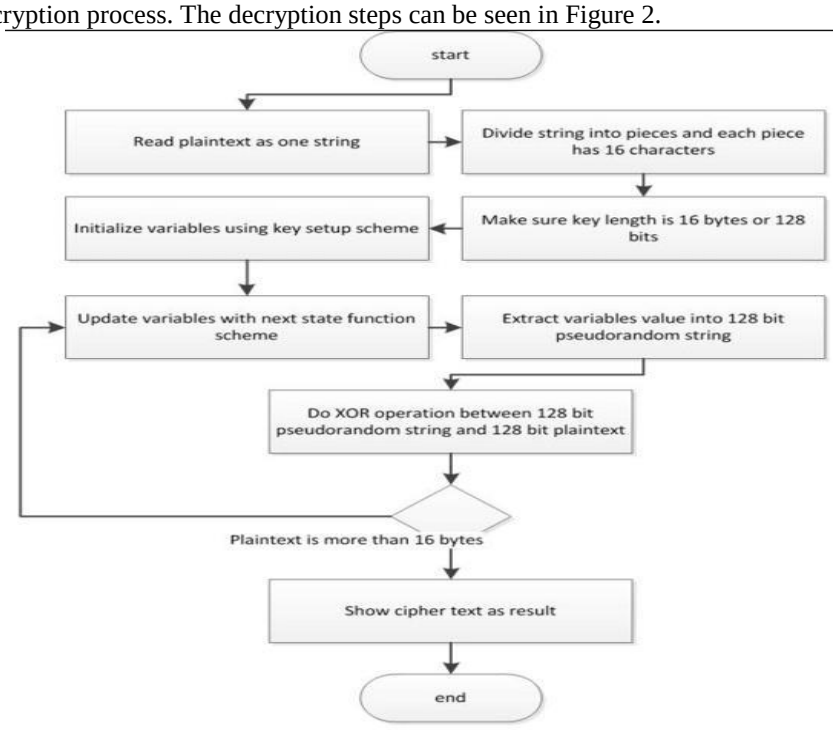


Figure 1. Encryption flowchart

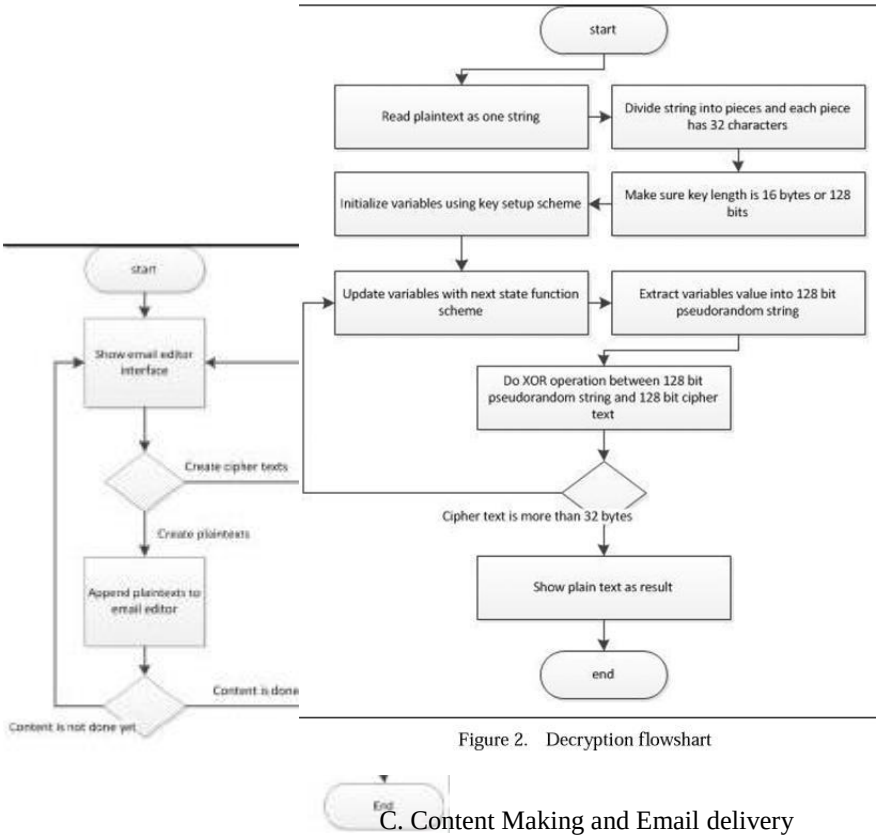


Figure 2. Decryption flowshart

C. Content Making and Email delivery

Creating content means creating ciphertext to be used as email content itself. When creating an email content session, the user You can insert plain or ciphertext or both. In order to separate encrypted text and plain text, a couple of labels are used. The The tags are "<enc>" and "</ enc>". In other words, all words these two labels are called ciphertext. Steps to Create Content and email can be seen in Figure 3.

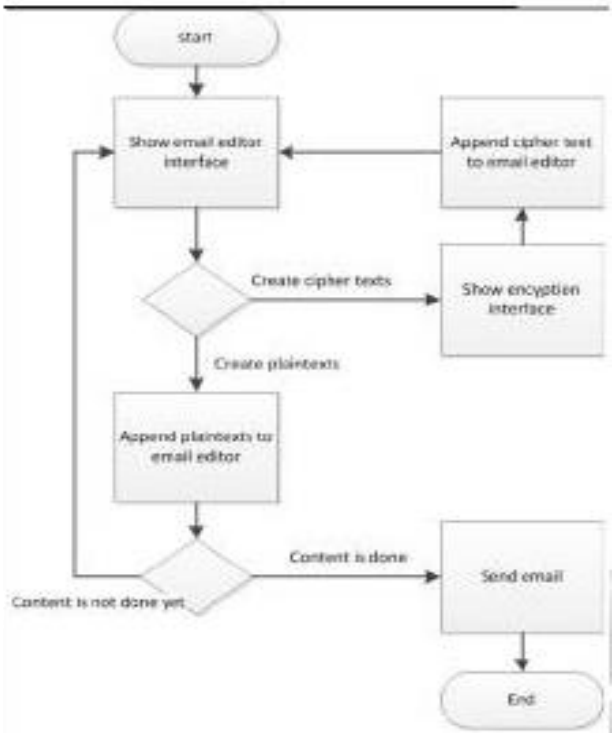


Figure 3.email delivery and content making flowchart

Any other desired content that is not encrypted Placed off the labels. Simple content – not encrypted Content - must be placed before "<enc>" or after "</ enc>. Placing plaintexts (plain content) and encrypted text can they may vary, such as ciphertext followed by plain text or vice versa. The email is sent using SMTP. JavaMail provides SMTP with TLS and SSL support. The content of the email is prepared to be sent as plain text/content. This is a simple type of email Content that contains only text.

D. Email retrival & containt reading

Recuperar el correo electrÃ³nico de internet se puede hacer con IMAP o POP3. El protocolo elegido es IMAP. IMAP es elegido Porque permite la recuperaciÃ³n de correos electrÃ³nicos desde Internet sin Suprimiendo esos email en Internet como lo hace POP3. Esta

Implica que incluso los correos electrÃ³nicos ya se leen mediante la aplicaciÃ³n Una vez, los email todavÃa pueden ser alcanzados. El contenido del correo electrÃ³nico se compone de textos cifrados y textos planos. Con el fin de leer el correo electrÃ³nico completamente, los textos cifrados y plaintexts Debe dividirse en partes donde cada parte contiene Textos cifrados o textos planos. Estas partes tambiÃ©n deben ser arregladas En la orden apropiada asÃ que cuando las piezas son ensambladas, el contenido no perderÃ¡ Su significado semÃ¡ntico. Cada parte contiene TYPE que define si el subtexto es Texto cifrado o texto sin formato y ONTENIDO que define Subtexto. Estas partes se pueden ver en la tabla a continuaciÃ³n. Si TYPEequals a cero, entonces CONTENT debe ser de texto sin formato y si TYPE Igual a uno, entonces CONTENIDO debe ser texto cifrado. DespuÃ©s Descifrado, todas las partes se unen en un solo contenido. Estos pasos Se puede ver en la Figura 4.

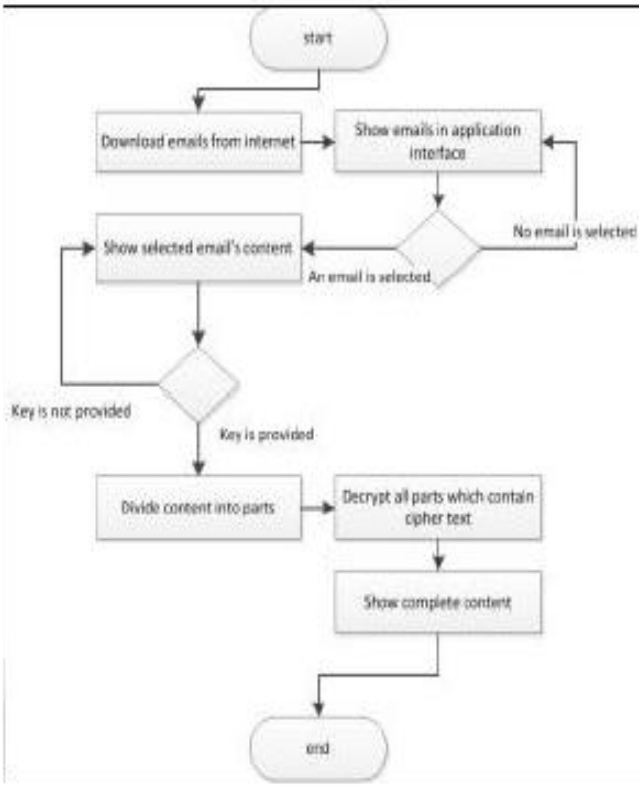


Figure 4. Email retrieval and content reading flowchart

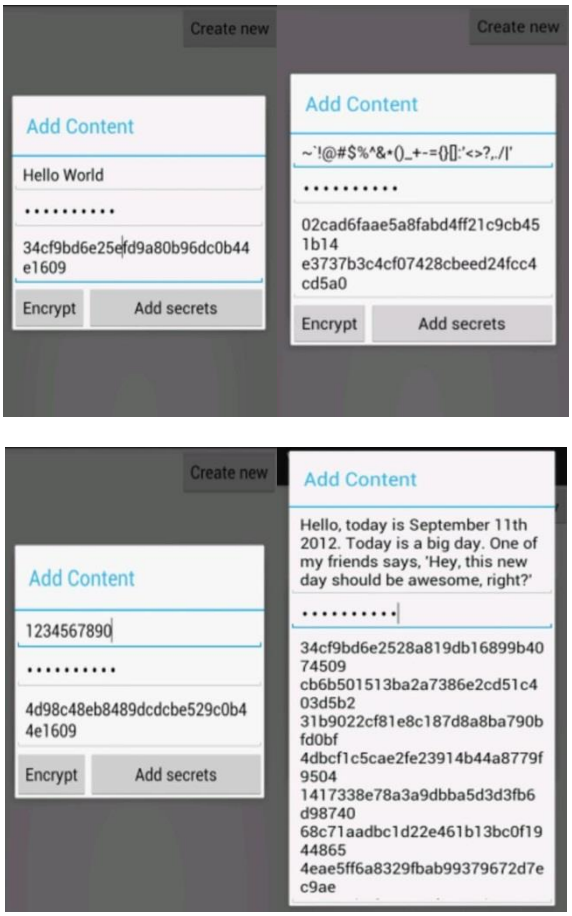
4. Implementation & experiments

The implementation is done on a personal computer. The application is enveloped with Eclipse 3.6 with Android Plugin, Android SDK, JDK 1.6 and JRE 16. The application has been tested on Sony Xperia Ray with Android 4.0.4 operating system. The experiments are performed to know if the encryption-encryption, email sharing, and content creation they work well. The encryption-decryption function is tested by using some texts to encrypt and the ciphertext is Decryption If the test data is the same as plain text from decryption the encryption-decryption function then performs well.

TABLE I. TABLE TEST DATA ENCRYPTION RESULT

No	Ciphertext
1	34cf9bd6e25efd9a80b96dc0b44e1609
2	02cad6faae5a8fabd4ff21c9cb451b14 e3737b3c4cf07428cbeed24fcc4cd5a0
3	4d98c48eb8489dcdcbe529c0b44e1609
4	34cf9bd6e2528a819db16899b4074509 cb6b501513ba2a7386e2cd51c403d5b2 31b9022cf81e8c187d8a8ba790bfd0bf 4dbcf1c5cae2fe23914b44a8779f9504 1417338e78a3a9dbba5d3d3fb6d98740 68c71aadbc1d22e461b13bc0f1944865 4eae5ff6a8329fbab99379672d7ec9ae 29a59eb4fa88159f49e59b172161da42

The content creation function is tested along with encryption proof. The test data that is used to be encrypted and inserted in Contains "Hello World", "~"! Eur-lex.europa.eu eur-lex.europa.eu September 11 of 2015. Today is a great day. One of my friends says, 'Hey, This new day must be awesome, right? '"These four test data they are defined by representing words, punctuation marks, numbers and your combination. These four test data are encrypted by using "hello world" as the key. The ciphertext of the words, Punctuation, numbers and combinations respectively can be Shown in TABLE I. Application Interfaces for All Test Data with "Hello world 'as the key are shown in Figure 5.



These four encrypted texts are then used as part of the main email content. Encrypted texts are inserted into the main content by placing between two tags, "<enc>" and "</ enc>".
Encryption is successful, encrypted texts must be decrypted. The result of decryption can be seen in TABLE II. Since decryption, the results are as readable as the original flat texts, so it's safe to assume that the encryption-decryption function works correctly.

TABLE II. TABLE TEST DATA DECRYPTION RESULT

No	Decryption Result
1	Hello World
2	~`!@#$%^&*()_+-={}[]:'<>?,./|'
3	1234567890
4	Hello, today is September 11th 2015. Today is a big day. One of my friends says, 'Hey, this new day should be awesome, right?'

Email content is created by combining clear text and encryption Texts in a writing. There are many arrangements that can they are formed by combining ciphertext with flat texts. Examples
The content can be seen in Figure 6 and Figure 7 below.

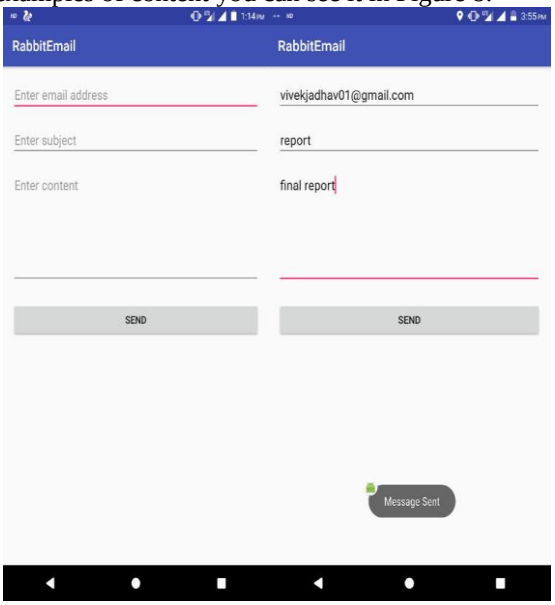

```
First two words which begin the world of
programming is "Hello World". Encrypted "Hello
World" is this.
<enc>
34cf9bd6e25efd9a80b96dc0b44e1609
</enc>
Not only words, punctuations also can be
encrypted like this one.
<enc>
02cad6faae5a8fabd4ff21c9cb451b14
</enc>
And, that is it. Farewell for now.
```

Figure 6. First example content

```
<enc>
4d98c48eb8489dcdcbe529c0b44e1609
<enc>
<enc>
cb6b501513ba2a7386e2cd51c403d5b2
31b9022cf81e8c187d8a8ba790bfd0bf
4dbcf1c5cae2fe23914b44a8779f9504
1417338e78a3a9dbba5d3d3fb6d98740
68c71aadbc1d22e461b13bc0f1944865
4eae5ff6a8329fbab99379672d7ec9ae
29a59eb4fa88159f49e59b172161da42
</enc>
First cipher text comes from numbers
encryption.
Second cipher text comes from paragraph
encryption.
```

Figure 7. Second example content

Application interfaces to create these 2examples of content you can see it in Figure 8.



There are many ways to place unencrypted text and ciphertext Two upstairs. These two examples from previous paragraphs are used for two email content. First is Rabbit email page and Emails are sent to add vivekjadhav01@gmail.com.
To check if emails are sent correctly, these two addresses are it is accessed through the use of a web browser. The result of delivery Experiment is that these two emails reach the desired addresses and the application succeeds in sending emails correctly.

5. Experiment result and Analysis:
The ciphertext is always larger than the text Character in clear text is transformed into two hexadecimal Characters in the ciphertext. The decryption results are not exactly the same as original plaintexts. The difference is at the end of the deciphered plaintexts. Decryption results may have Spaces at the end of the text. These spaces are added to make the length of plaintexts is divisible by 16. Labels are added to each ciphertext of a rabbit session. The rabbit session begins with a key configuration for one Plain text in any length. This means that if the content contains two Label pairs then the user must have used two rabbit sessions. Emails are sent using the SMTP protocol. The only email Service that is working for this application is Google Mail. Google Mail does not allow the transmission of e-mail by using unsecured protocol. That's why when using Java Mail, SMTP the TLS or SSL protocol must continue to be enabled. The IMAP protocol is used to retrieve emails. Through the use of IMAP Protocol, emails can be downloaded

and accessed over just once. The content type for emails is plain text. The reason the application is only able to send text messages. In order to read content completely, the content itself should be
Divided into parts based on whether it is plaintext or ciphertext. The breaking of content, for example, one and two, can be seen in TABLE III. AND TABLE IV respectively.

TABLE III. TABLE CONTENT PARTS FOR FIRST CONTENT

No	Type	Content
1	0	First two words which begin the world programming is "Hello World". Encrypted "Hello World" is this
2	1	34cf9bd6e25efd9a80b96dc0b44e1609
3	0	Not only words, punctuations are also can be encrypted like this one.
4	1	02cad6faae5a8fabd4ff21c9cb451b14
5	0	And, that is it. Farewell for now.

TABLE IV. TABLE CONTENT PARTS FOR SECOND CONTENT

No	Type	Content
1	1	4d98c48eb8489dcdcbe529c0b44e1609
2	1	cb6b501513ba2a7386e2cd51c403d5b2 31b9022cf81e8c187d8a8ba790bfd0bf 4dbcf1c5cae2fe23914b44a8779f9504 1417338e78a3a9dbba5d3d3fb6d98740 68c71aadbc1d22e461b13bc0f1944865 4eae5ff6a8329fbab99379672d7ec9ae 29a59eb4fa88159f49e59b172161da42
3	0	First cipher text comes from numbers encryption. Second cipher text comes from paragraph encryption.

Decryption is only done to content that has type one. After Decryption for all type one contents, all United in content. The bonding process is performed following. The order of the parts. This order must be followed for Keep the content semantic meaning.

Touching the transmission of email using WireShark is a mistake. This is due to JavaMail. JavaMail requires SMTP for With TLS. By using TLS all data frames are encrypted before exiting to the nearest network. This fact it means that the data tables can be intercepted but cannot be Read at all.

The result of using the mail debug function is a log of Application activity. This log records all messages from the network to the application and passing from the application of net. The content of the email is clearly shown on this record. Log in It shows that the encrypted texts are not changed.

CONCLUSION

We have studied to the implementation of Rabbit algorithm by using java. Labels are used to separation of plaintexts and ciphertexts. Emails are securely sent and received. Data dividing into parts which are ciphertext, and cipher text successfully decrypted.

REFERENCES

1. Muhammad Anwari Leksono, Rinaldi Munir,"Email Client Application with Rabbit Algorithm for Android Smart Phone".
2. Crispin. M, "Internet Message Access Protocol - Version 4rev1"
3. Postel. J.B, "Simple Main Transfer Protocol".
4. KetuWare, "Symmetric vs. Asymmetric Encryption".
5. Boesgaard. Martin, Pedersen. Thomas, Vesterager. Mette, "The Rabbit Stream Cipher – Design and Security Analysis".

