Multi-Recipient E-mail Messages: Privacy Issues and Possible Solutions

Shafiya Afzal SHEIKH, Mohammad Tariq BANDAY
Department of Electronics and Instrumentation Technology, University of Kashmir, Srinagar, 190006, India
sheikh.shafiya@gmail.com

***Abstract*—Chain and multi-recipient e-mails pose significant security and privacy threats such as phishing and the spread of Trojan horses. They also increase the chances of receiving spam e-mails. E-mails sent to multiple recipients at a time result in unwanted exposure of e-mail address to multiple recipients. The recipients of chain e-mails may include spammers or e-mail addresses of users whose e-mail account or device may have been compromised, thereby, exposing all e-mail addresses to spammers. Forwarding or sending a multi-recipient e-mail in a chain further increases the exposure of e-mail addresses to spammers. This paper discusses chain e-mails, multi-recipient e-mails and crucial security and privacy threats they pose to legitimate e-mail user. It also discusses various possible mechanisms to mitigate these threats and investigates their effectiveness. This study proposes a novel technique to counter these security risks by enhancing the default behaviour of e-mail client, SMPT server and SMTP protocol. The proposed technique has been implemented in the Java programming language which showed promising results against unnecessary exposure of multiple e-mail addresses while sending an e-mail to multiple recipients.**

***Index Terms*—electronic mail, information security, privacy, unified messaging, unsolicited electronic mail.**

I. Introduction

Email plays an indispensable role in carrying professional conversations in both academia and corporate world besides being used for multiple other purposes. Although, individual email users, organizations and email service providers implement best security measures and policies to secure their email accounts and associated infrastructure from unauthorized access, cybercriminals breach or bypass these underlying security mechanisms and exploit it for illicit uses such as spread of viruses, SPAM [1-3], hoax [4], chain emails [5], and phishing [6] and costs business and individuals millions of dollars annually [7]. Email based phishing attacks are one of the most common modes of email scams like advance-fee frauds. The phishing attacks may even come as spear phishing which is a more advanced form of phishing and is performed by sending emails to the victims that appear to come from trusted sources and are targeted towards specific individuals or positions in organizations [8]. Several software tools have been developed that help prevent phishing attacks to some extent, such as Google Safe Browsing, McAfee SiteAdvisor, Netcraft Anti-Phishing Toolbar, Spoof Guard etc. These tools are not efficient enough and victims are still being defrauded, and sensitive data stolen. [9]. Ordinarily, the mass mailing or forwarding of e-mail messages exposes email addresses of the sender's address book to all other recipients. If any of the recipients' computer is compromised or infected by malware that tends to steal email addresses from emails on the infected machine, it can get hold of all the email addresses in email messages. This kind of multi-recipient and forwarded email messages, containing a number of email addresses, are a good source of email harvesting for spammers. A good majority of users that exchange email addresses in this manner are not well versed with computer and Internet technology.

Computer viruses and human intruders are able to collect email messages from email servers and computers of email users and can also steal email login credentials to login either manually or automatically and then access email messages and address books. After gaining access, email addresses can be harvested from these stolen email messages and address books. The harvested email addresses can afterwards be used to send SPAMs or utilized for other email-based cyberattacks such as phishing. Involuntarily giving away email addresses to potential spammers or attackers by subscribing to newsletters and filling out registration forms on malicious websites can invite SPAM and phishing attacks. This results in privacy issues to both the individuals and corporations sending emails and can land them into legal trouble by unknowingly compromising the privacy of others. According to the European Commission's General Data Protection Regulation (679/2016/EU) implemented on 25 May 2018 the personal data including email address should not be available publicly without explicit consent [10]. Therefore, it is highly desired to prevent this unnecessary broadcasting of email address books which otherwise becomes a source of email addresses for spammers and a cause of privacy breach. The efforts of email users protecting their email addresses from falling into the hands of spammers and attackers are useless in case their email addresses are given away by other users or stolen from the accounts or devices belonging to other users.

A. Problem Definition

Multi-recipients email messages can reveal more email addresses than those sent to individual email addresses, exposing a large number of email addresses from a small number of email messages to attackers. When an email is sent to multiple recipients at a time and the email server, device or email account of one of the recipients is compromised, it reveals the email addresses of all the recipients of the message as well as the email address of the sender. Chain emails escalate the problem and the probability of the messages falling into the hands of spammers increases exponentially with every forwarded

Digital Object Identifier 10.4316/AECE.2021.04013

message [11]. In a chain email, the number of email addresses that can get accumulated in the message at every step can be simply calculated with N = (S*R) + 1 where R is the average number of recipients the email gets forwarded to at every step of the chain and S is the step number at which the calculation is done. The farther in the chain 'N' is calculated, the higher is the number of email addresses that get broadcasted and higher the number of recipient's 'R' in every step of the chain, higher is the number of email addresses exposed.

When sending an email message to multiple recipients, the default behavior of email clients as well as email protocol is to add the recipients in the TO header field. Once the email is sent for delivery, all the email addresses are included in every outgoing copy of the message. Hence, multi-recipient emails, chain emails, forwarded social email messages expose the email addresses of all the recipients to attackers.

A basic experiment was performed to demonstrate the efficiency with which email addresses can be extracted from email messages. The experiment also shows how an exceedingly large number of email addresses can be obtained from a small number of chain emails, multi-recipient, and forwarded email messages. The following algorithm has been used to extract the email addresses from the header and body text of email messages stored in a directory:

```
Begin
Set regex = "^[A-Z0-9._%+-]+@[A-Z0-9.-]+\\.[A-Z]{2,6}$";
ForEach EmailFile in Directory
  Set mailtxt = ReadFile(EmailFile);
  Set emailIds = Filter(regex, mailtxt);
End ForEach;
SaveToDB(emailIds)
End
```

The algorithm begins with defining a regular expression pattern that matches email addresses in a long text string. Then it starts to read email files stored in a directory using a loop until all the files are read. Once a file is read and the email text is extracted, the regular expression is applied on the text and all the email addresses are extracted from the text. The extracted email addresses are added to an array "emailIds". The process repeats for every email file in the directory. The algorithm not only extracts the email addresses from the envelope and header of the email messages but also from its body. The algorithm calls three functions namely ReadFile() - used to read the contents of an email file and returns text obtained from the file, Filter() – that applies a regular expression pattern to a given text string and returns an array of matches and SaveToDB() – used to save the email addresses into the database. These three functions can be implemented in any popular programming language and a database. In this work, the algorithm has been implemented and tested in Java language using a MySQL database. The implementation of the algorithm was tested on a directory containing 50 multi-recipient and forwarded email messages downloaded from a couple of personal Gmail accounts. From just 50 email messages, the algorithm was able to extract about 2000 unique email addresses.

B. Contribution

This work identifies a security and privacy issue in the default behavior of SMTP protocol and email servers while handling multi-recipient and forwarded email messages. It discusses the possible, already existing solutions to the identified issue and their limitations. This work addresses the problem via two methods. In the first method, the SMTP protocol is enhanced to change the default behavior of the email server with regards to how it handles multi-recipient and forwarded emails. This method requires enhancements to email clients as well in accordance with the enhancements made to SMTP protocol. In the second method, the issue is addressed by enhancing the User Interface and default behavior of email clients only in handling such emails to provide more security and privacy for multi-recipient and forwarded emails. The comparison between the two methods is also discussed, highlighting the advantages and disadvantages of each. Both of the proposed methods have been implemented and tested in this work.

II. Background Word

A possible workaround to avoid the above-mentioned issue is the use of the BCC [12] field while sending multi-recipient email messages. BCC field can, to some extent, help control exposing the email address of all the recipients to one another. The email addresses in the BCC field are not displayed to all recipients of the email message. However, there is a potential security issue even in the BCC field. As per the implementation of the SMTP protocol, all email addresses, including BCC addresses, are included in the RCPT TO field in every email as they are sent across the network but the BCC addresses are not added to the DATA section of the email, thus, invisible to the recipient. Hence, if the recipients or intruders can access the email server, they can easily view the BCC addresses in all the incoming email messages. SMTP is designed that way in order to save the processing time taken in creating a unique addressee list for each email for each BCC recipient. A single copy of the email can be sent to each domain, even if more than one BCC recipient belongs there. The recipient server of the domain then stores a copy of the same email for all the RCPT TO recipients at its domain [13]. Seldomly, an email server may not remove the BCC list from DATA of copies of emails it sends or stores, as a result of the incorrect configuration, thereby, revealing the complete BCC address list to the recipients of the email. Hence, BCC is a possible option but not much efficient and reliable in keeping addresses confidential in critical and sensitive email communications.

Also, all email addresses in TO and CC fields are visible on every copy of the email addressed to BCC addressees. The BCC addressees can easily notice that they have received the email via BCC field and can get suspicious [14], wondering who else got a copy of the email message and why they are in BCC. Some email clients don't even show the BCC recipient's email address in the email header and instead show the sender's email address in both FROM and TO fields. Since blind copying is usually considered a form of deception, some users consider it unethical to use BCC field.

The undercover use of Bcc header may unintentionally be revealed to the other 'TO' and 'CC' recipients, either by the Bcc recipient 'Replying-All' in an email client or by simply talking about the email in person with other recipients. Some scholars believe that the secretive nature of Bcc'ing will lead to negative outcomes when its use is revealed to the recipients in the 'TO' and 'CC' fields or even to the Bcc field recipients [14].

Using BCC field hinders in proper use of encryption of emails and use of email standards like OpenPGP and S/MIME because encryption will leak the identities of the BCC recipients. The presence of a BCC recipient will prevent decryption of the encrypted email by all other recipients, whether they are in 'TO' or 'CC' fields [15].

Therefore, by making relevant changes in the SMTP Protocol and/or the email clients, the involuntary exposure of email addresses to spammers and privacy issues can be avoided.

SMTP is an Internet standard communications protocol initially defined in RFC 821 [16] and is widely used across the world for email communication. Email clients, servers and relays use this protocol to communicate with each other making up the email infrastructure. SMTP lacks some basic features and every now and then, efforts have been made to improve or extend the scope and features of the protocol, making it more practical, reliable and adaptable to the changing nature of electronic communication.

SMTP, by default, has no mechanism for authentication and does not provide any security or privacy to email communication. The security and privacy features of email are provided by SMTP extensions and was updated by RFC 5321 [17] to allow various extensions. For example, the SMTP AUTH extension provides SMTP authentication feature to SMTP protocol. SMTP authentication is defined in RFC 4954 [18] that provides an extension for simple authentication and Security Layer. By default, all email servers used to be open relays before the introduction and implementation of SMTP AUTH extension. It emerged as a serious issue because of the spread of email SPAM and worms. Bulk mailers use several different techniques to send their spam including the misuse of SMTP protocol in badly configured MTAs, aka open-relays, to hide their tracks. SMTP AUTH makes it compulsory for senders to be authenticated using a password in order to send emails.

As stated above, SMTP provides no security or privacy by default, therefore, the communication between SMTP servers and clients occurs using plain text over unencrypted channels. Over an untrusted network, the email communication can be subjected to eves dropping attack without much effort. Another extension to SMTP service allows secure SMTP over Transport Layer Security which is defined in RFC 3207 [19]. This allows for the secure and encrypted transmission of emails over non-secure networks like the Internet.

Email SPAM or unwanted email is one of the major problems in the email communication system in contemporary times and does not have any fully functional permanent solutions. Three new types of policies have been proposed [20] that give the SMTP recipients possibilities to control the recipient of email messages, and the connection with the sender. The first proposed policy is the Service Level Agreement Policy (SLAP) which addresses how the recipient ESP interacts with a given sending ESP. The second proposed policy is the Message Scheduling Policy (MSP), which is the output from SLAP evaluation and specifies the methodology of dealing with email messages in the sending ESP. The third proposed policy is the Mailbox Resource Allocation Policy (MRAP) which regulates the presentation of email messages to the recipient. This work has also proposed an extension to the SMTP protocol architecture by extending the handshake procedure and adding SHLO command and additional headers for the policy support which are X-Attributes, X-HdrHash, and X-Hash. The proposed extension of the SMTP protocol leads to additional processes and expands the commands dialogue before the email message transfer to the addressee.

The SMTP protocol does not specify any rules to account for the delivery time and performance of email communication. The delivery times and performance of the protocol may vary under different circumstances. The proposed smart SMTP protocol enhances the email delivery process which adds reliability and high performance and tries to decrease the delivery time by decreasing the time taken in the scheduling phase achieved by processing all messages addressed to the same email server as a bulk. The proposed SMTP protocol contains two smart agents which are Mail Transfer Agent (MTA) and Delayed Delivery Agent (DDA), these agents use a few additional components to improve and enhance the delivery of email message [21].

There are a few points in SMTP protocol that introduce the unwanted latency occurring during the SMTP client/server communication procedure. A new architecture for the client/server interaction has been proposed to improve the current SMTP standards enhancing the email communication performance and decrease the latency in delivery time. To improve SMTP standards and delivery time an enhancement in the SMTP client/server procedure by removing a couple of greeting message commands dialogue using the T/TCP [22] protocol, and use a modified pipelining approach in the SMTP procedure as a means to decrease the time taken to transfer the email message from the client to email server have been proposed [23].

SMTP protocol allows sending email without any data compression by default and email messages can take a long time for transmission over a network. A compression extension has been proposed and introduced into the email system to enable embedded data compression mechanisms [24]. Using this technique, the efficiency of SMTP and POP3 [25] protocols can be enhanced. By utilizing the embedded data compression technique, the average time required to send an E-mail message is reduced, but the number of processes increase before starting the transaction. This happens because a compression algorithm is applied before submitting the email message for delivery.

Further, an active E-mail system SMTP protocol monitoring algorithm has been proposed which monitors the SMTP protocol in real-time in a real working environment and detects any problems or failures resulting because of the SMTP protocol [26].

The above-mentioned works have been undertaken in order to overcome some of the many shortcomings of the SMTP protocol. The improvements are accomplished by

extending the protocol itself or changing its implementations and algorithms of the email servers that use this protocol.

III. Proposed Solutions

This work proposes two solutions to integrate the privacy concerns of e-mail users in multi-recipient email messages. These are:

- Solution 1: Modification in E-mail Client; and
- Solution 2: Enhancement in SMTP Protocol.

A. Modification in E-mail Client

The above discussed problem can be addressed by making changes to the way email clients work and how they are designed. The email clients use the TO, CC, BCC, and the Message text input fields to generate an email message, its header and its envelope. By default, the email clients add all the email addresses, provided in all the three address fields, to the email envelope and add all the TO addressees and CC addressees to the email header. The BCC addressees are not added to the header field. The email is then sent to the sending email server for delivery across the network.

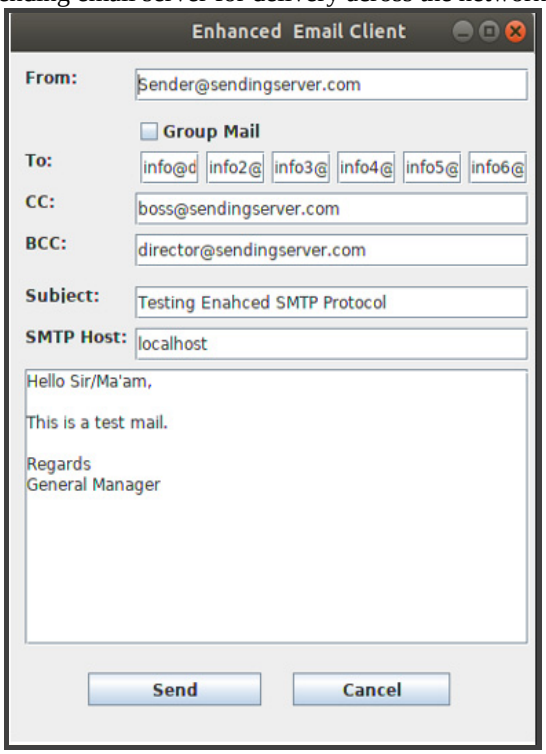

Figure 1. Screenshot of the email client with the "Group Mail" checkbox unchecked and multiple TO fields for multiple recipients

This work proposes to provide an option to the UI of the email client to allow users explicitly declare whether or not the email messages should be sent privately to all recipients. If the user opts to send the messages separately, it creates separate copies of the email for each addressee, adding only the addressee in the email envelope and email header before delivering the copies to the sending server for delivery.

B. Implementation of Solution 1

Both proposed solutions have been implemented and tested in this study. The implementations have been using Java programming language. For both the methods, a simple GUI email client was developed in JAVA Swing with basic email composition features. Other than the regular TO, CC and BCC fields, a checkbox with label "Group mail" was added right before the TO field. The TO field was enhanced to make it react to the changing state of the "Group Mail" checkbox. If the checkbox is not checked, the TO field breaks down into several smaller text fields. This makes the user know that the multiple recipients must be added in separate text boxes because it is not a group email and a separate copy of the email will be sent to every recipient.

If the "Group Mail" checkbox is checked, all the small textboxes are replaced with a single large textbox where the user can type in multiple commas separated email addresses. The user also gets to know that it is a group mail and all the recipients of the email will get to know about the rest of the addressees.

In this implementation, no changes were made to the email server or the SMTP protocol. The send button triggers a function that sends email messages based on the state of the "Group Mail" checkbox. the working of the Send function of the email client is depicted through a flowchart as shown in Figure 3.

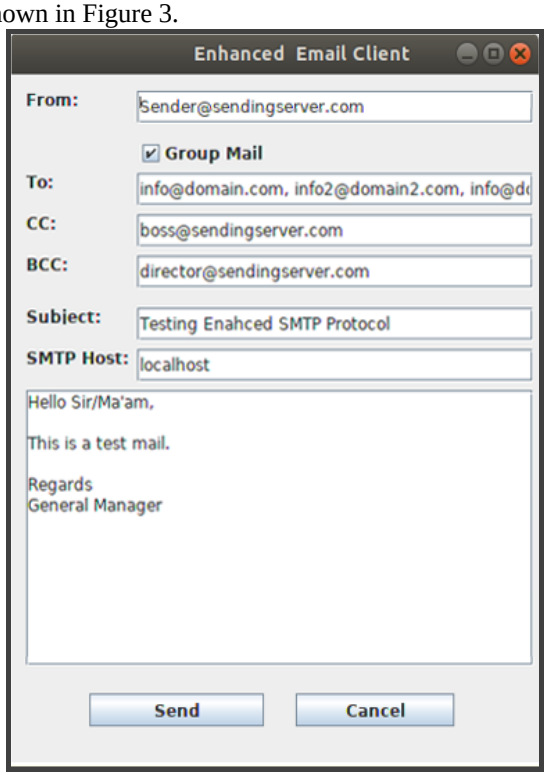

Figure 2. Screenshot of the email client with the "Group Mail" checkbox checked and a single TO fields for multiple recipients

The algorithm checks the value of "Group Mail" checkbox. If the checkbox is checked, the comma separated text from the TO, CC, and BCC fields are read and converted to arrays of addresses. After validation of the format of email addresses, email client sends the email messages just like a regular email client does. The email addresses obtained from the TO, CC and BCC fields are added to the email envelope and the email addresses from the TO and CC fields are added to the email header. The client then connects to a locally installed email server and submits the message for transmission. If the "Group Mail" checkbox is not checked, separate email addresses from multiple TO fields are converted in an array of addresses. The comma separated email addresses from the CC and BCC fields are obtained in separate arrays. Looping through all the email addresses obtained from the multiple TO fields,

the addresses are validated and a copy of the email is created. In the envelope of every copy of the email, only one of the TO recipient addresses is added. The email addresses from the CC and BCC fields are also added to the envelopes of all the copies of the email message. The email address from the TO field and those from the CC field are added to the header of the email message and the email messages are submitted to a locally installed email server for delivery.

C. Testing and Results of Solution 1

Testing of this method is carried out by sending a maximum of eighty (80) email messages using the enhanced email client. The messages are sent to email addresses created on a local email server. For this test, Mercury/32 Mail server was used.

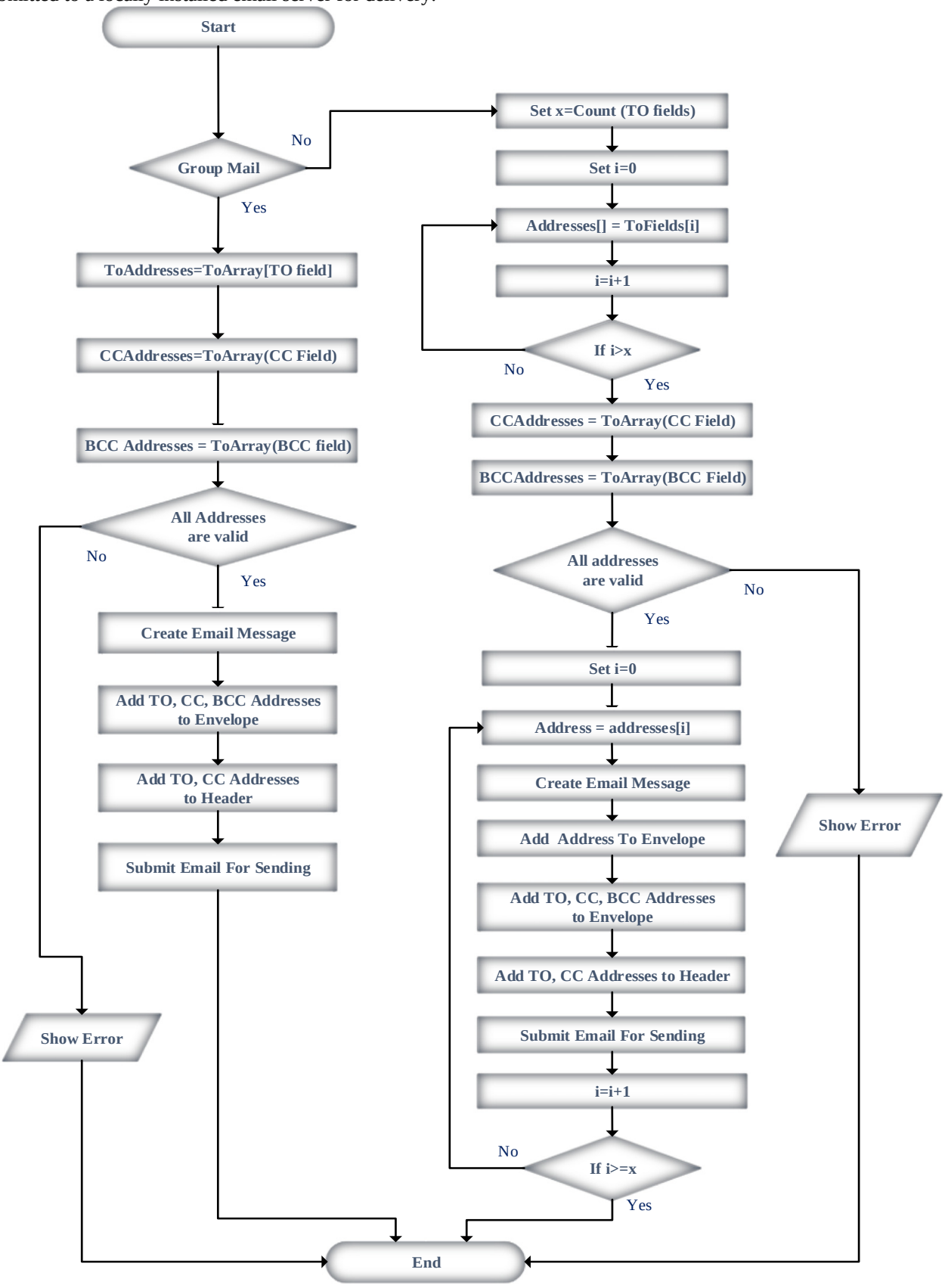

Figure 3. Process Logic of Modified Email Client

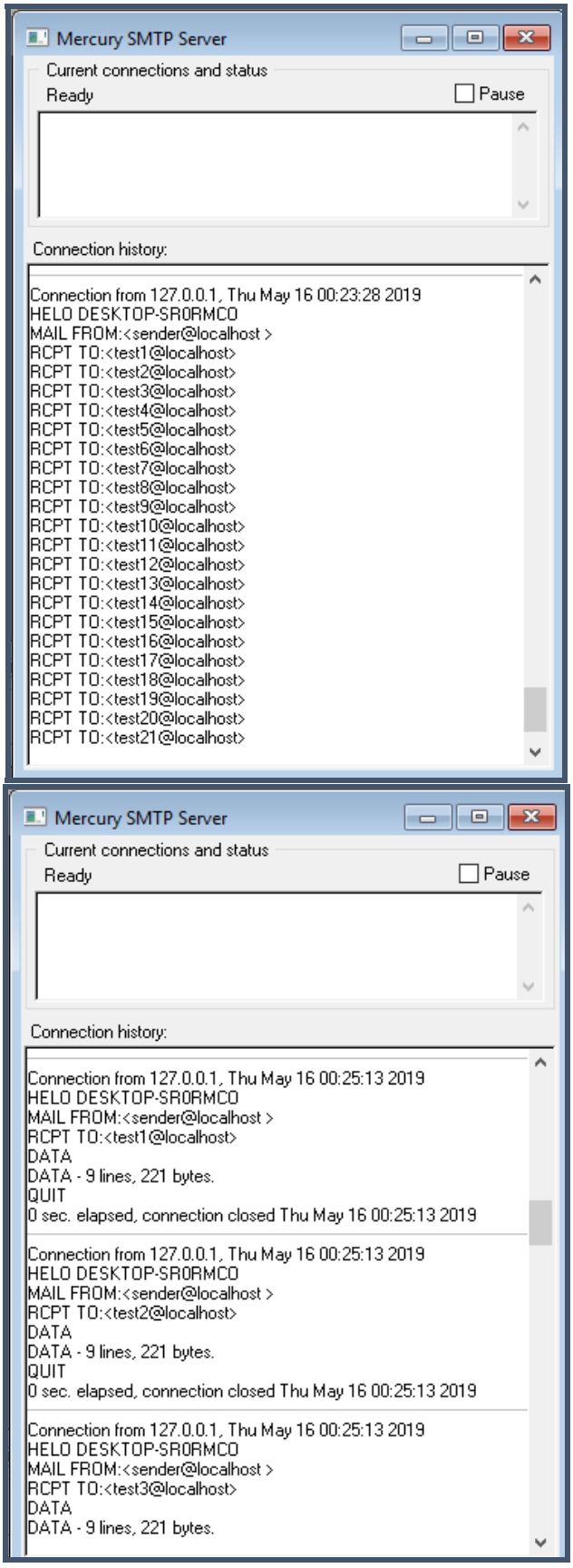

Figure 4. SMTP Server Log with and Without Group Mail

Eighty (80) email messages addressed to local email addresses were sent with and without the group mail option of the email client checked. The Email messages are received by the mail server. Outlook 2016 was used to fetch the email messages using Internet Message Access Protocol (IMAP) [27], [28] from the local Mercury/32 email server. The emails sent with Group Mail option checked, displayed all the recipient addresses as well as those of the CC addresses correctly. With Group Mail option un-checked, the email messages received in the various inboxes in Outlook 2016 did not reveal any of the other recipients. Figure 6 shows screenshot of SMTP server log with and without Group mail option checked in the email client. Since this method does not require any modification in the SMTP protocol; therefore, Mercury/32 mail server was used.

TABLE I. RESULTS OF EMAIL MESSAGES SENT USING THE EMAIL CLIENT ENHANCED IN THE FIRST METHOD

Test #	Group Mail	Number of addresses in			Number of addresses revealed per email	
		TO	CC	BCC	Server	Client
1	Yes	10	0	30	40	10
2		20	0	30	50	20
3		30	5	30	65	35
4		40	10	30	80	50
5	No	10	0	30	0	0
6		20	0	30	0	0
7		30	5	30	5	5
8		40	10	30	10	10

Table 1 shows the result of the tests beds. Tests 1 through 4 show that the number of email addresses revealed to the server is equal to the total number of email addresses in all the three address fields. The number of email addresses revealed to the client are equal to those revealed to the server excluding the number of addresses in the BCC field. The addresses in the BCC field are not revealed to the clients. For the emails sent with the Group Mail option un-checked, the number of email addresses revealed to the recipient server as well as the recipient client are the same and equal to the number of email addresses in CC. The email messages only reveal the CC addresses and one of the TO or BCC addresses to the recipient server and the recipient client. The only additional email addresses revealed to any recipient are those of the addressees in CC field. The email address of the recipient has not been counted in the list of addresses revealed. The results clearly show the method is highly effective sending email messages anonymously without revealing email addresses unnecessarily.

D. Enhancement in SMTP Protocol

This solution implements enhancements to the SMTP protocol and the process of email transfer between various nodes. Both email Server and email Clients require changes in accordance with the enhancements to the SMTP protocol. The User Interface of the email clients' needs improvement to allow the user to explicitly indicate whether or not the email recipients should see the list of all other addressees. The SMTP protocol is enhanced by adding an option to communicate the user's choice to the sending server which handles rest of the process. The changes to the SMTP protocol include adding a couple of additional commands to the protocol. This work suggests adding ANONYMOUS, RCPT CC and RCPT BCC to the protocol alongside the existing RCPT TO command. Through this, the email client can communicate with the server regardless of whether or

not email recipients can see the full list of addressees and it allows the email server generate the email header.

Figure 5 shows a sample communication between an email client and an email server using the enhanced SMTP Protocol. In Step 6 of the communication, a new command "ANONYMOUS" has been added which helps the client instruct the server that the email should be sent to the recipients anonymously without the recipients knowing about rest of the addressees. Also, in step 12 and 14, the commands "RCPT CC" and "RCPT BCC" have been added. These commands instruct the server to send a copy of the email to the "RCPT CC" addressee and a blind carbon copy of all messages to "RCPT BCC" addressee.

E. Implementation of Solution 2

The implementation of this method is done by modifying the email client UI by adding the extra "Group Mail" checkbox before the TO input field. The seemingly regular TO field was modified so that it converts into multiple TO fields on checking the "Group Mail" option. On unchecking the "Group Mail" option, the multiple TO fields join to form a single TO input field. The send button in this method triggers a function, in which the client, after connecting to the server, sends the ANONYMOUS command and after receiving an acknowledgment from the server, sends the list of recipients and DATA without the TO and CC header fields. The header information is sent to the server using the newly introduced commands RCPT CC and RCPT BCC. The working of the email client's send function with enhancements is shown in through a flowchart depicted in Figure 5.

The client checks whether or not the "Group Mail" checkbox is marked, if marked, indicating that the client wants to explicitly send a group mail, it submits the email using the regular way without using any newly added commands. The data sent to the server also includes the TO and CC header fields as usual. The server handles the email in the regular way as well. In case the user has not marked the "Group Mail" checkbox indicating that he doesn't intend to send a group email, the client, after connecting to the server, sends ANONYMOUS command to the server and then sends the email addresses from TO field, CC field and BCC field via the RCPT TO, RCPT CC and RCPT BCC commands respectively. Finally, the DATA is sent without any addresses in the header field. The server in this way gets three sets of email recipients and can add the header fields to the email DATA.

The algorithm for Enhanced Email Client is listed below:

```
Begin
If isGroupmail.Value==True then
   Set ToAddress= []
   ForEach ToField in ToAddressFields
       Set ToAddresses.Push(ToField.Text)
   End ForEach
Else
   Set
   ToAddresses=ToArray(ToAddressField.Text)
EndIf
Set CCAddresses=ToArray(CCAddressField.Text)
Set
BCCAddresses=ToArray(CCAddressesField.Text)
If Valid(ToAddresses) AND Valid (CCAddresses)
AND Valid(BCCAddresses) Then
   If isGroupMail.Value==True Then
       Set email=new Email()
       email.AddRecepients(ToAddresses)
       email.AddReceipents(CCAddresses)
       email.AddReceipents(BCCAddresses)
       email.AddHeaders(ToAddresses)
       email.AddHeaders(CCAddresses)
       email.send()
  Else // Not a Group Mail
       Set email=new Email()
       email.AddToRecepient(ToAddresses)
       email.AddCCRecepient(CCAddresses)
       email.AddBCCReceipent(BCCAddresse)
       email.SetAnonymous(True)
       email.Set Body (BodyField.Text)
       email.Send()
  EndIf
Else
  ShowMessage("Email Address Not Valid")
EndIf
End
```

The algorithm for modified SMTP server elaborating the handling of the submitted messages is listed below:

```
Begin
Open Connection
Set Command = ReadClientCommand()
While Command! = ENDSEQUENCE
Switch Command
       Case "RCPT TO"
              ToAddresses.Push(Command Param)
       Case "RCPT CC"
              CCAddresses.Push(Command Param)
       Case "RCTP BCC"
              BCCAddresses.Push(Command
       Param)
       Case "Anonymous"
              Set anonymous=True
       Case "DATA"
              Read DATA
              Set Data=DATA
       //Handle Other Commands
       End While
       Set    Addresses=Merge    (ToAddresses,
       CCAdresses, BCCAddresses)
If anonymous == True
For Each address in Addresses
              email=new Email()
              email.AddReceipent(address)
              email.Body(DATA)
              email.AddHeaders(receipent)
              email.AddHeaders(CCAddresses)
              email.AddToQueue()
End ForEach
Else
       Send Regular Way
End If
  End
```

In this method, the server SubEtha(https://github.com/voodoodyne/subethasmtp) SMTP is created to accommodate the new commands of the proposed enhanced SMTP protocol to receive and relay email messages. The commands ANONYMOUS, RCPT CC and RCPT BCC have been added to the list of accepted command in the SMTP server. On receiving the ANONYMOUS command, the server responds with a "250 OK" message and stores the incoming email addresses arriving via the RCPT CC and RCPT BCC commands in two additional sets of recipient email addresses. The SMTP server is also modified such

that, before storing or relaying the messages to the next hop, it checks the ANONYMOUS field and if set to true, it creates a separate copy of the email for every recipient from the three sets of recipient email addresses and creates envelopes and headers for every copy of the message.

The flowchart shown in figure 6, depicts how email messages are handled by the enhanced SMTP Server. The process logic is shown in Figure 7.

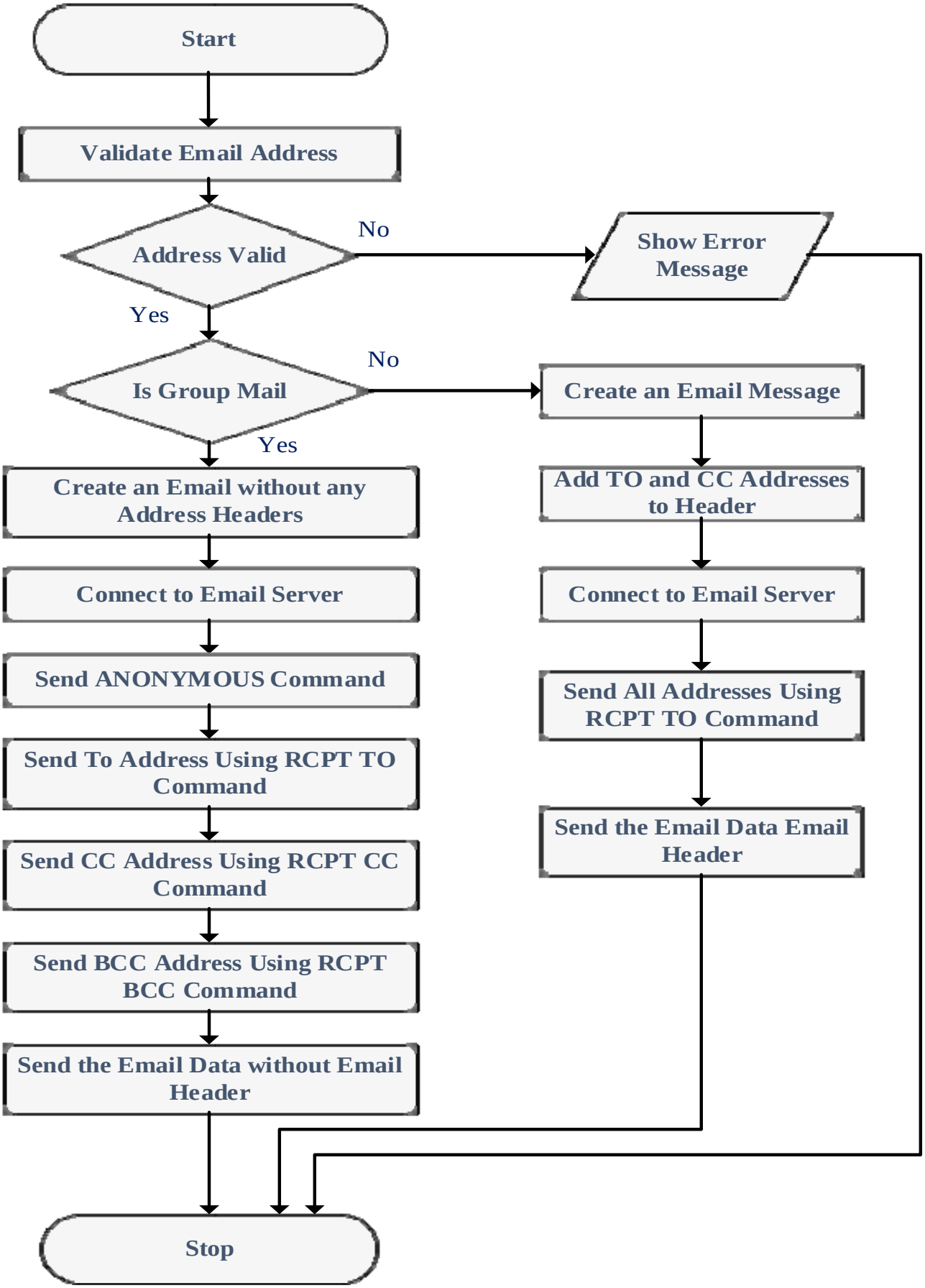

Figure 5. Process Logic of Modified Email Client

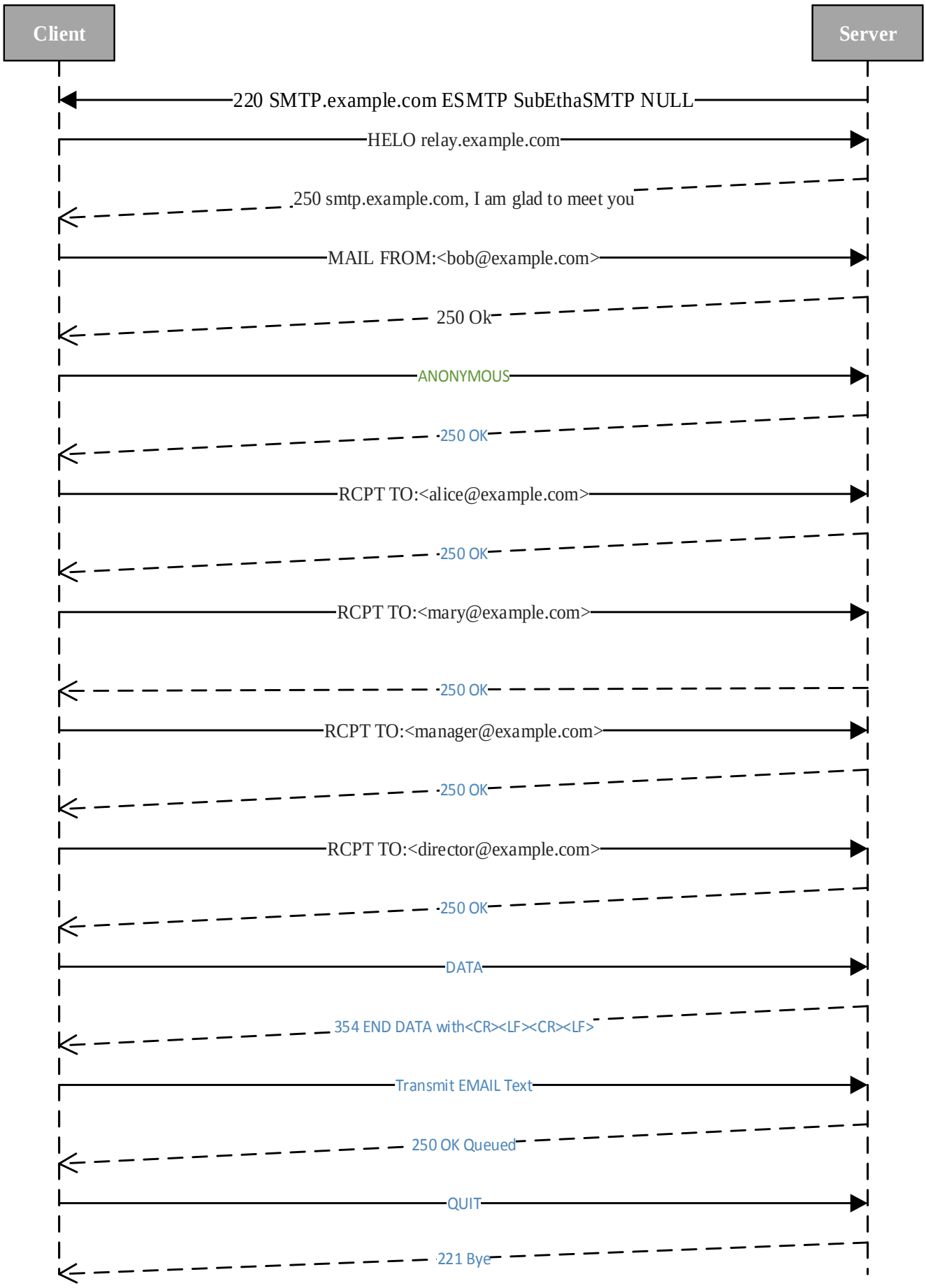

Figure 6. Communication between client and server using modified SMTP Protocol

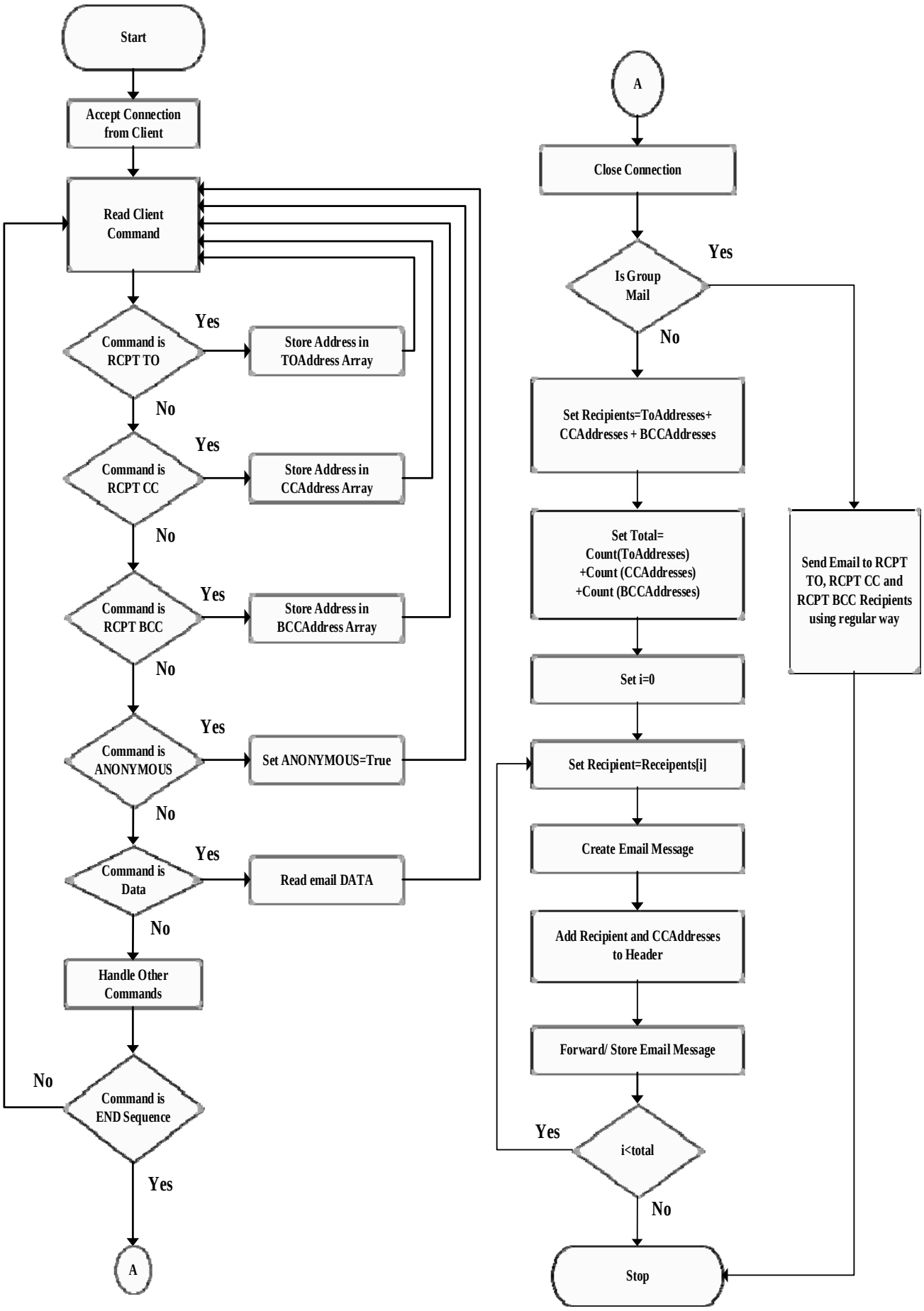

Figure 7. Process Logic of Modified SMTP Server

F. Testing and Results of Solution 2

The testing of the enhanced SMTP server with modified SMTP protocol and the email client has been carried out on two VPS servers [29]. The enhanced SMTP server was

installed on the first VPS server (the sending server) and a regular email server was installed on the second VPS server (the recipient server). Two domain names were configured on the two VPS servers as well. The enhanced email client was installed on a local computer and configured to use the sending email server. Using the email client, email messages are drafted and submitted to the first VPS server addressed to email addresses created on recipient server and domain name. The email messages were sent with the Anonymous checkbox set to "checked" as well as "unchecked" state to check its effects on the email messages. The tests resulted in email messages being sent in a way that different recipients got a copy of email message without any sign of other email recipients' email addresses. The results obtained through this method are exactly the same as those obtained through first method which are shown in Table 1.

IV. Comparison Between Proposed Solutions

In the first method i.e., the enhancement in email client, only the email client needs enhancements and no modifications are required on the email server or the communication protocols. This method is independent of Email Server and works with all types of Email Server without any compatibility issue. Therefore, it is very easy to implement and doesn't need to be implemented on the entire email infrastructure at once to start functioning. The solution can be implemented by incorporating the proposed UI changes into stand-alone email clients as well as web-based email clients. In web-based email clients, the changes can be implemented by adding the suggested UI components using HTML and JavaScript and handling the submission algorithm of Fig. 3 using the back-end code. The disadvantage of this method is that additional network resources will be required for submitting multiple copies of the email to the sending email server instead of a single copy.

In the second method, enhancements are required on both sides, i.e., SMTP Server as well as the email clients. The changes in the email server help email client not require additional network resource because only one copy of the email will be submitted to the sending email server. The enhanced email sever incorporates part of the solution and hence knows how to handle the email message. The disadvantage of this method, however, is that implementing changes in a communication protocol or email server which is used as widely as SMTP protocol is not very easy. One of the reasons for SMTP protocol not having been modified for a long time is its vast implementation and difficulty of propagating the enhancements over the entire email infrastructure. Modifying only the email clients makes no difference in this method because the sever will still handle the submitted messages in the regular manner. Another advantage of this method is that the proposed solution can be implemented as a default behavior in the email server, forcing the email clients to upgrade and comply with the changes.

V. Conclusion

Chain and multi-recipient emails can reveal a huge number of email addresses to spammers. When forwarding an email message, the list of previous senders and recipients is preserved in the forwarded email. If the message is forwarded to more than one email address at a single time, all the recipients by default receive the list of all other recipients. A large number of email users may never want to share their email addresses publicly and may never forward chain emails, but if their email address is added to a list of recipients by any other user, the email address is shared with all the other future recipients of the chain email. If an intruder is able to gain access to the email account of any of the future recipients of the chain, all the email addresses of previous recipients can be extracted and used for sending spam email and other email-borne attacks. Use of BCC field for adding recipients can in help avoiding this problem, in which the list of BCC recipients is not shown to all other recipients of the email. But the BCC field has its own limitations. The use of BCC field in email messages undermines the trust between the communicating parties as it is clearly displayed to the recipients that their address was added in a BCC field, thus raising suspicions and mistrust.

This work highlights security threats, privacy and psychological implications of the unintended multi-recipient email message and chain emails. In order to minimize the threats, this work has proposed, implemented and tested two techniques which include the changes to both the email clients and email server. In addition, the SMTP protocol has been enhanced to support the changes. The first technique proposes to modify the User interface and functionality of email clients only. The proposed method effectively solves these issues by providing users additional UI options, which help them to explicitly decide whether or not the recipients of email messages are supposed to be able to see the other recipients of the same message and by default letting them send emails individually and avoid unnecessary broadcasting of the email addresses of all recipients. The second method proposed in this research enhances the SMTP protocol and involves changes in email clients and email servers. This method solves the discussed problems effectively without using additional network resources on part of the end user. The proposed method solves the problems effectively by adding an additional command in the SMTP protocol. The command, namely "ANONYMOUS" instructs the email server to send the email messages to the recipients without including the other recipients in the message. The ANONYMOUS command is issued by the Email Client to the Email Server, using the enhanced Email Client. The enhancement in the UI of the Email clients let's user choose between displaying or hiding the recipient email addresses from the recipients of the email message.

VI. Limitations and Future Scope

The first proposed solution, which includes changes only to the UI of the email clients can effectively solve the problems highlighted in this research. The changes can be made to standalone, desktop-based email clients as well as web-based email clients or interfaces which interact with the email server data, directly. However, it on the other hand, uses additional network resources to make the enhancements possible. On the other hand, the second proposed method, which requires enhancements to the SMTP protocol and the

SMTP Server and doesn't require any additional network resources, is very difficult to implement globally. Although, the solution has been practically implemented, tested and proven effective against these security and privacy threats, updating email Servers on a global scale is extremely difficult and not an overnight process, because the SMTP protocol is implemented by all Email Servers worldwide and the protocol has hardly been changed ever since its initial implementation. The proposed solution can be accommodated in future releases of web servers and will take time to implement globally.

This work can be taken forward to address the discussed shortcomings of the proposed methods and decrease the use of any additional resources. The methods of storing the email messages by the email client and the email server can be optimized to minimize the use of additional disk space. The recipient server can be modified to store copies of messages addressed to different users in ways that decrease the use of disk space.

REFERENCES

[1] F. A. Mir, M. T. Banday, "Control of spam: A comparative approach with special reference to India," Journal of Information Technology Law, vol. 19, no. 1, pp. 22-59, 2010. doi:10.1080/13600831003589350

[2] R. Kaur, S. Singh, H. Kumar, "Rise of spam and compromised accounts in online social networks: A state-of-the-art review of different combating approaches," Journal of Network and Computer Applications, vol. 112, pp. 53-88, 2018. doi:doi.org/10.1016/j.jnca.2018.03.015

[3] D. Wang, D. Irani, and C. Pu., "Study on evolution of email spam over fifteen years," 9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, Austin, TX, USA, 20-23 Oct. 2013. doi:10.4108/icst.collaboratecom.2013.254082

[4] Y. Y. Chen, S. Yong, A. Ishak. "Email hoax detection system using levenshtein distance method," Journal of Computers, vol. 9, no. 2, pp. 441-446, 2014. doi:10.4304/jcp.9.2.441-446

[5] E. P. Sanz, J. M. G. Hidalgo, J. C. C. Perez, "Chapter 3 email spam filtering," Advances in Computers, Elsevier, vol 74, pp. 45-114, pp. 45-114, doi:10.1016/S0065-2458(08)00603-7

[6] E. J. Williams, J. Hinds, A. N. Joinson, "Exploring susceptibility to phishing in the workplace," International Journal of Human-Computer Studies, vol. 120, pp. 1-13, 2018. doi:10.1016/j.ijhcs.2018.06.004

[7] I. AbdulNabi, Q. Yaseen, "Spam email detection using deep learning techniques," Procedia Computer Science, vol. 184, 2021. pp. 853-858, 2021. doi:10.1016/j.procs.2021.03.107

[8] Y. Kwaka, S. Leeb, A. Damianoc, A. Vishwanathd, "Why do users not report spear phishing emails?," Telematics and Informatics, Elsevier, vol. 48, 2020. doi:10.1016/j.tele.2020.101343

[9] B. B. Gupta, K. Yadav, I. Razzak, K. Psannis, A. Castiglione, X. Chang, "A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment," Computer Communications, vol. 175, pp. 47-57, 2021. doi:10.1016/j.comcom.2021.04.023

[10] P. Voigt, A. Bussche, "The EU general data protection regulation (GDPR)- A practical guide," Springer International Publishing AG, pp. 201-217, 2017. doi:10.1007/978-3-319-57959-7

[11] C. M. Ryan, M. R. Obrien, "Method and system for forcing e-mail addresses into blind carbon copy ("BCC") to enforce privacy," US Patent no. US9015252B2, 2015

[12] P. Resnick., "Internet message format," RFC 2822, 2001. https://tools.ietf.org/html/rfc2822

[13] A. Barth, D. Boneh, "Correcting privacy violations in blind-carbon-copy (BCC) encrypted email," 2005. http://crypto.stanford.edu/portia/papers/bb-bcc.pdf

[14] T. Haesevoets, D. D. Cremer, J. McGuire, "How the use of Cc, Bcc, forward, and rewrite in email communication impacts team dynamics," Computers in Human Behavior, vol. 112, 2020. doi:10.1016/j.chb.2020.106478

[15] J. Schwenk, M. Brinkmann, D. Poddebniak, J. Müller, J. Somorovsky, S. Schinzel, "Mitigation of attacks on email end-to-end encryption," Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications, CCS '20, pp. 1647–1664, 2020. doi:10.1145/3372297.3417878

[16] J. Postel, "Simple mail transfer protocol," RFC 821, 1982. https://tools.ietf.org/html/rfc5321

[17] J. Klensin, "Simple mail transfer protocol," RFC 5321, 2008. https://tools.ietf.org/html/rfc5321

[18] R. Siemborski, A. Melnikov, "SMTP service extension for authentication," RFC 4954, 2007. https://tools.ietf.org/html/rfc4954

[19] P. Hoffman, "SMTP Service Extension for Secure SMTP over Transport Layer Security," RFC 3207, 2002. https://tools.ietf.org/html/rfc3207

[20] S. Kaushik, P. Ammann, D. Wijesekera, W. Winsborough, R. Ritchey, "A policy driven approach to email services," 5th IEEE International Workshop on Policies for Distributed Systems and Networks. Yorktown Heights. NY. USA, 2004. doi:10.1109/POLICY.2004.1309163

[21] M. H. Haggag, "Enhanced delivery through a smart SMTP client," The International Journal on Intelligent Cooperative Information, vol. 4, no. 1, pp. 112-124, 2004

[22] R. Braden, "T/TCP - TCP extensions for transactions functional specification," RFC 1644, 1994. https://tools.ietf.org/html/rfc1644

[23] H. A. A. Bazar, S. Ramadass, O. Abuabdalla, "A new approach to enhance e-mail performance through SMTP protocol," IJCSNS International Journal of Computer Science and Network Security, vol. 8, no. 4, pp. 299-303, 2008

[24] A. Nand, T. L. Yu, "Mail servers with embedded data compression mechanisms," IEEE Computer Society, Proceedings of the Conference on Data Compression, vol. 1, pp. 566, 1998. doi:10.1109/DCC.1998.672308

[25] R. Siemborski, A. Menon-Sen, "The post office protocol (POP3)," RFC 5034, 2007, https://tools.ietf.org/html/rfc5034

[26] R. Sureswaran, H. A. Bazar, O. Abouabdalla, A. M. Manasrah, H. El-Taj, "Active e-mail system SMTP protocol monitoring algorithm," 2nd IEEE International Conference on Broadband Network & Multimedia Technology, 2009. doi:10.1109/ICBNMT.2009.5348490

[27] M. Karamollahi, C. Williamson, "Characterization of IMAPS email traffic," IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, Rennes, France, 2019. doi:10.1109/MASCOTS.2019.00030

[28] R. Nayak, S. A. Jiwani, B. Rajitham, "Spam email detection using machine learning algorithm," Materials Today: Proceedings, April 2021. doi:10.1016/j.matpr.2021.03.147

[29] A. Bhardwaj S. Goundar, "Security challenges for cloud-based email infrastructure," Network Security, vol. 2017, no.11, pp. 8-15, 2017. doi:10.1016/S1353-4858(17)30094-6