**MAIL-TAS**

Mohammed Taha, Mubarak Shah, Sanjay Kumar

[K191088@nu.edu.pk](mailto:K191088@nu.edu.pk) [k191074@nu.edu.pk](mailto:k191074@nu.edu.pk) [k191052@nu.edu.pk](mailto:k191052@nu.edu.pk)

Abstract

A lot of personal and private information is sent through emails from a company to another or from one colleague to another. However, there is no protection provided on the email platform which can secure email messages from malware attacks. The following report guides how SMTP and IMAP protocols to transfer emails were used along with encryption to make email messages as secure as possible. Using SMTP on the sending email and IMAP on the receiving side, this application asks the user to enter a key which appends into email message with encryption algorithm and sends it to receiver. The receiver side has a decryption algorithm in application which shows the message to receiver by eliminating the key in between the message and then decrypting it. But on the email platform, this message will be seen in encrypted form and it will be very difficult for the attacker to extract it through Wireshark or any other system.

Keywords: SMTP, Encryption, IMAP, Decryption, Mail Transfer, Advanced Encryption Standard, Receive Mail, Caesar Cipher, Secret Key

1. Introduction

SMTP (Simple Mail Transfer Protocol) is the standard protocol for sending emails across internet. IMAP (Internet Message Access Protocol) is a mail protocol used for accessing email on a remote web server from a local client. AES (Advanced Encryption Standard) is a symmetric key encryption and decryption algorithm, the most used encryption algorithm in the worldwide. SMTP and IMAP implementation with AES provides a mechanism for sending and receiving mails which are encrypted.

1. Problem Statement

Sending emails via SMTP protocol is still the most common way to communicate on the internet. However SMTP has been built without a native security layer: meaning that emails will always be exposed and quite easily hackable.

1. Approach for Development
2. Requirement

One user can send mail to other user and this mail should be sent in the encrypted form.

If any attacker has received that encrypted message then he should not be able to decrypt that.

On receiver side user can decrypt that mail.

1. Architectural View

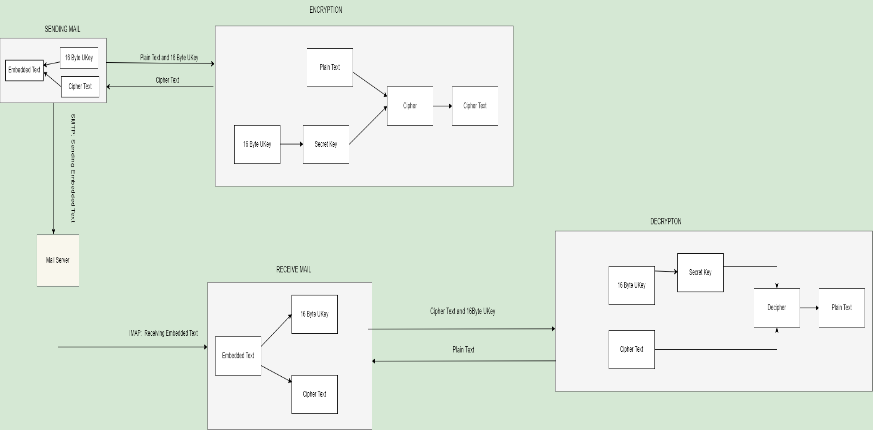


Fig1

The SENDING MAIL module request for plain text and 16 Byte UKey then this UKey is passed to the ENCRYPTION module where UKey will generate Secret Key. Plain Text and Secret Key is provided to the encrypt method/cipher where encryption is performed using AES algorithm after that Cipher Text will be generated and it is returned to the SENDING MAIL module where UKey is embedded into that Cipher Text and we call this text as Embedded Text. Embedded Text is sent to the user using SMTP protocol.

The RECEIVE MAIL module is receiving mail which contains embedded text using IMAP. UKey and Cipher Text are extracted using the RECEIVE MAIL side from that embedded text and sent to the DECRYPTION module where secret Key is generated using 16 Byte UKey and this secret Key is same as on sender side because algorithm used here is AES which is Symmetric Key Algorithm. Cipher Text and Secret Key is passed into decrypt method/decipher then it will generate plain text.

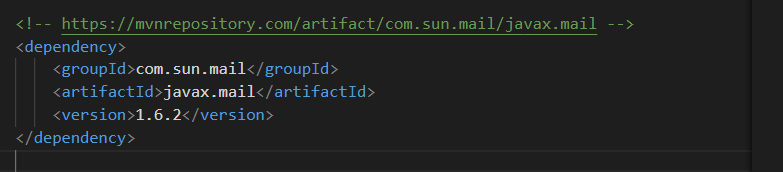
DECRYPTION module then send this Plain text to the RECEIVE MAIL module where this plain text is displayed to the receiver.

1. Development

Maven project is created for this application because it can manage a project’s build, reporting and documentation from a central piece of information.

Core Java and Swing which is a platform-independent, model view controller GUI framework for java are used for coding of application. Java Mail API is used which is platform and protocol independent framework to build mail.

Dependency added in pom.xml file for using java mail API.

Fig 2

1. Testing

UKey is critical field in our application which is basis of encryption and decryption of entire message so it is important to test this field and specially its length because UKey should be of 16 byte only. Equivalence Class partitioning method is applied here for classifying the valid and invalid inputs and there outputs.

UKey\* is user key which is entered to formulate Secret Key for encryption and decryption.

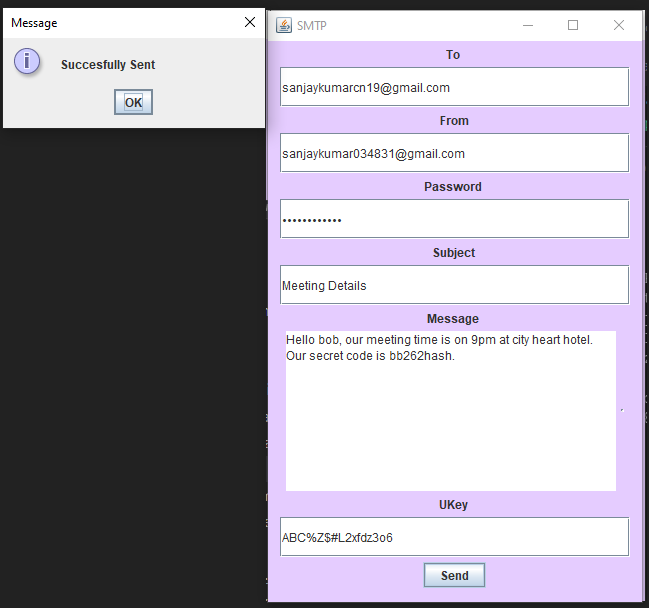


Fig 3

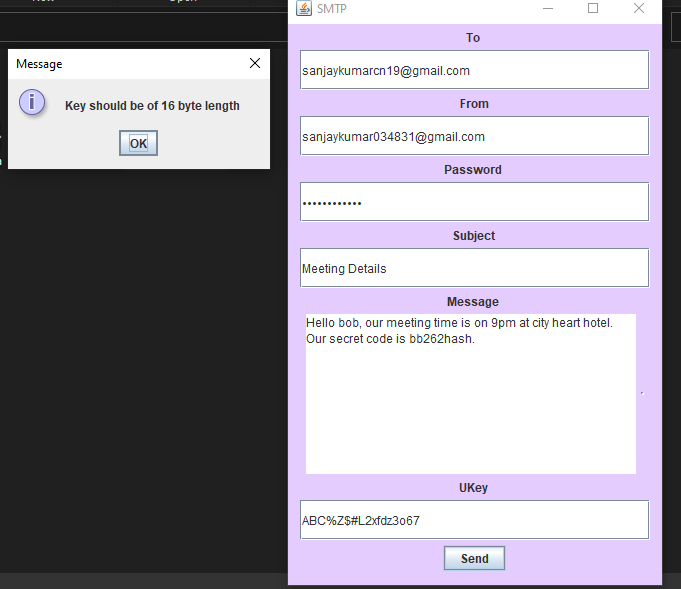


Fig 4

|  |  |  |
| --- | --- | --- |
| Equivalence Partitioning | | |
| Invalid | valid | Invalid |
| UKey. Length<16 | UKey. Length==16 | UKey. Length>16 |
| Key should be of 16 byte length | Successfully Sent | Key should be of 16 byte length |

1. Future Work

Our future work includes

* Sending and receiving mails with attachments (audio, image, video) files which will be sent with encryption.
* Refactoring and use of design patterns for scalability and reusability.
* Changing entire project into plugin play component which can easily be plugged into other applications for services of SMTP and IMAP with encryption.

1. Results

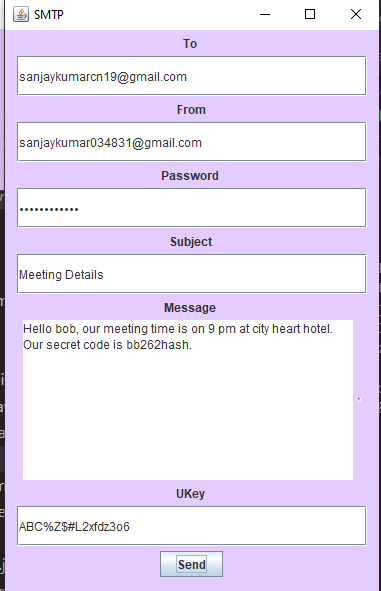


Fig 5

Mail that is sent from sender frame.

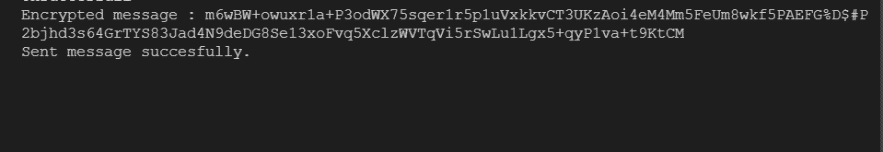


Fig 6

Above message is encrypted message that is generated from original message on the basis of UKey entered by the user.

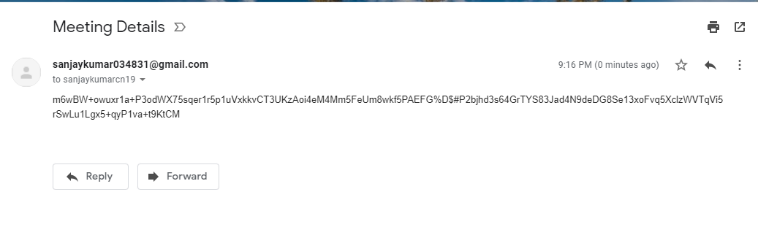


Fig 7

The mail box of users (senders, receivers) contains encrypted message which is very hard to decrypt using tools available. It is tested over websites which are used for encryption and decryption and the results are as follows:

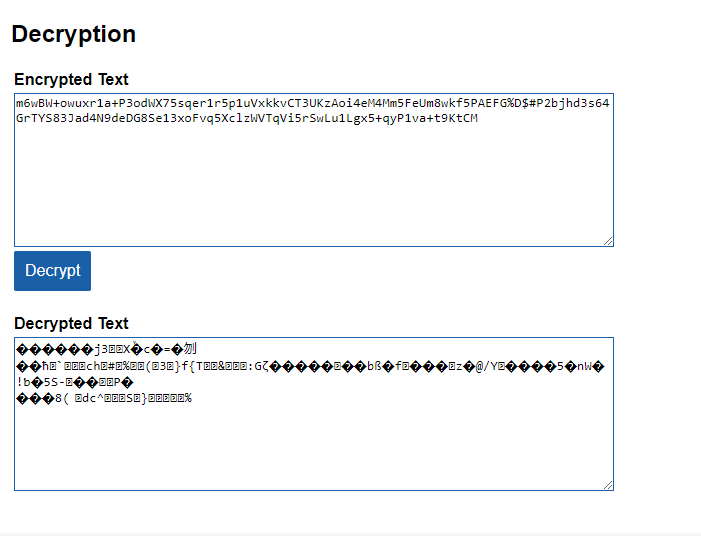


Fig 8

Results after decryption of message are still not understable to any person.

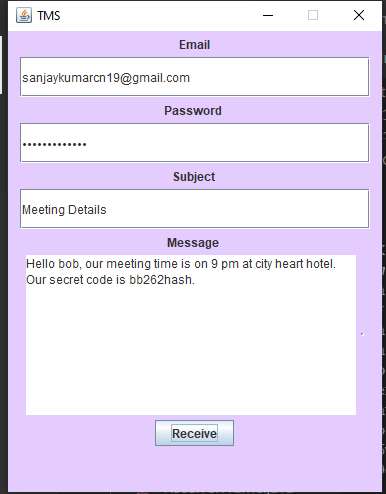


Fig 9

On receiver frame the user will enter his Email and Password and then click on receive button then decrypted message will be displayed to the user.

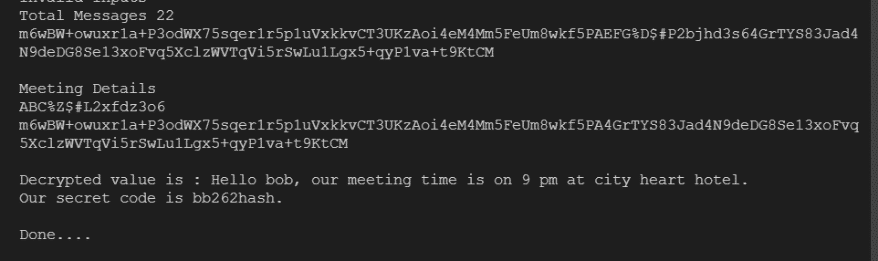


Fig 10

Here is console view which shows total messages present in receiver’s mail box and decrypted message on receiver side.

1. Conclusion

One layer of abstraction over email service provider provides a very elegant mechanism of encryption for sending and receiving emails. Encryption and decryption on input data like strings, password based data can be performed using more secure encryption AES algorithm in java.

1. References

[1] <https://mkyong.com/java/java-aes-encryption-and-decryption/>

[2]<https://en.m.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol>

[3]<https://en.m.wikipedia.org/wiki/Internet_Message_Access_Protocol>